

IBM Z

*Hardware Management Console Web
Services API
Version 2.15.0*



Note:

Before you use this information and the product it supports, read the information in [“Safety” on page li](#), [Appendix D, “Notices,” on page 987](#), and *IBM Systems Environmental Notices and User Guide*, Z125-5823.

This edition, SC27-2638-02, applies to the IBM Z and IBM LinuxONE servers. This edition replaces SC27-2638-01.

There might be a newer version of this document in a **PDF** file available on **Resource Link**. Go to <http://www.ibm.com/servers/resourcelink> and click **Library** on the navigation bar.

© **Copyright International Business Machines Corporation 2019, 2020.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures.....	xiii
Tables.....	xxxv
Safety.....	li
Safety notices.....	li
World trade safety information.....	li
Laser safety information.....	li
Laser compliance.....	li
About this publication.....	lii
Related publications.....	lii
Related HMC and SE console information.....	lii
Revisions.....	lii
Accessibility.....	lii
Accessibility features.....	liv
Keyboard navigation.....	liv
Consult assistive technologies.....	liv
IBM and accessibility.....	liv
How to send your comments.....	liv
Part 1. Web Services API fundamentals.....	1
Chapter 1. Introduction.....	3
Overview.....	3
Components of the API.....	3
Enabling and accessing the API.....	4
Authentication and access control.....	5
Compatibility.....	5
API versioning.....	5
Allowable changes within a major version.....	6
Requirements on client applications.....	6
Summary of API version updates.....	7
Chapter 2. Base definitions.....	39
Data types.....	39
Input and output representation.....	40
Representing API data types in JSON.....	41
Chapter 3. Invoking API operations.....	43
HTTP protocol standard.....	43
Connecting to the API HTTP server.....	43
HTTP header field usage.....	43
Required request header fields.....	44
Optional request headers.....	44
Standard response headers.....	45
Additional response headers.....	46
Media types.....	46
HTTP status codes.....	47

Error response bodies.....	48
Common request validation reason codes.....	50
Common request processing reason codes.....	54
Use of chunked response encoding.....	54
Filter query parameters.....	55
Regular expression syntax.....	56
Chapter 4. Asynchronous notification.....	57
JMS basics.....	57
Connecting to the API message broker.....	57
Per-session notification topics.....	58
Notification message formats.....	59
Common message characteristics.....	59
Status change notification.....	61
Property change notification.....	62
Inventory change notification.....	63
Job completion notification.....	64
Log entry notification.....	64
Operating system message notification.....	65
Chapter 5. Data model definitions.....	67
Data model concepts.....	67
Objects in the data model.....	67
Properties in the data model.....	68
Shared data model schema elements.....	70
Base managed object properties schema.....	70
Chapter 6. Firmware features.....	73
Firmware feature concepts.....	73
dpm-storage-management.....	73
Part 2. General services.....	75
Chapter 7. General API services.....	77
General API services operations summary.....	77
Session management services.....	77
Query API Version.....	78
Logon.....	80
Establish Shared Secret Key.....	86
Provide Requested MFA Information.....	88
Change Logon Password.....	91
Logoff.....	93
Get Notification Topics.....	94
Request aggregation services.....	96
Submit Requests.....	96
Asynchronous job processing.....	103
Query Job Status.....	103
Delete Completed Job Status.....	106
Cancel Job.....	108
Chapter 8. Inventory and metrics services.....	111
Inventory services operations summary.....	111
Metrics service operations summary.....	111
Inventory service.....	112
Get Inventory.....	112
Metrics service.....	121
Create Metrics Context.....	121

Get Metrics.....	124
Delete Metrics Context.....	127
Chapter 9. Metric groups.....	129
Monitors dashboard metric groups.....	129
Channels.....	129
CPC overview.....	130
DPM system overview.....	132
Logical partitions.....	132
Partitions.....	133
zCPC environmentals and power.....	134
Power status.....	134
zCPC processors.....	137
Cryptos.....	138
Adapters.....	138
Flash memory adapters.....	138
RoCE adapters.....	139
Network management metrics.....	139
Network adapter port metric group.....	139
Network interface metric group.....	141
Part 3. CPC management.....	145
Chapter 10. Dynamic Partition Manager (DPM).....	147
FICON storage configuration.....	147
Operations summary.....	148
Partition operations summary.....	149
Adapter operations summary.....	151
Virtual Switch operations summary.....	152
Capacity Group operations summary.....	152
Storage Site operations summary.....	153
Storage Fabric operations summary.....	153
Storage Switch operations summary.....	153
Storage Subsystem operations summary.....	154
Storage Control Unit operations summary.....	155
Storage Group operations summary.....	155
Storage Template operations summary.....	157
Partition object.....	158
Data model.....	158
List Partitions of a CPC.....	180
List Permitted Partitions.....	182
Create Partition.....	185
Delete Partition.....	191
Delete Partition Asynchronously.....	193
Get Partition Properties.....	195
Update Partition Properties.....	198
Update Partition Properties Asynchronously.....	202
Start Partition.....	206
Stop Partition.....	210
Dump Partition.....	213
Start Dump Program.....	216
Perform PSW Restart.....	220
Create Virtual Function.....	223
Delete Virtual Function.....	225
Get Virtual Function Properties.....	227
Update Virtual Function Properties.....	228
Create NIC.....	230

Delete NIC.....	235
Get NIC Properties.....	237
Update NIC Properties.....	238
Increase Crypto Configuration.....	242
Change Crypto Domain Configuration.....	245
Decrease Crypto Configuration.....	247
Zeroize Crypto Domain.....	250
Mount ISO Image.....	252
Unmount ISO Image.....	254
Attach Storage Group to Partition.....	256
Detach Storage Group from Partition.....	258
Create HBA.....	260
Delete HBA.....	263
Update HBA Properties.....	265
Get HBA Properties.....	267
Reassign Storage Adapter Port.....	268
Send OS Command.....	271
Open OS Message Channel.....	272
List OS Messages of a Partition.....	274
Delete Partition OS Message.....	278
Get ASCII Console WebSocket URI.....	279
Inventory service data.....	281
Adapter object.....	283
Data model.....	284
List Adapters of a CPC.....	295
Get Adapter Properties.....	298
Update Adapter Properties.....	299
Change Crypto Type.....	302
Create Hipersocket.....	304
Delete Hipersocket.....	306
Get Partitions Assigned to Adapter.....	307
Get Network Port Properties.....	309
Update Network Port Properties.....	311
Get Storage Port Properties.....	313
Update Storage Port Properties.....	314
Change Adapter Type.....	316
Inventory service data.....	318
Virtual Switch object.....	319
Data model.....	319
List Virtual Switches of a CPC.....	321
Get Virtual Switch Properties.....	323
Get Connected VNICs of a Virtual Switch.....	324
Update Virtual Switch Properties.....	326
Inventory service data.....	327
Capacity Group element object.....	328
Data model.....	328
List Capacity Groups of a CPC.....	330
Create Capacity Group.....	332
Delete Capacity Group.....	334
Get Capacity Group Properties.....	336
Add Partition to Capacity Group.....	338
Remove Partition from Capacity Group.....	340
Update Capacity Group Properties.....	342
Inventory service data.....	343
Storage Site object.....	344
Data model.....	344
List Storage Sites.....	345
Create Storage Site.....	347

Delete Storage Site.....	350
Get Storage Site Properties.....	352
Update Storage Site Properties.....	353
Inventory service data.....	355
Storage Fabric object.....	356
Data model.....	356
List Storage Fabrics.....	357
Create Storage Fabric.....	359
Delete Storage Fabric.....	362
Get Storage Fabric Properties.....	363
Update Storage Fabric Properties.....	365
Inventory service data.....	367
Storage Switch object.....	367
Data model.....	368
List Storage Switches of a Storage Site.....	369
List Storage Switches of a Storage Fabric.....	371
Define Storage Switch.....	373
Undefine Storage Switch.....	376
Get Storage Switch Properties.....	377
Update Storage Switch Properties.....	379
Move Storage Switch to Storage Site.....	381
Move Storage Switch to Storage Fabric.....	383
Inventory service data.....	385
Storage Subsystem object.....	385
Data model.....	385
List Storage Subsystems of a Storage Site.....	387
Define Storage Subsystem.....	389
Undefine Storage Subsystem.....	392
Get Storage Subsystem Properties.....	393
Update Storage Subsystem Properties.....	395
Move Storage Subsystem to Storage Site.....	397
Add Connection Endpoint.....	399
Remove Connection Endpoint.....	401
Inventory service data.....	403
Storage Control Unit object.....	404
Data model.....	404
List Storage Control Units of a Storage Subsystem.....	407
Define Storage Control Unit.....	409
Undefine Storage Control Unit.....	411
Get Storage Control Unit Properties.....	413
Update Storage Control Unit Properties.....	414
Add Volume Range.....	416
Remove Volume Range.....	418
Create Storage Path.....	420
Delete Storage Path.....	423
Get Storage Path Properties.....	424
Update Storage Path Properties.....	426
Inventory service data.....	428
Storage Group object.....	429
Data model.....	432
List Storage Groups.....	447
Create Storage Group.....	450
Delete Storage Group.....	456
Get Storage Group Properties.....	459
Modify Storage Group Properties.....	461
Resend Request.....	469
Add Candidate Adapter Ports to an FCP Storage Group.....	472
Remove Candidate Adapter Ports from an FCP Storage Group.....	474

List Storage Volumes of a Storage Group.....	476
Get Storage Volume Properties.....	479
Fulfill FICON Storage Volume.....	481
Fulfill FICON Storage Volumes.....	484
Fulfill FCP Storage Volume.....	488
Accept Mismatched FCP Storage Volumes.....	490
Reject Mismatched FCP Storage Volumes.....	492
List Virtual Storage Resources of a Storage Group.....	495
Get Virtual Storage Resource Properties.....	498
Update Virtual Storage Resource Properties.....	499
Get Partitions for a Storage Group.....	502
Validate LUN Path.....	504
Start FCP Storage Discovery.....	506
Get Connection Report.....	509
Get Storage Group Histories.....	517
Inventory service data.....	531
Storage Template object.....	534
Data model.....	534
List Storage Templates.....	541
Create Storage Template.....	543
Delete Storage Template.....	546
Get Storage Template Properties.....	548
Modify Storage Template Properties.....	549
List Storage Template Volumes of a Storage Template.....	555
Get Storage Template Volume Properties.....	558
Inventory service data.....	559
Chapter 11. Core IBM Z resources.....	561
Operations Summary.....	561
Console operations summary.....	561
User operations summary.....	562
User Role operations summary.....	563
Task operations summary.....	564
User Pattern operations summary.....	564
Password Rule operations summary.....	564
LDAP Server Definition operations summary.....	565
MFA Server Definition operations summary.....	565
Group operations summary.....	566
CPC operations summary.....	567
Logical partition operations summary.....	568
Activation profile operations summary.....	570
Capacity record operations summary.....	571
Shared nested objects.....	572
Console object.....	576
Data model.....	576
Get Console Properties.....	582
Restart Console.....	588
Shutdown Console.....	589
Reorder User Patterns.....	591
Get Console Audit Log.....	593
Get Console Security Log.....	598
List Console Hardware Messages.....	601
Get Console Hardware Message Properties.....	603
Delete Console Hardware Message.....	605
Request Console Service.....	606
Get Console Service Request Information.....	608
Decline Console Service.....	610
List Unmanaged CPCs.....	611

Get Mobile App Preferences.....	613
Set Mobile App Preferences.....	615
Get CPC Notification Preferences for Device.....	618
Update CPC Notification Preferences for Device.....	622
Inventory service data.....	625
User-related-access permission.....	629
User object.....	630
Data model.....	630
List Users.....	637
Get User Properties.....	639
Update User Properties.....	641
Add User Role to User.....	644
Remove User Role from User.....	646
Create User.....	648
Delete User.....	652
Inventory service data.....	654
User Role object.....	655
Data model.....	656
List User Roles.....	659
Get User Role Properties.....	661
Update User Role Properties.....	663
Add Permission to User Role.....	665
Remove Permission from User Role.....	668
Create User Role.....	671
Delete User Role.....	673
Inventory service data.....	674
Task object.....	675
Data model.....	675
List Tasks.....	676
Get Task Properties.....	678
Inventory service data.....	679
User Pattern object.....	680
Data model.....	680
List User Patterns.....	682
Get User Pattern Properties.....	684
Update User Pattern Properties.....	685
Create User Pattern.....	687
Delete User Pattern.....	690
Inventory service data.....	691
Password Rule object.....	692
Data model.....	692
List Password Rules.....	695
Get Password Rule Properties.....	697
Update Password Rule Properties.....	699
Create Password Rule.....	701
Delete Password Rule.....	703
Inventory service data.....	704
LDAP Server Definition object.....	705
Data model.....	706
List LDAP Server Definitions.....	708
Get LDAP Server Definition Properties.....	710
Update LDAP Server Definition Properties.....	712
Create LDAP Server Definition.....	714
Delete LDAP Server Definition.....	716
Inventory service data.....	717
MFA Server Definition object.....	718
Data model.....	718
List MFA Server Definitions.....	719

Get MFA Server Definition Properties.....	721
Update MFA Server Definition Properties.....	723
Create MFA Server Definition.....	724
Delete MFA Server Definition.....	726
Inventory service data.....	727
Group Object.....	728
Data model.....	729
List Custom Groups.....	730
Get Custom Group Properties.....	731
Create Custom Group.....	733
Delete Custom Group.....	735
Add Member to Custom Group.....	736
Remove Member from Custom Group.....	738
List Custom Group Members.....	739
Inventory service data.....	741
CPC object.....	741
Data model.....	741
List CPC Objects.....	759
Get CPC Properties.....	761
Update CPC Properties.....	769
Start CPC.....	771
Stop CPC.....	773
Activate CPC.....	775
Deactivate CPC.....	778
Import Profiles.....	780
Export Profiles.....	781
Set Auto-Start List.....	783
Add Temporary Capacity.....	785
Remove Temporary Capacity.....	788
Swap Current Time Server.....	789
Set STP Configuration.....	791
Change STP-only Coordinated Timing Network.....	793
Join STP-only Coordinated Timing Network.....	794
Leave STP-only Coordinated Timing Network.....	796
Get CPC Audit Log.....	797
Get CPC Security Log.....	799
List CPC Hardware Messages.....	802
Get CPC Hardware Message Properties.....	805
Delete CPC Hardware Message.....	807
Request CPC Service.....	809
Get CPC Service Request Information.....	811
Decline CPC Service.....	813
Export WWPN List.....	814
Import DPM Configuration.....	817
Inventory service data.....	828
Logical Partition object.....	828
Data model.....	828
List Logical Partitions of CPC.....	850
List Permitted Logical Partitions.....	852
Get Logical Partition Properties.....	855
Update Logical Partition Properties.....	860
Activate Logical Partition.....	861
Deactivate Logical Partition.....	863
Reset Normal.....	865
Reset Clear.....	867
Load Logical Partition.....	869
PSW Restart.....	871
Start Logical Partition.....	873

Stop Logical Partition.....	874
Send OS Command.....	876
Open OS Message Channel.....	878
List OS Messages of a Logical Partition.....	880
Delete Logical Partition OS Message.....	883
SCSI Load.....	885
SCSI Dump.....	887
List Managed Virtual Machines of a Logical Partition.....	890
Inventory service data.....	892
Reset activation profile.....	893
Data model.....	893
List Reset Activation Profiles.....	894
Get Reset Activation Profile Properties.....	896
Update Reset Activation Profile Properties.....	898
Inventory service data.....	899
Image activation profile.....	900
Data model.....	900
List Image Activation Profiles.....	915
Get Image Activation Profile Properties.....	917
Update Image Activation Profile Properties.....	920
Inventory service data.....	922
Load activation profile.....	922
Data model.....	923
List Load Activation Profiles.....	925
Get Load Activation Profile Properties.....	927
Update Load Activation Profile Properties.....	930
Inventory service data.....	931
Group profile.....	931
Data model.....	932
List Group Profiles.....	933
Get Group Profile Properties.....	935
Update Group Profile Properties.....	937
Inventory service data.....	938
Capacity records.....	939
Data model.....	939
List Capacity Records.....	942
Get Capacity Record Properties.....	943
Inventory service data.....	944
Chapter 12. Energy management.....	945
Groups.....	946
Special states.....	947
Power saving.....	948
Group power saving.....	948
Power capping.....	948
Group capping.....	948
Energy management operations summary.....	949
Energy Management for CPC object.....	949
Data model.....	950
Operations.....	950
Set CPC Power Save.....	950
Set CPC Power Capping.....	952
Set zCPC Power Save.....	955
Set zCPC Power Capping.....	957
Get CPC Energy Management Data.....	959
Get Energy Optimization Advice Summary.....	960
Get Energy Optimization Advice Details.....	966

Appendix A. Enum values for a type of managed objects within User Roles.....	973
Appendix B. Enum values for the User Role object.....	975
Appendix C. Enum values for the Task object.....	977
Appendix D. Notices.....	987
Trademarks.....	987
Class A Notices.....	988
Index.....	993

Figures

1. Query API Version: Request.....	79
2. Query API Version: Response.....	80
3. Logon: Request.....	85
4. Logon: Response.....	86
5. Establish Shared Secret Key: Request.....	87
6. Establish Shared Secret Key: Response.....	88
7. Provide Requested MFA Information: Request.....	90
8. Provide Requested MFA Information: Response.....	90
9. Change Logon Password: Request.....	92
10. Change Logon Password: Response.....	92
11. Logoff: Request.....	93
12. Logoff: Response.....	94
13. Get Notification Topics: Request.....	95
14. Get Notification Topics: Response.....	96
15. Submit Requests: Request.....	102
16. Submit Requests: Response.....	102
17. Query Job Status: Request.....	106
18. Query Job Status: Response.....	106
19. Delete Completed Job Status: Request.....	107
20. Delete Completed Job Status: Response.....	107
21. Cancel Job: Request.....	109
22. Cancel Job: Response.....	109
23. Get Inventory: Request.....	116

24. Get Inventory: Response (Part 1).....	117
25. Get Inventory: Response (Part 2).....	118
26. Get Inventory: Response (Part 3).....	119
27. Get Inventory: Response (Part 4).....	120
28. Create Metrics Context: Request.....	123
29. Create Metrics Context: Response.....	124
30. Get Metrics: Request.....	127
31. Get Metrics: Response.....	127
32. Delete Metrics Context: Request.....	128
33. Delete Metrics Context: Response.....	128
34. List Partitions of a CPC: Request.....	181
35. List Partitions of a CPC: Response.....	182
36. List Permitted Partitions: Request.....	184
37. List Permitted Partitions: Response.....	185
38. Create Partition: Request.....	190
39. Create Partition: Response.....	190
40. Delete Partition: Request.....	192
41. Delete Partition: Response.....	192
42. Delete Partition Asynchronously: Request.....	195
43. Delete Partition Asynchronously: Response.....	195
44. Get Partition Properties: Request.....	196
45. Get Partition Properties: Response (Part 1).....	197
46. Get Partition Properties: Response (Part 2).....	198
47. Update Partition Properties: Request.....	202
48. Update Partition Properties: Response.....	202

49. Update Partition Properties Asynchronously: Request.....	206
50. Update Partition Properties Asynchronously: Response.....	206
51. Start Partition: Request.....	210
52. Start Partition: Response.....	210
53. Stop Partition: Request.....	212
54. Stop Partition: Response.....	213
55. Dump Partition: Request.....	216
56. Dump Partition: Response.....	216
57. Start Dump Program: Request.....	220
58. Start Dump Program: Response.....	220
59. Perform PSW Restart: Request.....	222
60. Perform PSW Restart: Response.....	222
61. Create Virtual Function: Request.....	225
62. Create Virtual Function: Response.....	225
63. Delete Virtual Function: Request.....	226
64. Delete Virtual Function: Response.....	226
65. Get Virtual Function Properties: Request.....	228
66. Get Virtual Function Properties: Response.....	228
67. Update Virtual Function Properties: Request.....	230
68. Update Virtual Function Properties: Response.....	230
69. Create NIC: Request.....	234
70. Create NIC: Response.....	235
71. Delete NIC: Request.....	236
72. Delete NIC: Response.....	236
73. Get NIC Properties: Request.....	238

74. Get NIC Properties: Response.....	238
75. Update NIC Properties: Request.....	242
76. Update NIC Properties: Response.....	242
77. Increase Crypto Configuration: Request.....	244
78. Increase Crypto Configurations: Response.....	245
79. Change Crypto Domain Configuration: Request.....	247
80. Change Crypto Domain Configuration: Response.....	247
81. Decrease Crypto Configuration: Request.....	249
82. Decrease Crypto Configuration: Response.....	250
83. Zeroize Crypto Domain: Request.....	252
84. Zeroize Crypto Domain: Response.....	252
85. Mount ISO Image: Request.....	254
86. Mount ISO Image: Response.....	254
87. Unmount ISO Image: Request.....	255
88. Unmount ISO Image: Response.....	256
89. Attach Storage Group to Partition: Request.....	258
90. Attach Storage Group to Partition: Response.....	258
91. Detach Storage Group from Partition: Request.....	260
92. Detach Storage Group from Partition: Response.....	260
93. Create HBA: Request.....	262
94. Create HBA: Response.....	263
95. Delete HBA: Request.....	264
96. Delete HBA: Response.....	264
97. Update HBA Properties: Request.....	266
98. Update HBA Properties: Response.....	267

99. Get HBA Properties: Request.....	268
100. Get HBA Properties: Response.....	268
101. Reassign Storage Adapter Port: Request.....	270
102. Reassign Storage Adapter Port: Response.....	270
103. Send OS Command: Request.....	272
104. Send OS Command: Response.....	272
105. Open OS Message Channel: Request.....	274
106. Open OS Message Channel: Response.....	274
107. List OS Messages of a Partition: Request.....	277
108. List OS Messages of a Partition: Response.....	277
109. Delete Partition OS Message: Request.....	279
110. Delete Partition OS Message: Response.....	279
111. Get ASCII Console WebSocket URI: Request.....	281
112. Get ASCII Console WebSocket URI: Response.....	281
113. Partition object: Sample inventory data - Response (Part 1).....	282
114. Partition object: Sample inventory data - Response (Part 2).....	283
115. Partition object: Sample inventory data - Response (Part 3).....	283
116. List Adapters of a CPC: Request.....	297
117. List Adapters of a CPC: Response.....	297
118. Get Adapter Properties: Request.....	299
119. Get Adapter Properties: Response.....	299
120. Update Adapter Properties: Request.....	301
121. Update Adapter Properties: Response.....	301
122. Change Crypto Type: Request.....	303
123. Change Crypto Type: Response.....	304

124. Create Hipersocket: Request.....	306
125. Create Hipersocket: Response.....	306
126. Delete Hipersocket: Request.....	307
127. Delete Hipersocket: Response.....	307
128. Get Partitions Assigned to Adapter: Request.....	309
129. Get Partitions Assigned to Adapter: Response.....	309
130. Get Network Port Properties: Request.....	310
131. Get Network Port Properties: Response.....	311
132. Update Network Port Properties: Request.....	312
133. Update Network Port Properties: Response.....	312
134. Get Storage Port Properties: Request.....	314
135. Get Storage Port Properties: Response.....	314
136. Update Storage Port Properties: Request.....	316
137. Update Storage Port Properties: Response.....	316
138. Change Adapter Type: Request.....	318
139. Change Adapter Type: Response.....	318
140. Adapter object: Sample inventory data.....	319
141. List Virtual Switches of a CPC: Request.....	322
142. List Virtual Switches of a CPC: Response.....	323
143. Get Virtual Switch Properties: Request.....	324
144. Get Virtual Switch Properties: Response.....	324
145. Get Connected VNICs of a Virtual Switch: Request.....	325
146. Get Connected VNICs of a Virtual Switch: Response.....	326
147. Update Virtual Switch Properties: Request.....	327
148. Update Virtual Switch Properties: Response.....	327

149. Virtual Switch object: Sample inventory data - Response.....	328
150. List Capacity Groups of a CPC: Request.....	331
151. List Capacity Groups of a CPC: Response.....	332
152. Create Capacity Group: Request.....	334
153. Create Capacity Group: Response.....	334
154. Delete Capacity Group: Request.....	336
155. Delete Capacity Group: Response.....	336
156. Get Capacity Group Properties: Request.....	337
157. Get Capacity Group Properties: Response.....	337
158. Add Partition to Capacity Group: Request.....	339
159. Add Partition to Capacity Group: Response.....	340
160. Remove Partition from Capacity Group: Request.....	341
161. Remove Partition from Capacity Group: Response.....	341
162. Update Capacity Group Properties: Request.....	343
163. Update Capacity Group Properties: Response.....	343
164. List Storage Sites: Request.....	347
165. List Storage Sites: Response.....	347
166. Create Storage Site: Request.....	349
167. Create Storage Site: Response.....	350
168. Delete Storage Site: Request.....	351
169. Delete Storage Site: Response.....	351
170. Get Storage Site Properties: Request.....	353
171. Get Storage Site Properties: Response.....	353
172. Update Storage Site Properties: Request.....	355
173. Update Storage Site Properties: Response.....	355

174. Storage Site object: Sample inventory data - Response.....	356
175. List Storage Fabrics: Request.....	359
176. List Storage Fabrics: Response.....	359
177. Create Storage Fabric: Request.....	361
178. Create Storage Fabric: Response.....	362
179. Delete Storage Fabric: Request.....	363
180. Delete Storage Fabric: Response.....	363
181. Get Storage Fabric Properties: Request.....	364
182. Get Storage Fabric Properties: Response.....	365
183. Update Storage Fabric Properties: Request.....	366
184. Update Storage Fabric Properties: Response.....	367
185. Storage Fabric object: Sample inventory data - Response.....	367
186. List Storage Switches of a Storage Site: Request.....	371
187. List Storage Switches of a Storage Site: Response.....	371
188. List Storage Switches of a Storage Fabric: Request.....	373
189. List Storage Switches of a Storage Fabric: Response.....	373
190. Define Storage Switch: Request.....	375
191. Define Storage Switch: Response.....	376
192. Undefine Storage Switch: Request.....	377
193. Undefine Storage Switch: Response.....	377
194. Get Storage Switch Properties: Request.....	378
195. Get Storage Switch Properties: Response.....	379
196. Update Storage Switch Properties: Request.....	380
197. Update Storage Switch Properties: Response.....	381
198. Move Storage Switch to Storage Site: Request.....	382

199. Move Storage Switch to Storage Site: Response.....	383
200. Move Storage Switch to Storage Fabric: Request.....	384
201. Move Storage Switch to Storage Fabric: Response.....	385
202. Storage Switch object: Sample inventory data - Response.....	385
203. List Storage Subsystems of a Storage Site: Request.....	389
204. List Storage Subsystems of a Storage Site: Response.....	389
205. Define Storage Subsystem: Request.....	391
206. Define Storage Subsystem: Response.....	391
207. Undefine Storage Subsystem: Request.....	393
208. Undefine Storage Subsystem: Response.....	393
209. Get Storage Subsystem Properties: Request.....	394
210. Get Storage Subsystem Properties: Response.....	395
211. Update Storage Subsystem Properties: Request.....	396
212. Update Storage Subsystem Properties: Response.....	397
213. Move Storage Subsystem to Storage Site: Request.....	398
214. Move Storage Subsystem to Storage Site: Response.....	399
215. Add Connection Endpoint: Request.....	401
216. Add Connection Endpoint: Response.....	401
217. Remove Connection Endpoint: Request.....	403
218. Remove Connection Endpoint: Response.....	403
219. Storage Subsystem object: Sample inventory data - Response.....	404
220. List Storage Control Units of a Storage Subsystem: Request.....	408
221. List Storage Control Units of a Storage Subsystem: Response.....	409
222. Define Storage Control Unit: Request.....	411
223. Define Storage Control Unit: Response.....	411

224. Undefine Storage Control Unit: Request.....	412
225. Undefine Storage Control Unit: Response.....	412
226. Get Storage Control Unit Properties: Request.....	414
227. Get Storage Control Unit Properties: Response.....	414
228. Update Storage Control Unit Properties: Request.....	416
229. Update Storage Control Unit Properties: Response.....	416
230. Add Volume Range: Request.....	418
231. Add Volume Range: Response.....	418
232. Remove Volume Range: Request.....	419
233. Remove Volume Range: Response.....	420
234. Create Storage Path: Request.....	422
235. Create Storage Path: Response.....	423
236. Delete Storage Path: Request.....	424
237. Delete Storage Path: Response.....	424
238. Get Storage Path Properties: Request.....	425
239. Get Storage Path Properties: Response.....	426
240. Update Storage Path Properties: Request.....	428
241. Update Storage Path Properties: Response.....	428
242. Storage Control Unit object: Sample inventory data - Response.....	429
243. List Storage Groups: Request.....	449
244. List Storage Groups: Response.....	450
245. Create Storage Group: Request.....	456
246. Create Storage Group: Response.....	456
247. Delete Storage Group: Request.....	459
248. Delete Storage Group: Response.....	459

249. Get Storage Group Properties: Request.....	460
250. Get Storage Group Properties: Response.....	460
251. Modify Storage Group Properties: Request.....	469
252. Modify Storage Group Properties: Response.....	469
253. Resend Request: Request.....	471
254. Resend Request: Response.....	472
255. Add Candidate Adapter Ports to an FCP Storage Group: Request.....	474
256. Add Candidate Adapter Ports to an FCP Storage Group: Response.....	474
257. Remove Candidate Adapter Ports from an FCP Storage Group: Request.....	476
258. Remove Candidate Adapter Ports from an FCP Storage Group: Response.....	476
259. List Storage Volumes of a Storage Group: Request.....	478
260. List Storage Volumes of a Storage Group: Response.....	479
261. Get Storage Volume Properties: Request.....	480
262. Get Storage Volume Properties: Response.....	481
263. Fulfill FICON Storage Volume: Request.....	484
264. Fulfill FICON Storage Volume: Response.....	484
265. Fulfill FICON Storage Volumes: Request.....	487
266. Fulfill FICON Storage Volumes: Response.....	487
267. Fulfill FCP Storage Volume: Request.....	490
268. Fulfill FCP Storage Volume: Response.....	490
269. Accept Mismatched FCP Storage Volumes: Request.....	492
270. Accept Mismatched FCP Storage Volumes: Response.....	492
271. Reject Mismatched FCP Storage Volumes: Request.....	495
272. Reject Mismatched FCP Storage Volumes: Response.....	495
273. List Virtual Storage Resources of a Storage Group: Request.....	497

274. List Virtual Storage Resources of a Storage Group: Response.....	497
275. Get Virtual Storage Resource Properties: Request.....	499
276. Get Virtual Storage Resource Properties: Response.....	499
277. Update Virtual Storage Resource Properties: Request.....	501
278. Update Virtual Storage Resource Properties: Response.....	502
279. Get Partitions for a Storage Group: Request.....	503
280. Get Partitions for a Storage Group: Response.....	504
281. Validate LUN Path: Request.....	506
282. Validate LUN Path: Response.....	506
283. Start FCP Storage Discovery: Request.....	508
284. Start FCP Storage Discovery: Response.....	509
285. Get Connection Report: Request.....	513
286. Get Connection Report: Response (Part 1).....	514
287. Get Connection Report: Response (Part 2).....	515
288. Get Connection Report: Response (Part 3).....	516
289. Get Connection Report: Response (Part 4).....	517
290. Get Storage Group Histories: Request.....	526
291. Get Storage Group Histories: Response (Part 1).....	527
292. Get Storage Group Histories: Response (Part 2).....	528
293. Get Storage Group Histories: Response (Part 3).....	529
294. Get Storage Group Histories: Response (Part 4).....	530
295. Get Storage Group Histories: Response (Part 5).....	531
296. Storage Group object: Sample inventory data - Response (Part 1).....	533
297. Storage Group object: Sample inventory data - Response (Part 2).....	534
298. List Storage Templates: Request.....	542

299. List Storage Templates: Response.....	543
300. Create Storage Template: Request.....	546
301. Create Storage Template: Response.....	546
302. Delete Storage Template: Request.....	547
303. Delete Storage Template: Response.....	547
304. Get Storage Template Properties: Request.....	549
305. Get Storage Template Properties: Response.....	549
306. Modify Storage Template Properties: Request.....	555
307. Modify Storage Template Properties: Response.....	555
308. List Storage Template Volumes of a Storage Template: Request.....	557
309. List Storage Template Volumes of a Storage Template: Response.....	558
310. Get Storage Template Volume Properties: Request.....	559
311. Get Storage Template Volume Properties: Response.....	559
312. Storage Template object: Sample inventory data - Response.....	560
313. Get Console Properties: Request.....	583
314. Get Console Properties: Response (Part 1).....	583
315. Get Console Properties: Response (Part 2).....	584
316. Get Console Properties: Response (Part 3).....	585
317. Get Console Properties: Response (Part 4).....	586
318. Get Console Properties: Response (Part 5).....	587
319. Get Console Properties: Response (Part 6).....	588
320. Shutdown Console: Request.....	591
321. Shutdown Console: Response.....	591
322. Reorder User Patterns: Request.....	592
323. Reorder User Patterns: Response.....	592

324. Get Console Audit Log: Request.....	595
325. Get Console Audit Log: Response (Part 1).....	596
326. Get Console Audit Log: Response (Part 2).....	597
327. Get Console Audit Log: Response (Part 3).....	598
328. Get Console Security Log: Request.....	599
329. Get Console Security Log: Response (Part 1).....	600
330. Get Console Security Log: Response (Part 2).....	601
331. List Console Hardware Messages: Request.....	603
332. List Console Hardware Messages: Response.....	603
333. Get Console Hardware Message Properties: Request.....	604
334. Get Console Hardware Message Properties: Response.....	605
335. Delete Console Hardware Message: Request.....	606
336. Delete Console Hardware Message: Response.....	606
337. Request Console Service: Request.....	608
338. Request Console Service: Response.....	608
339. Get Console Service Request Information: Request.....	609
340. Get Console Service Request Information: Response.....	610
341. Decline Console Service: Request.....	611
342. Decline Console Service: Response.....	611
343. List Unmanaged CPCs: Request.....	613
344. List Unmanaged CPCs: Response.....	613
345. Get Mobile App Preferences: Request.....	614
346. Get Mobile App Preferences: Response.....	615
347. Set Mobile App Preferences: Request.....	618
348. Set Mobile App Preferences: Response.....	618

349. Get CPC Notification Preferences for Device: Request.....	621
350. Get CPC Notification Preferences for Device: Response.....	621
351. Update CPC Notification Preferences for Device: Request.....	625
352. Update CPC Notification Preferences for Device: Response.....	625
353. Console object: Sample inventory data (Part 1).....	626
354. Console object: Sample inventory data (Part 2).....	627
355. Console object: Sample inventory data (Part 3).....	628
356. Console object: Sample inventory data (Part 4).....	629
357. List Users: Request.....	639
358. List Users: Response.....	639
359. Get User Properties: Request.....	640
360. Get User Properties: Response.....	641
361. Update User Properties: Request.....	644
362. Update User Properties: Response.....	644
363. Add User Role to User: Request.....	646
364. Add User Role to User: Response.....	646
365. Remove User Role from User: Request.....	648
366. Remove User Role from User: Response.....	648
367. Create User: Request.....	652
368. Create User: Response.....	652
369. Delete User: Request.....	653
370. Delete User: Response.....	654
371. User object: Sample inventory data.....	655
372. List User Roles: Request.....	660
373. List User Roles: Response.....	661

374. Get User Role Properties: Request.....	662
375. Get User Role Properties: Response.....	663
376. Update User Role Properties: Request.....	665
377. Update User Role Properties: Response.....	665
378. Add Permission to User Role: Request.....	668
379. Add Permission to User Role: Response.....	668
380. Remove Permission from User Role: Request.....	671
381. Remove Permission from User Role: Response.....	671
382. Create User Role: Request.....	672
383. Create User Role: Response.....	673
384. Delete User Role: Request.....	674
385. Delete User Role: Response.....	674
386. User Role object: Sample inventory data.....	675
387. List Tasks: Request.....	677
388. List Tasks: Response.....	678
389. Get Task Properties: Request.....	679
390. Get Task Properties: Response.....	679
391. Task object: Sample inventory data.....	680
392. List User Patterns: Request.....	683
393. List User Patterns: Response.....	684
394. Get User Pattern Properties: Request.....	685
395. Get User Pattern Properties: Response.....	685
396. Update User Pattern Properties: Request.....	687
397. Update User Pattern Properties: Response.....	687
398. Create User Pattern: Request.....	689

399. Create User Pattern: Response.....	689
400. Delete User Pattern: Request.....	691
401. Delete User Pattern: Response.....	691
402. User Pattern object: Sample inventory data.....	691
403. List Password Rules: Request.....	697
404. List Password Rules: Response.....	697
405. Get Password Rule Properties: Request.....	698
406. Get Password Rule Properties: Response.....	699
407. Update Password Rule Properties: Request.....	700
408. Update Password Rule Properties: Response.....	701
409. Create Password Rule: Request.....	703
410. Create Password Rule: Response.....	703
411. Delete Password Rule: Request.....	704
412. Delete Password Rule: Response.....	704
413. Password Rule object: Sample inventory data.....	705
414. List LDAP Server Definitions: Request.....	710
415. List LDAP Server Definitions: Response.....	710
416. Get LDAP Server Definition Properties: Request.....	711
417. Get LDAP Server Definition Properties: Response.....	712
418. Update LDAP Server Definition Properties: Request.....	713
419. Update LDAP Server Definition Properties: Response.....	713
420. Create LDAP Server Definition: Request.....	716
421. Create LDAP Server Definition: Response.....	716
422. Delete LDAP Server Definition: Request.....	717
423. Delete LDAP Server Definition: Response.....	717

424. LDAP Server Definition object: Sample inventory data.....	718
425. List MFA Server Definitions: Request.....	721
426. List MFA Server Definitions: Response.....	721
427. Get MFA Server Definition Properties: Request.....	722
428. Get MFA Server Definition Properties: Response.....	722
429. Update MFA Server Definition Properties: Request.....	724
430. Update MFA Server Definition Properties: Response.....	724
431. Create MFA Server Definition: Request.....	726
432. Create MFA Server Definition: Response.....	726
433. Delete MFA Server Definition: Request.....	727
434. Delete MFA Server Definition: Response.....	727
435. MFA Server Definition object: Sample inventory data.....	728
436. List Custom Groups: Request.....	731
437. List Custom Groups: Response.....	731
438. Get Custom Group Properties: Request.....	732
439. Get Custom Group Properties: Response.....	733
440. Create Custom Group: Request.....	734
441. Create Custom Group: Response.....	735
442. Delete Custom Group: Request.....	736
443. Delete Custom Group: Response.....	736
444. Add Member to Custom Group: Request.....	737
445. Add Member to Custom Group: Response.....	737
446. Remove Member from Custom Group: Request.....	739
447. Remove Member from Custom Group: Response.....	739
448. List Custom Group Members: Request.....	740

449. List Custom Group Members: Response.....	741
450. List CPC Objects: Request.....	761
451. List CPC Objects: Response.....	761
452. Get CPC Properties: Request.....	763
453. Get CPC Properties: Response (Part 1).....	764
454. Get CPC Properties: Response (Part 2).....	765
455. Get CPC Properties: Response (Part 3).....	766
456. Get CPC Properties: Response (Part 4).....	767
457. Get CPC Properties: Response (Part 5).....	768
458. Get CPC Properties: Response (Part 6).....	769
459. Set Auto-Start List: Request.....	785
460. Set Auto-Start List: Response.....	785
461. Get CPC Audit Log: Request.....	798
462. Get CPC Audit Log: Response.....	799
463. Get CPC Security Log: Request.....	801
464. Get CPC Security Log: Response.....	802
465. List CPC Hardware Messages: Request.....	804
466. List CPC Hardware Messages: Response.....	804
467. Get CPC Hardware Message Properties: Request.....	806
468. Get CPC Hardware Message Properties: Response.....	807
469. Delete CPC Hardware Message: Request.....	808
470. Delete CPC Hardware Message: Response.....	809
471. Request CPC Service: Request.....	810
472. Request CPC Service: Response.....	810
473. Get CPC Service Request Information: Request.....	812

474. Get CPC Service Request Information: Response.....	813
475. Decline CPC Service: Request.....	814
476. Decline CPC Service: Response.....	814
477. Export WWPN List: Request.....	816
478. Export WWPN List: Response.....	816
479. Import DPM Configuration: Request (Part 1).....	823
480. Import DPM Configuration: Request (Part 2).....	824
481. Import DPM Configuration: Request (Part 3).....	825
482. Import DPM Configuration: Request (Part 4).....	826
483. Import DPM Configuration: Request (Part 5).....	827
484. Import DPM Configuration: Response.....	827
485. List Logical Partitions of CPC: Request.....	852
486. List Logical Partitions of CPC: Response.....	852
487. List Permitted Logical Partitions: Request.....	854
488. List Permitted Logical Partitions: Response.....	855
489. Get Logical Partition Properties: Request.....	857
490. Get Logical Partition Properties: Response (Part 1).....	858
491. Get Logical Partition Properties: Response (Part 2).....	859
492. Get Logical Partition Properties: Response (Part 3).....	860
493. Send OS Command: Request.....	877
494. Send OS Command: Response.....	878
495. Open OS Message Channel: Request.....	879
496. Open OS Message Channel: Response.....	880
497. List OS Messages of a Logical Partition: Request.....	882
498. List OS Messages of a Logical Partition: Response.....	883

499. Delete Logical Partition OS Message: Request.....	884
500. Delete Logical Partition OS Message: Response.....	885
501. List Managed Virtual Machines of a Logical Partition: Request.....	892
502. List Managed Virtual Machines of a Logical Partition: Response.....	892
503. List Reset Activation Profiles: Request.....	895
504. List Reset Activation Profiles: Response.....	896
505. Get Reset Activation Profile Properties: Request.....	897
506. Get Reset Activation Profile Properties: Response.....	898
507. List Image Activation Profiles: Request.....	916
508. List Image Activation Profiles: Response.....	917
509. Get Image Activation Profile Properties: Request.....	919
510. Get Image Activation Profile Properties: Response (Part 1).....	919
511. Get Image Activation Profile Properties: Response (Part 2).....	920
512. List Load Activation Profiles: Request.....	927
513. List Load Activation Profiles: Response.....	927
514. Get Load Activation Profile Properties: Request.....	929
515. Get Load Activation Profile Properties: Response.....	929
516. List Group Profiles: Request.....	935
517. List Group Profiles: Response.....	935
518. Get Group Profile Properties: Request.....	936
519. Get Group Profile Properties: Response.....	937
520. Energy management as applied throughout layers of enterprise management.....	945
521. Example of a CPC group that contains a zCPC.....	947
522. Get Energy Optimization Advice Summary: Request.....	962
523. Get Energy Optimization Advice Summary: Response (Part 1).....	963

524. Get Energy Optimization Advice Summary: Response (Part 2).....	964
525. Get Energy Optimization Advice Summary: Response (Part 3).....	965
526. Get Energy Optimization Advice Summary: Response (Part 4).....	966
527. Get Energy Optimization Advice Details: Request.....	971
528. Get Energy Optimization Advice Details: Response.....	971

Tables

- 1. Summary of updates for API version 1.1 (HMC/SE Version 2.11.1)..... 7
- 2. Summary of updates for API version 1.2 (HMC/SE Version 2.11.1)..... 7
- 3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0)..... 8
- 4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1)..... 9
- 5. Summary of updates for API version 1.5 (HMC/SE Version 2.12.1)..... 11
- 6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0)..... 12
- 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1)..... 14
- 8. Summary of updates for API version 2.1 (HMC/SE Version 2.13.1)..... 19
- 9. Summary of updates for API version 2.2 (HMC/SE Version 2.13.1)..... 20
- 10. Summary of updates for API version 2.3 (HMC/SE Version 2.13.1)..... 20
- 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0)..... 21
- 12. Summary of updates for API version 2.21 (HMC/SE Version 2.14.0)..... 24
- 13. Summary of updates for API version 2.22 (HMC/SE Version 2.14.0)..... 24
- 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0)..... 24
- 15. Summary of updates for API version 2.24 (HMC/SE Version 2.14.0)..... 26
- 16. Summary of updates for API version 2.25 (HMC/SE Version 2.14.0)..... 27
- 17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1)..... 28
- 18. Summary of updates for API version 2.36 (HMC/SE Version 2.14.1)..... 29
- 19. Summary of updates for API version 2.37 (HMC/SE Version 2.14.1)..... 30
- 20. Summary of updates for API version 2.38 (HMC/SE Version 2.14.1)..... 30
- 21. Summary of updates for API version 2.39 (HMC/SE Version 2.14.1)..... 30
- 22. Summary of updates for API version 2.40 (HMC/SE Version 2.14.1)..... 30
- 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0)..... 31

24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0).....	35
25. Summary of updates for API version 3.3 (HMC/SE Version 2.15.0).....	36
26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0).....	37
27. Summary of updates for API version 3.5 (HMC/SE Version 2.15.0).....	38
28. Primitive data types.....	39
29. Compound data types.....	40
30. Primitive data types in JSON notation.....	41
31. Compound data types in JSON notation.....	41
32. error-feature-info object properties.....	54
33. General API services: operations summary.....	77
34. General API services: URI variables.....	77
35. Logon: HTTP status and reason codes.....	83
36. mfa-info-request nested object properties.....	84
37. change-password-info nested object properties.....	84
38. Provide Requested MFA Information: HTTP status and reason codes.....	89
39. mfa-requested-information-failures object.....	90
40. mfa-requested-information-failing-factor object.....	90
41. Change Logon Password: HTTP status and reason codes.....	92
42. topic-info object.....	94
43. Submit Requests: HTTP status and reason codes.....	101
44. Inventory service: operations summary.....	111
45. Metrics service: operations summary.....	111
46. Metrics service: URI variables.....	111
47. Channels metric group.....	129
48. CPC overview metric group.....	130

49. DPM system overview metric group.....	132
50. Logical partitions metric group.....	133
51. Partitions metric group.....	133
52. zCPC environmental and power metric group.....	134
53. Power status metric group.....	135
54. zCPC processors metric group.....	137
55. Crypto metric group.....	138
56. Adapters metric group.....	138
57. Flash memory adapters metric group.....	139
58. RoCE adapters metric group.....	139
59. Network adapter port metric group.....	140
60. Network interface metric group.....	142
61. DPM - Partition: operations summary.....	149
62. DPM - Partition: URI variables.....	150
63. DPM - Adapter: operations summary.....	151
64. DPM - Adapter: URI variables.....	151
65. DPM - Virtual Switch: operations summary.....	152
66. DPM - Virtual Switch: URI variables.....	152
67. DPM - Capacity Group: operations summary.....	152
68. DPM - Capacity Group: URI variables.....	152
69. DPM - Storage Site: operations summary.....	153
70. DPM - Storage Site: URI variables.....	153
71. DPM - Storage Fabric: operations summary.....	153
72. DPM - Storage Fabric: URI variables.....	153
73. DPM - Storage Switch: operations summary.....	153

74. DPM - Storage Switch: URI variables.....	154
75. DPM - Storage Subsystem: operations summary.....	154
76. DPM - Storage Subsystem: URI variables.....	155
77. DPM - Storage Control Unit: operations summary.....	155
78. DPM - Storage Control Unit: URI variables.....	155
79. DPM - Storage Group: operations summary.....	156
80. DPM - Storage Group: URI variables.....	157
81. DPM - Storage Template: operations summary.....	157
82. DPM - Storage Template: URI variables.....	157
83. Partition object: base managed object properties specializations.....	158
84. Partition object: class specific properties.....	160
85. partition-feature-info object properties.....	174
86. crypto-configuration nested object properties.....	174
87. crypto-domain-configuration nested object properties.....	175
88. Partition object - Virtual Function element properties.....	175
89. Partition object - NIC element object properties.....	176
90. Partition object - HBA element object properties.....	179
91. Create Partition: HTTP status and reason codes.....	189
92. Delete Partition: HTTP status and reason codes.....	191
93. Delete Partition Asynchronously: HTTP status and reason codes.....	194
94. Delete Partition Asynchronously: Job status and reason codes.....	194
95. Update Partition Properties: HTTP status and reason codes.....	200
96. Update Partition Properties Asynchronously: HTTP status and reason codes.....	204
97. Update Partition Properties Asynchronously: Job status and reason codes.....	204
98. Start Partition: Job status and reason codes.....	208

99. Stop Partition: Job status and reason codes.....	212
100. Dump Partition: HTTP status and reason codes.....	215
101. Dump Partition: Job status and reason codes.....	215
102. Start Dump Program: Job status and reason codes.....	220
103. Perform PSW Restart: Job status and reason codes.....	222
104. Create Virtual Function: HTTP status and reason codes.....	224
105. Delete Virtual Function: HTTP status and reason codes.....	226
106. Get Virtual Function Properties: HTTP status and reason codes.....	227
107. Update Virtual Function Properties: HTTP status and reason codes.....	229
108. Create NIC: HTTP status and reason codes.....	233
109. Delete NIC: HTTP status and reason codes.....	236
110. Get NIC Properties: HTTP status and reason codes.....	237
111. Update NIC Properties: HTTP status and reason codes.....	240
112. Increase Crypto Configuration: HTTP status and reason codes.....	243
113. Change Crypto Domain Configuration: HTTP status and reason codes.....	246
114. Decrease Crypto Configuration: HTTP status and reason codes.....	248
115. Zeroize Crypto Domain: HTTP status and reason codes.....	251
116. Mount ISO Image: HTTP status and reason codes.....	253
117. Unmount ISO Image: HTTP status and reason codes.....	255
118. Attach Storage Group to Partition: HTTP status and reason codes.....	257
119. Detach Storage Group from Partition: HTTP status and reason codes.....	259
120. Create HBA: HTTP status and reason codes.....	261
121. Delete HBA: HTTP status and reason codes.....	264
122. Update HBA Properties: HTTP status and reason codes.....	266
123. Get HBA Properties: HTTP status and reason codes.....	268

124. Reassign Storage Adapter Port: HTTP status and reason codes.....	270
125. Open OS Message Channel: HTTP status and reason codes.....	273
126. Adapter object: base managed object properties specializations.....	284
127. Adapter object: class-specific properties.....	285
128. Network Port element object properties.....	293
129. Storage Port element object properties.....	294
130. List Adapters of a CPC: HTTP status and reason codes.....	297
131. Update Adapter Properties: HTTP status and reason codes.....	300
132. Change Crypto Type: HTTP status and reason codes.....	303
133. Create Hipersocket: HTTP status and reason codes.....	305
134. Delete Hipersocket: HTTP status and reason codes.....	307
135. Get Partitions Assigned to Adapter: HTTP status and reason codes.....	309
136. Get Network Port Properties: HTTP status and reason codes.....	310
137. Update Network Port Properties: HTTP status and reason codes.....	312
138. Get Storage Port Properties: HTTP status and reason codes.....	313
139. Update Storage Port Properties: HTTP status and reason codes.....	315
140. Change Adapter Type: HTTP status and reason codes.....	317
141. Virtual Switch object: base managed object properties specializations.....	319
142. Virtual Switch object: class specific properties.....	320
143. Update Virtual Switch Properties: HTTP status and reason codes.....	327
144. Capacity Group element object properties.....	328
145. Create Capacity Group: HTTP status and reason codes.....	333
146. Delete Capacity Group: HTTP status and reason codes.....	335
147. Get Capacity Group Properties: HTTP status and reason codes.....	337
148. Add Partition to Capacity Group: HTTP status and reason codes.....	339

149. Remove Partition from Capacity Group: HTTP status and reason codes.....	341
150. Update Capacity Group Properties: HTTP status and reason codes.....	343
151. Storage Site object properties.....	344
152. Storage Site object: class specific properties.....	345
153. List Storage Sites: HTTP status and reason codes.....	347
154. Create Storage Site: HTTP status and reason codes.....	349
155. Delete Storage Site: HTTP status and reason codes.....	351
156. Get Storage Site Properties: HTTP status and reason codes.....	352
157. Update Storage Site Properties: HTTP status and reason codes.....	354
158. Storage Fabric object properties.....	356
159. Storage Fabric object: class specific properties.....	357
160. List Storage Fabrics: HTTP status and reason codes.....	359
161. Create Storage Fabric: HTTP status and reason codes.....	361
162. Delete Storage Fabric: HTTP status and reason codes.....	363
163. Get Storage Fabric Properties: HTTP status and reason codes.....	364
164. Update Storage Fabric Properties: HTTP status and reason codes.....	366
165. Storage Switch object: base managed object properties specializations.....	368
166. Storage Switch object: class specific properties.....	368
167. List Storage Switches of a Storage Site: HTTP status and reason codes.....	370
168. List Storage Switches of a Storage Fabric: HTTP status and reason codes.....	373
169. Define Storage Switch: HTTP status and reason codes.....	375
170. Undefine Storage Switch: HTTP status and reason codes.....	377
171. Get Storage Switch Properties: HTTP status and reason codes.....	378
172. Update Storage Switch Properties: HTTP status and reason codes.....	380
173. Move Storage Switch to Storage Site: HTTP status and reason codes.....	382

174. Move Storage Switch to Storage Fabric: HTTP status and reason codes.....	384
175. Storage Subsystem object: base managed object properties specializations.....	386
176. Storage Subsystem object: class specific properties.....	386
177. subsystem-connection-endpoint nested object properties.....	387
178. List Storage Subsystems of a Storage Site: HTTP status and reason codes.....	389
179. Define Storage Subsystem: HTTP status and reason codes.....	391
180. Undefine Storage Subsystem: HTTP status and reason codes.....	392
181. Get Storage Subsystem Properties: HTTP status and reason codes.....	394
182. Update Storage Subsystem Properties: HTTP status and reason codes.....	396
183. Move Storage Subsystem to Storage Site: HTTP status and reason codes.....	398
184. Add Connection Endpoint: HTTP status and reason codes.....	400
185. Remove Connection Endpoint: HTTP status and reason codes.....	402
186. Storage Control Unit object: base managed object properties specializations.....	404
187. Storage Control Unit object: class specific properties.....	405
188. Storage Control Unit object: volume-range nested object properties.....	406
189. Storage Control Unit object: storage path element object properties.....	406
190. List Storage Control Units of a Storage Subsystem: HTTP status and reason codes.....	408
191. Define Storage Control Unit: HTTP status and reason codes.....	410
192. Undefine Storage Control Unit: HTTP status and reason codes.....	412
193. Get Storage Control Unit Properties: HTTP status and reason codes.....	413
194. Update Storage Control Unit Properties: HTTP status and reason codes.....	415
195. Add Volume Range: HTTP status and reason codes.....	417
196. Remove Volume Range: HTTP status and reason codes.....	419
197. Create Storage Path: HTTP status and reason codes.....	421
198. Delete Storage Path: HTTP status and reason codes.....	424

199. Get Storage Path Properties: HTTP status and reason codes.....	425
200. Update Storage Path Properties: HTTP status and reason codes.....	427
201. Storage Group object: base managed object properties specializations.....	432
202. Storage Group object: class specific properties.....	432
203. Storage Volume element object properties.....	437
204. partition-volume-path-info nested object.....	445
205. Virtual Storage Resource element object properties.....	445
206. world-wide-port-name-info object: properties.....	447
207. List Storage Groups: HTTP status and reason codes.....	449
208. Create Storage Group: HTTP status and reason codes.....	454
209. created-object-info object.....	455
210. Delete Storage Group: HTTP status and reason codes.....	458
211. Get Storage Group Properties: HTTP status and reason codes.....	460
212. storage-volume-request-info nested object for "create" operations on "fc" storage volumes.....	462
213. storage-volume-request-info nested object for "create" operations on "fcp" storage volumes.....	463
214. storage-volume-request-info nested object for "modify" operations on "fc" storage volumes.....	463
215. storage-volume-request-info nested object for "modify" operations on "fcp" storage volumes.....	465
216. storage-volume-request-info nested object for "delete" operations on "fc" or "fcp" storage volumes.....	465
217. Modify Storage Group Properties: HTTP status and reason codes.....	467
218. Resend Request: HTTP status and reason codes.....	471
219. Add Candidate Adapter Ports to an FCP Storage Group: HTTP status and reason codes.....	473
220. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes.....	475
221. List Storage Volumes of a Storage Group: HTTP status and reason codes.....	478
222. Get Storage Volume Properties: HTTP status and reason codes.....	480
223. Fulfill FICON Storage Volume: HTTP status and reason codes.....	483

224. Fulfill FICON Storage Volumes: HTTP status and reason codes.....	486
225. Fulfill FCP Storage Volume: HTTP status and reason codes.....	489
226. Accept Mismatched FCP Storage Volumes: HTTP status and reason codes.....	491
227. Reject Mismatched FCP Storage Volumes: HTTP status and reason codes.....	494
228. List Virtual Storage Resources of a Storage Group: HTTP status and reason codes.....	497
229. Get Virtual Storage Resource Properties: HTTP status and reason codes.....	498
230. Update Virtual Storage Resource Properties: HTTP status and reason codes.....	500
231. Get Partitions for a Storage Group: HTTP status and reason codes.....	503
232. Validate LUN Path: HTTP status and reason codes.....	505
233. Start FCP Storage Discovery: HTTP status and reason codes.....	508
234. Start FCP Storage Discovery: Job status and reason codes.....	508
235. fcp-fabric-info nested object.....	510
236. world-wide-port-name-zone-info nested object.....	511
237. adapter-info nested object.....	511
238. fcp-storage-subsystem-info nested object.....	511
239. storage-configuration-info nested object.....	512
240. ficon-storage-subsystem-info nested object.....	512
241. Get Connection Report: HTTP status and reason codes.....	513
242. storage-group-history-info nested object.....	518
243. storage-group-action-info nested object.....	519
244. storage-group-configuration nested object.....	521
245. storage-volume-info nested object.....	523
246. virtual-storage-resource-info nested object.....	525
247. Get Storage Group Histories: HTTP status and reason codes.....	526
248. Storage Template object: base managed object properties specializations.....	534

249. Storage Template object: class specific properties.....	535
250. Storage Template Volume element object properties.....	537
251. List Storage Templates: HTTP status and reason codes.....	542
252. Create Storage Template: HTTP status and reason codes.....	545
253. Delete Storage Template: HTTP status and reason codes.....	547
254. Get Storage Template Properties: HTTP status and reason codes.....	548
255. storage-template-volume-request-info nested object for "create" operations on "fc" storage template volumes.....	550
256. storage-template-volume-request-info nested object for "create" operations on "fcp" storage template volumes.....	551
257. storage-template-volume-request-info nested object for "modify" operations on "fc" storage template volumes.....	551
258. storage-template-volume-request-info nested object for "modify" operations on "fcp" storage template volumes.....	552
259. storage-template-volume-request-info nested object for "delete" operations on "fc" or "fcp" storage template volumes.....	552
260. Modify Storage Template Properties: HTTP status and reason codes.....	554
261. List Storage Template Volumes of a Storage Template: HTTP status and reason codes.....	557
262. Get Storage Template Volume Properties: HTTP status and reason codes.....	559
263. Core Z resources - Console: operations summary.....	561
264. Core Z resources - Console: URI variables.....	562
265. Core Z resources - User: operations summary.....	562
266. Core Z resources - User: URI variables.....	563
267. Core Z resources - User Role: operations summary.....	563
268. Core Z resources - User Role: URI variables.....	563
269. Core Z resources - Task: operations summary.....	564
270. Core Z resources - Task: URI variables.....	564
271. Core Z resources - User Pattern: operations summary.....	564

272. Core Z resources - User Pattern: URI variables.....	564
273. Core Z resources - Password Rule: operations summary.....	564
274. Core Z resources - Password Rule: URI variables.....	565
275. Core Z resources - LDAP Server Definition: operations summary.....	565
276. Core Z resources - LDAP Server Definition: URI variables.....	565
277. Core Z resources - MFA Server Definition: operations summary.....	566
278. Core Z resources - MFA Server Definition: URI variables.....	566
279. Core Z resources - Group: operations summary.....	566
280. Core Z resources - Groups: URI variables.....	567
281. Core Z resources - CPC: operations summary.....	567
282. Core Z resources - CPC: URI variables.....	568
283. Core Z resources - Logical partitions: operations summary.....	569
284. Core Z resources - Logical partitions: URI variables.....	570
285. Core Z resources - Reset activation profile: operations summary.....	570
286. Core Z resources - Image activation profile: operations summary.....	570
287. Core Z resources - Load activation profile: operations summary.....	570
288. Core Z resources - Group profile: operations summary.....	571
289. Core Z resources - Activation profile: URI variables.....	571
290. Core Z resources - Capacity record: operations summary.....	571
291. Core Z resources - Capacity record: URI variables.....	571
292. ec-mcl-description object.....	572
293. action object.....	572
294. ec object.....	572
295. mcl object.....	573
296. stp-config object.....	573

297. stp-node object.....	573
298. psw-description object.....	574
299. zaware-network object.....	574
300. ssc-network object.....	574
301. ip-info object.....	575
302. network-ip-info object.....	575
303. absolute-capping object.....	576
304. Console object: base managed object properties specializations.....	576
305. Console object: class specific additional properties.....	576
306. network-info object properties.....	577
307. detailed-network-info properties.....	577
308. ipv4-info properties.....	578
309. ipv6-info properties.....	578
310. machine-info properties.....	578
311. hardware-message object properties.....	578
312. hardware-message-details base properties.....	579
313. hardware-message-details type-specific properties when the type value is "basic":.....	579
314. hardware-message-details type-specific properties when the type value is "common-problem":..	579
315. hardware-message-node-details:.....	580
316. hardware-message-data-details:.....	580
317. mobile-app-preferences object properties.....	580
318. action-settings object properties.....	582
319. Reorder User Patterns: HTTP status and reason codes.....	592
320. log-entry-info object properties.....	593
321. event-details-info object properties.....	594

322. event-data-item-info object properties.....	594
323. List Console Hardware Messages: HTTP status and reason codes.....	602
324. List Unmanaged CPCs: HTTP status and reason codes.....	612
325. User object: base managed object properties specializations.....	630
326. User object: class specific additional properties.....	631
327. List Users: HTTP status and reason codes.....	638
328. Update User Properties: HTTP status and reason codes.....	643
329. Add User Role to User: HTTP status and reason codes.....	645
330. Remove User Role from User: HTTP status and reason codes.....	647
331. Create User: HTTP status and reason codes.....	651
332. Delete User: HTTP status and reason codes.....	653
333. User Role object: base managed object properties specializations.....	656
334. User Role object: class specific additional properties.....	656
335. permission-info object properties.....	658
336. Update User Role Properties: HTTP status and reason codes.....	664
337. Add Permission to User Role: HTTP status and reason codes.....	667
338. Remove Permission from User Role: HTTP status and reason codes.....	670
339. Create User Role: HTTP status and reason codes.....	672
340. Delete User Role: HTTP status and reason codes.....	674
341. Task object: properties.....	675
342. User Pattern object: properties.....	680
343. Update User Pattern Properties: HTTP status and reason codes.....	686
344. Create User Pattern: HTTP status and reason codes.....	689
345. Password Rule object: properties.....	692
346. character-rule object properties.....	694

347. custom-character-set object properties.....	695
348. Update Password Rule Properties: HTTP status and reason codes.....	700
349. Create Password Rule: HTTP status and reason codes.....	702
350. LDAP Server Definition object: properties.....	706
351. Create LDAP Server Definition: HTTP status and reason codes.....	715
352. MFA Server Definition object: properties.....	718
353. Create MFA Server Definition: HTTP status and reason codes.....	726
354. Group object: base managed object properties specializations.....	729
355. Group object: class specific additional properties.....	729
356. match-info object properties.....	730
357. CPC object: base managed object properties specializations.....	741
358. CPC object: class specific additional properties.....	742
359. ipv6-info object properties.....	751
360. hardware-message object properties.....	751
361. cpc-feature-info object properties.....	752
362. auto-start-entry object base properties.....	752
363. auto-start-entry object type-specific properties when type value is "partition"	753
364. auto-start-entry object type-specific properties when type value is "partition-group"	753
365. CPC object: energy management related additional properties.....	753
366. Set Auto-Start List: HTTP status and reason codes.....	784
367. Get CPC Audit Log: HTTP status and reason codes.....	798
368. Get CPC Security Log: HTTP status and reason codes.....	800
369. List CPC Hardware Messages: HTTP status and reason codes.....	804
370. Get CPC Hardware Message Properties: HTTP status and reason codes.....	805
371. Delete CPC Hardware Message: HTTP status and reason codes.....	808

372. Export WWPN List: HTTP status and reason codes.....	816
373. cpc-info nested object	820
374. adapter-mapping-info nested object	820
375. Logical Partition object: base managed object properties specializations.....	828
376. Logical Partition object: class specific additional properties.....	829
377. central-storage-allocation nested object properties.....	849
378. expanded-storage-allocation nested object properties.....	850
379. Open OS Message Channel: HTTP status and reason codes.....	879
380. List Managed Virtual Machines of a Logical Partition: HTTP status and reason codes.....	891
381. Reset activation profile: properties.....	893
382. Image activation profile: properties.....	900
383. Load activation profile: properties.....	923
384. Group profile: properties.....	932
385. Capacity records: class-specific properties.....	939
386. caprec-proc-info object.....	941
387. caprec-target object.....	941
388. Energy management: operations summary.....	949
389. Energy management: URI variables.....	949
390. Get Energy Optimization Advice Summary: HTTP status and reason codes.....	962
391. Get Energy Optimization Advice Details: HTTP status and reason codes.....	970
392. Enum values for a class of managed objects.....	973
393. Enum values for the name property of User Role objects with a type of "system-defined"	975
394. Enum values for the name property of Task objects.....	977

Safety

Safety notices

Safety notices may be printed throughout this guide. **DANGER** notices warn you of conditions or procedures that can result in death or severe personal injury. **CAUTION** notices warn you of conditions or procedures that can cause personal injury that is neither lethal nor extremely hazardous. **Attention** notices warn you of conditions or procedures that can cause damage to machines, equipment, or programs.

World trade safety information

Several countries require the safety information contained in product publications to be presented in their translation. If this requirement applies to your country, a safety information booklet is included in the publications package shipped with the product. The booklet contains the translated safety information with references to the US English source. Before using a US English publication to install, operate, or service this product, you must first become familiar with the related safety information in the *Systems Safety Notices*, G229-9054. You should also refer to the booklet any time you do not clearly understand any safety information in the US English publications.

Laser safety information

All IBM Z® (Z) and IBM® LinuxONE (LinuxONE) models can use I/O cards such as FICON®, Open Systems Adapter (OSA), RoCE Express®, Integrated Coupling Adapter (ICA SR), zHyperLink Express, or other I/O features which are fiber optic based and utilize lasers (short wavelength or long wavelength lasers).

Laser compliance

All lasers are certified in the US to conform to the requirements of DHHS 21 CFR Subchapter J for Class 1 or Class 1M laser products. Outside the US, they are certified to be in compliance with IEC 60825 as a Class 1 or Class 1M laser product. Consult the label on each part for laser certification numbers and approval information.

Laser Notice: U.S. FDA CDRH NOTICE if low power lasers are utilized, integrated, or offered with end product systems as applicable. Complies with 21 CFR 1040.10 and 1040.11 except for conformance with IEC 60825-1 Ed. 3., as described in Laser Notice No. 56, dated May 8, 2019.

CAUTION: Data processing environments can contain equipment transmitting on system links with laser modules that operate at greater than Class 1 power levels. For this reason, never look into the end of an optical fiber cable or open receptacle. (C027)

CAUTION: This product contains a Class 1M laser. Do not view directly with optical instruments. (C028)

About this publication

This publication defines, for reference purposes, the external interface of the Hardware Management Console (HMC) Web Services Application Programming Interface (Web Services API) for IBM Z and IBM LinuxONE, HMC version 2.15.0. This document specifies the capabilities, input and output formats, and behaviors of the Web Services API as viewed by an application external to the HMC that is leveraging that interface.

Related publications

The following publications provide information which supplements the information found within this document:

- *Capacity On Demand User's Guide*, SC28-6985
- *Processor Resource/Systems Manager Planning Guide*, SB10-7175
- *z/VM CP Planning and Administration Guide*, SC24-6178
- *z/VM CP Commands and Utility Reference*, SC24-6175
- *Dynamic Partition Manager (DPM) Guide*, SB10-7176
- *8561 Installation Manual for Physical Planning*, GC28-7002
- *8562 Installation Manual for Physical Planning*, GC28-7011
- *3907 Installation Manual for Physical Planning*, GC28-6974
- *3906 Installation Manual for Physical Planning*, GC28-6965
- *z13[®] Installation Manual for Physical Planning*, GC28-6938

Related HMC and SE console information

Hardware Management Console (HMC) and Support Element (SE) information can be found on the console help system.

Revisions

A technical change from the previous edition of this document is indicated by a thick vertical line to the left of the change.

For more information about what has changed since the last publication, see [“Summary of API version updates”](#) on page 7.

Accessibility

Accessible publications for this product are offered in EPUB format and can be downloaded from Resource Link[®] at <http://www.ibm.com/servers/resourcelink>.

If you experience any difficulty with the accessibility of any IBM Z and IBM LinuxONE information, go to Resource Link at <http://www.ibm.com/servers/resourcelink> and click **Feedback** from the navigation bar on the left. In the **Comments** input area, state your question or comment, the publication title and number, choose **General comment** as the category and click **Submit**. You can also send an email to reslink@us.ibm.com providing the same information.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Accessibility features

The following list includes the major accessibility features in IBM Z and IBM LinuxONE documentation, and on the Hardware Management Console and Support Element console:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Customizable display attributes such as color, contrast, and font size
- Communication of information independent of color
- Interfaces commonly used by screen magnifiers
- Interfaces that are free of flashing lights that could induce seizures due to photo-sensitivity.

Keyboard navigation

This product uses standard Microsoft Windows navigation keys.

Consult assistive technologies

Assistive technology products such as screen readers function with our publications, the Hardware Management Console, and the Support Element console. Consult the product information for the specific assistive technology product that is used to access the EPUB format publication or console.

IBM and accessibility

See <http://www.ibm.com/able> for more information about the commitment that IBM has to accessibility.

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. Send your comments by using Resource Link at <http://www.ibm.com/servers/resourcelink>. Click **Feedback** on the navigation bar on the left. You can also send an email to reslink@us.ibm.com. Be sure to include the name of the book, the form number of the book, the version of the book, if applicable, and the specific location of the text you are commenting on (for example, a page number, table number, or a heading).

Part 1. Web Services API fundamentals

Topics in this part describe the fundamentals of Web Services API.

Topics covered in this part are:

- [Chapter 1, “Introduction,” on page 3](#)
- [Chapter 2, “Base definitions,” on page 39](#)
- [Chapter 3, “Invoking API operations,” on page 43](#)
- [Chapter 4, “Asynchronous notification,” on page 57](#)
- [Chapter 5, “Data model definitions,” on page 67](#)
- [Chapter 6, “Firmware features,” on page 73](#)

Chapter 1. Introduction

This chapter provides an overview of IBM z Unified Resource Manager (zManager) APIs, how to enable and access them, and considerations for compatibility.

Overview

The IBM z Unified Resource Manager (zManager) is a collection of advanced hardware and virtualization management functions delivered as Z firmware. The functions of zManager are implemented as a cooperating set of components hosted on the Hardware Management Console (HMC), the Support Element (SE), and the Z CPC. It provides a uniform, integrated administrative model for the heterogeneous computing configuration provided by a Z environment. The functions provided by zManager include:

- Hardware inventory, initialization, configuration, monitoring and problem analysis for the components of a Z CPC.
- Firmware installation and update for the HMC, SE, and traditional CPC components.
- Operational control and energy management for these hardware elements.

The HMC serves as the administrative access point for zManager. In that capacity, the HMC provides a web-based, remote-able graphical user interface (UI) to make the zManager functions available to users. In addition, it hosts the implementation of zManager Web Service API (Web Services API) that is described in this document.

The Web Services API is a web-oriented programming interface that makes the underlying zManager capabilities available for use by higher level management applications, system automation functions, or custom scripting. The functions that are exposed through the API support several important usage scenarios in virtualization management, including resource inventory, provisioning, monitoring, automation and workload-based optimization among others.

Components of the API

The Web Services API consists of two major components. Both components are accessed by client applications by establishing TCP/IP network connections with the HMC.

Web services interface

The web services interface is a request-and-response oriented programming interface by which client applications obtain information about the system resources managed by zManager, and by which those applications can perform provisioning, configuration or control actions on those resources.

As is the case for any web-oriented interface, client applications interact with this interface by means of the Hypertext Transfer Protocol (HTTP), an application protocol that flows over TCP/IP socket connections. Client applications request operations by forming and sending text-oriented request messages as defined by HTTP, and the Web Services API responds with text-oriented HTTP response messages. The use of HTTP makes the API client-programming-language neutral, and thus accessible to a wide variety of client applications. Client applications can be developed in programming languages such as Java, or in scripting languages such as Perl or Python that include extensive support for performing HTTP operations.

The design of the API's mapping to HTTP has been influenced by the Representational State Transfer (REST) style of interface design. The manageable resources of the system are associated with and identified by durable URIs, and the basic get, update, create and delete operations on those manageable resources are mapped directly to the HTTP GET, PUT, POST and DELETE methods. Request and response data is provided using JavaScript Object Notation (JSON), a simple, open and portable transfer representation. Mapping the functions of the API to HTTP in this way simplifies client application

development and allows access to the API without the need for extensive client side tooling or libraries as is often the case in other approaches to web services interface design.

Broadly speaking, the web service interface provides two categories of operations:

- Resource (or object) oriented operations, in which a particular request is targeted at a single manageable resource instance and typically affects just that single resource instance. The majority of the API has this orientation, for example providing functions for interacting with the virtual servers, virtualization hosts, virtual networks and workloads of the system.
- Service oriented operations, in which a particular request operates across many or all manageable resources of the system. The service-oriented operations are provided to support usage scenarios that cannot be accomplished efficiently using an object-by-object sequence of individual requests. The operations provided by the Metrics and Inventory services of the API are examples of service-oriented operations.

Asynchronous notification facility

The web services interface described above is useful to satisfy many usage scenarios, particularly those in which the client application's interest in and interaction with zManager is focused on performing a short-term task. In these kinds of applications (typical of automation or simple provisioning), the client application forms a request, gets a response, processes the response and then "forgets" about the zManager resource it interacted with. That is, the application does not attempt to retain (or cache) information about zManager resources long term and then keep that cache up to date.

However, more sophisticated management applications, including those for discovery, monitoring and advanced provisioning, are not single-request-and-forget with respect to their interest in zManager. Rather, such applications have a need to obtain and retain (i.e., cache) information about the inventory, configuration and status of many zManager resources, and to keep that cached information up to date.

In order to support these more sophisticated applications, the Web Services API provides an asynchronous notification facility by which zManager can inform interested client applications about changes to the resources managed by zManager.

The API's asynchronous notification facility is designed around the Java Message Service (JMS), an open, standard framework and API for sending messages between two or more applications.

Enabling and accessing the API

The Web Services API is provided on an HMC that is running with firmware version 2.11.1 or later. The API can be used to query, configure and control Central Processing Complexes (CPCs) containing a Support Element (SE) that is running with firmware version 2.11.1 or higher.

By default, the Web Services API is disabled on the HMC. When disabled, the HMC internal firewall is configured to prohibit connections to any of the TCP/IP ports used by the API. When in this state, requests to connect to the API network ports are completely ignored by the HMC without a connection-refused response.

The Web Services API can be enabled and the scope of access to it configured using the **Customize API Settings** task in the HMC UI. The **Customize API Settings** task allows an installation to enable the API via an overall enabled/disabled setting. When enabled, the HMC internal firewall is reconfigured to allow access to the relevant network ports. When the API is enabled with default settings, the HMC allows connections to the API functions from client applications accessing the HMC from any TCP/IP address. For additional security, an installation can configure the HMC to permit connections to the API ports only from selected network addresses or subnets. These addresses or subnets are specified by the **Customize API Settings** task as well. If specified, these connection restrictions are enforced by the HMC internal firewall.

In addition to the overall enablement on/off control and the optional client network address filtering, access to the API is further secured by the requirement for per-user authorization.

The HMC **User Management** task defines access and other characteristics of an HMC user. This task manages a user property (**Allow access to Web Services management interfaces**) to indicate whether a particular HMC user is to be permitted to use the API or not. By default, this setting is disabled for an HMC user profile and thus attempts to establish an API session by that user are rejected. The installation can use the **Customize API Settings** or **User Management** tasks of the HMC to set this property for one or more HMC users and thus allow those users to access the API.

Once a user is permitted to establish API sessions, its actions within those sessions are subject to the HMC's access control model, as is described in the section that follows.

Authentication and access control

The HMC provides a built-in access control model in which an HMC user authenticates itself to the HMC to establish its identity, and then based on that identity is permitted or denied the ability to perform certain operations as specified by the access control configuration. These operations, and the objects on which they are permitted, are managed with object and task/action permissions that are grouped into roles that are assigned to HMC users. Roles are managed and assigned to users with the **User Management** task on the HMC.

Use of the Web Services API is subject to the same access control policy as is used for UI operations.

Establishing an API session with the HMC requires the initiating application to provide a valid HMC logon ID and corresponding password in order to authenticate and establish the identity under which its requests will be performed. (See “Logon” on page 80 for more information.) The API requires the use of SSL connections so that these login credentials can be flowed securely. The user credentials are validated by the HMC in the same way they are validated for a logon to the UI, either via the HMC's built-in user registry or by use of an LDAP directory server. If the HMC logon ID is configured to require multi-factor authentication, then an additional authentication token is required.

Once a client application has established an API session, its ability to access various managed object instances and the operations that can be performed on those instances is regulated based on the identity associated with the API session and the access control policy configured in the HMC for those managed object instances. Access control requirements vary based on the class of managed object and the operation for the managed object. These access control requirements for API actions mirror the requirements for corresponding tasks in the HMC user interface. Details on the authorization requirements for an operation are specified in the description of that operation.

Compatibility

The capabilities of the Web Services API will evolve as additional management functionality is added to zManager. Over time, this evolution could result in a mixture of HMC and client application versions coexisting in a customer environment. The principles and guidelines outlined in this section are intended to maximize the compatibility and interoperability among HMC and client applications in such a mixed environment.

API versioning

Since the functionality of the Web Services API may evolve over time, each functional level of the API is identified by a version number. This version number is represented in major.minor form, with the initial version of the API designated as version 1.1.

The API version offered by an HMC can be determined before API logon by using the Query API Version operation (GET /api/version). The version number of the API is also provided in the response from the Logon operation.

Enhancements to the API specification that maintain compatibility with previous versions (see principles below) are indicated by incrementing the minor portion of the version number. So, for example, the first set of compatible changes to the API would be designated as version 1.2, following the initial 1.1 version.

Because the minor versions within a major version stream (e.g. the 1.x versions) are considered compatible, the HMC always offers and behaves according to the latest minor version of the API

specification it supports. That means, for example, the API does not offer any facility by which a client can request version 1.1 behaviors on an HMC that offers version 1.2 level of functionality.

While reasonable effort will be made to preserve compatibility, it may become necessary to make changes to zManager (and thus the API) that do not maintain compatibility with the previous version. If this occurs, the introduction of this new (incompatible) behavior is indicated by incrementing the major part of the version number, and starting the minor part of the version number again at 1. The first such version would thus be identified as version 2.1.

Allowable changes within a major version

The following kinds of changes to the API specification are allowable within a major version, and thus result in changes to the minor but not major parts of the API version number.

- Adding new object classes or new operations on existing object classes.
- Adding new properties to the data model of an existing object class.
- Changing existing properties of an existing object class from read-only or mutable to writable using the API.
- Adding new URIs and operations related to those URIs.
- Adding new optional query parameters to existing URIs where the behavior in the absence of this query parameter is unchanged.
- Adding new optional fields into input bodies where the behavior in the absence of these new fields is unchanged.
- Adding new fields to the response bodies of existing operations.
- Adding additional header or body fields to existing notification messages.
- Adding data for new classes of objects to the results provided by the Inventory service.
- Generating new types of notification messages.
- Generating property change notifications for new properties, or for existing properties that did not provide those notifications previously.
- Adding new enumeration values to enumeration-type fields returned in response bodies without removing or changing the meaning of any existing enumeration values.
- Adding new error status and reason codes.
- Adding new metric groups.
- Adding new metric fields to the end of existing metric groups.

Requirements on client applications

In order for a client application to correctly interoperate with an HMC that may be offering a higher minor version of the API, client applications must be designed and developed following the simple principle of "ignore what you don't understand" when interpreting responses or messages received from the HMC. This is necessary because the principles of allowable changes specified in the preceding section allow new fields to be added to preexisting responses or messages.

More specifically, a client application must:

- Ignore, without error, any field in a response body that is not recognized by the application.
- Ignore, without error, any header or body field in a notification message that is not recognized by the application.
- Ignore, without error, any notification message of an unrecognized type that may be received by the application.
- Ignore, without error, any object appearing in the response to an Inventory request that are of a class not recognized by the application.

- Tolerate receiving a value in a field with an enumeration data type that is an unexpected value. If the application is attempting to display this field, it might consider mapping the unrecognized enumeration value to some value indicating "other" or "unknown".
- Ignore, without error, extra values provided in a row of metric group data that reside beyond the last field currently expected by the application.

These conditions can arise as a result of API extensions that are considered allowable within a given major version. Following the "ignore what you don't understand" principle prepares a client application to tolerate these API additions should they occur.

Summary of API version updates

The following functions were introduced in the respective API version:

Note: For each of the following API version summary tables, when an API extension indicates the addition of new properties to the data model for a specified object class, such an extension also includes standard changes to several related operations as well even though, for brevity, these related changes are not specifically mentioned in the table. In general, an extension to an object's data model will also include corresponding changes to the inputs to or responses from `Get Properties`, `Update Properties` and `Create` operations for that object class, as appropriate. In addition, the new properties are included in Inventory Service data for objects of the specified class.

Table 1. Summary of updates for API version 1.1 (HMC/SE Version 2.11.1)		
Description	HMC MCL	SE MCL
Added reason codes 0, 105 and 108 as possible HTTP status code 409 (Conflict) error conditions reported by the <code>Migrate Virtual Server</code> operation.	N48180.278	N48168.275
Changed the backing-virtualization-host-storage-resource property of the Virtual Server data model (in the <code>fullpack-virtual-disk</code> nested object) from a read-only property to a writable property.	N48180.287	None
Increased the maximum request body size for the <code>Import Storage Access List</code> operation to 1 MB, and increased the maximum request body size for most other operations to 64KB.	N48180.288	N48168.294
Added the cores-per-processor property to the Blade object data model (read-only).	N48180.296	None
Added inventory and property-change notification support for the Virtualization Host Storage Resource object.	N48180.308	N48168.315
Added the inventory-error-details field and related <code>inventory-error-info</code> nested object to the <code>inventory-error</code> document returned by the <code>Get Inventory</code> operation when error condition 5 is encountered.	N48180.314	N48168.321
• Added the properties=common query parameter to the <code>Get Virtual Server Properties</code> operation. • Added the <code>virtual-server-common</code> category and <code>power-vm-virtual-server-common</code>, <code>prsm-virtual-server-common</code>, <code>x-hyp-virtual-server-common</code> and <code>zvm-virtual-server-common</code> classes to the <code>Get Inventory</code> operation.	N48180.319	N48168.325
Changed the resources field in the request body for the <code>Get Inventory</code> operation from required to optional.	N48180.340	None
Corrected the range checking for the load-address field of the <code>Load Logical Partition</code> operation so that the operation correctly supports loading from an alternate subchannel.	N48180.360	None

Table 2. Summary of updates for API version 1.2 (HMC/SE Version 2.11.1)		
Description	HMC MCL	SE MCL
• Added the <code>Mount Virtual Media Image</code> operation. • Increased API version number from 1.1 to 1.2.	N48180.361	None

Table 2. Summary of updates for API version 1.2 (HMC/SE Version 2.11.1) (continued)		
Description	HMC MCL	SE MCL
Corrected the format of the URI returned by the List Members of Virtual Network operation for a zBX TOR port to reflect the correct canonical URI format for zBX TOR port elements.	N48180.363	None
Added the power-vm-partition-id property to the Virtual Server object data model for PowerVM® virtual servers (read-only)	N48180.363	N48168.378
Added HTTP status code 409 (Conflict) as a possible error condition for the List Virtualization Host Storage Resources and Get Virtualization Host Storage Resource Properties operations.	N48180.376	None
Added the feature-list property to the Virtualization Host object data model. This property is provided for virtualization hosts on all CPCs supported by the Web Services API, but the particular features provided by a given virtualization host will differ based on the release and MCL level of the CPC.	N48180.380	N48168.402

Table 3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.0, and apply to all CPCs supported by the Web Services API: - Increased API version number from 1.2 to 1.3. - Added the power-saving-state property to the BladeCenter and Blade objects data models, and added the cpc-power-saving-state and zcpc-power-saving-state properties to the CPC object data model. - Added "not-supported" as a possible enumeration value for the power-save-allowed and power-cap-allowed properties of BladeCenter and Blade objects, and added "not-supported" as a possible enumeration value of the cpc-power-save-allowed, cpc-power-cap-allowed, zcpc-power-save-allowed and zcpc-power-cap-allowed properties of the CPC object. - Added the status (read-only), acceptable-status (writable), perf-status (read-only) and compliant-perf-status (writable) properties to the Workload Resource Group object data model. - Added most-severe-perf-status and perf-status-data-points fields, and related perf-status-data-point nested object to the response from the Generate Workload Resource Groups Report operation. - Added the perf-policies property to the Virtual Server object data model, and also added related virtual server performance policy nested object. - Added "data-retrieval-error" as a possible enumeration value for the status-detailed field in the response for the Generate Load Balancing Report operation. - Added an optional request body containing an optional force input field to the Unmount Virtual Media operation. - Changed the Create Virtual Server operation for a zVM virtual server to require the password field on input rather than allowing it to be optional. This change has been made to improve security. - Added HTTP status code 409 (Conflict) as a possible error response reported by the following Storage Management operations: - Import Storage Access List - Create Virtualization Host Storage Resource - Delete Virtualization Host Storage Resource - Add Virtualization Host Storage Resource Paths - Remove Virtualization Host Storage Resource Paths - Discover Virtualization Host Storage Resources.	H09182.023	H09173.028

Table 3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0) (continued)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.0, but apply only to CPCs with SE version 2.12.0: - Added cp-cpu-consumption-percent, ifl-cpu-consumption-percent and other-cpu-consumption-percent fields to the response from the Generate Hypervisor Report operation for zVM virtualization hosts. These new fields are provided for zVM virtualization hosts running at version 6.2 or greater. - Added the following in support of IBM zAware partitions. These changes apply only for partitions of the new "zaware" type: - Added "zaware" as a possible value of the activation-mode property of Logical Partition and Image Activation Profile objects. - Added the zaware-network, network-ip-info and ip-info nested objects as common nested object definitions used for new properties of Logical Partition objects. - Added the zaware-host-name, zaware-master-userid, zaware-master-pw, zaware-network-info, zaware-gateway-info and zaware-dns-info properties to the Logical Partition and Image Activation Profile object data models. - Added HTTP status 400 reason code 306 as a possible error response from the Load Logical Partition, PSW Restart, Start Logical Partition, Stop Logical Partition, and Update Image Activation Profile Properties operations when these operations are attempted on an IBM zAware partition. - Added the Cryptos and Flash Memory Adapters metric groups for CPC objects. Data entries are provided for a CPC in these metric groups if the CPC has one or more Cryptos or Flash Memory Adapters installed. - Added new cp-cpu-time, ifl-cpu-time, zaap-cpu-time, ziip-cpu-time and icf-cpu-time metrics to the Virtualization Host CPU and Memory metric group (for zVM virtualization hosts).	H09182.023	H09173.028
Added HTTP status code 409 (Conflict) as a possible error condition for the List Virtualization Host Storage Resources and Get Virtualization Host Storage Resource Properties operations.	H09182.062	None
Added property change support for the unique-device-id property of the Storage Resource object.	H09182.102	None
Added the feature-list property to the Virtualization Host object data model. This property is provided for virtualization hosts on all CPCs supported by the Web Services API, but the particular features provided by a given virtualization host will differ based on the release and MCL level of the CPC.	H09182.119	H09173.149

Table 4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.1 or later, and apply to all CPCs supported by the Web Services API: - Increased API version number from 1.3 to 1.4. - Added the Get Network Adapter Properties operation for Virtual Server objects. - Added the List Virtualization Host Storage Resources of a Storage Resource operation for Storage Resource objects. - Added the List Virtual Disks of a Virtualization Host Storage Resource operation for Virtualization Host objects. - Added API support for absolute capping to the Logical Partition and Image Activation Profile objects, including the following API extensions: - Added the absolute-processing-capping, absolute-aap-capping, absolute-ifl-capping, absolute-ziip-capping and absolute-cf-capping properties to the data model for Logical Partition objects. - Added the absolute-general-purpose-capping, absolute-aap-capping, absolute-ifl-capping, absolute-ziip-capping and absolute-icf-capping properties to the data model for Image Activation Profile elements of CPC objects. - Added the partition-identifier property to the data model for Logical Partition objects. The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.1 or later, but primarily apply only to CPCs with SE version 2.12.1 or later:	H49574.020	H49564.021

Table 4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1) (continued)

Description	HMC MCL	SE MCL
- Added support for management of processor performance for virtual servers on x-hyp virtualization hosts, comprising the following API extensions: - Added the cpu-perf-mgmt-enabled-x-hyp property to the data model for Ensemble objects. - Added the cpu-shares-supported property to the data model for Virtualization Host objects. - Extended the existing cpu-perf-mgmt-enabled, initial-shares, minimum-shares and maximum-shares properties of Virtual Server objects to now also be applicable to virtual servers of type x-hyp. - Added the workload-processor-mgmt-status, workload-processor-mgmt-status-reason, initial-shares, shares, min-shares, and max-shares fields to the response for the Generate Hypervisor Report operation of Ensemble objects. when issued for x-hyp virtualization hosts. These fields were previously provided for PowerVM and/or z/VM® virtualization hosts but not for x-hyp virtualization hosts - Added support for ensemble availability management, comprising the following API extensions: - Added the workload-element-groups, active-avail-policy, default-avail-policy, custom-avail-policies, avail-status, and compliant-avail-status properties and related nested objects to the data model for Workload Resource Group objects. - Added the Availability Policy element of a Workload Resource Group and corresponding operations for elements of this class, including List, Create, Delete, Get Properties, Update Properties and Activate operations for Availability Policy elements of Workload Resource Group objects. - Added the Workload Element Group object and corresponding operations on objects of this class, including List, Create, Delete, Get Properties and Update Properties operations for Workload Element Group objects. - Added Add To and Remove From operations for managing the inclusion of Workload Element Groups within Workload Resource Group objects. - Added List, Add To and Remove From operations for managing the inclusion of Virtual Servers within Workload Element Group objects. - Added reporting operations for availability management, including the Generate Workload Resource Groups Report (Ensemble Availability Management), Generate Workload Resource Group Availability Status Report, Generate Virtual Server Report (Ensemble Availability Management), and Generate Availability Status Report operations. - Added an enumeration value of "workload-element-group" as a possible value of the inclusion-type field in the response for the List Virtual Servers of a Workload Resource Group operation, to specify the additional way in which virtual servers can become members of a Workload Resource Group. - Added the avail-status, acceptable-avail-status, avail-policies and workload-element-group properties and related nested objects to the data model for Virtual Server objects. - Added the heat-load, heat-load-forced-air and heat-load-water metrics to the zCPC Environmentals and Power metric group. - Added RoCE Adapters and Ensemble Power Metric groups.	H49574.020	H49564.021

Table 5. Summary of updates for API version 1.5 (HMC/SE Version 2.12.1)

Description	HMC MCL	SE MCL
Added the ability to specify per-virtual-server shutdown timeouts and to perform deactivate actions that either use or override the shutdown timeout specified in the virtual server or its hosting virtualization host configuration. This new capability applies to CPCs with SE version 2.12.1 or later that have the specified MCL installed. This new capability comprises the following detailed API extensions: • Increased API version number from 1.4 to 1.5. • Added the enumeration value "virtual-server-shutdown-timeout-override-support" as a possible feature identifier included in the feature-list property of the Virtualization Host object to indicate the availability of this new capability to virtual servers hosted on a Virtualization Host instance. • Added the shutdown-timeout-source and shutdown-timeout properties to the data model of the Virtual Server object to allow a customized default shutdown timeout to be configured for a particular virtual server. • Added the shutdown-timeout field as an optional request body field for the Deactivate Virtual Server operation to allow an individual deactivation action to be performed using a shutdown timeout that is different than the timeout configured as a default for the virtual server or its hosting virtualization host.	H49574.075	H49564.070

Table 6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 1.5 to 1.6. - Added API support for managing HMC user and role definitions, comprising the following API extensions: - Added the User object representing a defined HMC user, and associated List, Create, Delete, Get Properties, Update Properties, Add User Role, and Remove User Role operations for User objects. - Added the User Role object representing a security role for HMC users, and associated List, Create, Delete, Get Properties, Update Properties, Add Permission and Remove Permission operations for User Role objects. - Added the Task object representing the permission to invoke an HMC UI task or request an associated API operation and associated List and Get Properties operations for Task objects. - Added the User Pattern object representing a pattern string used to match user IDs during logon and associated List, Create, Delete, Get Properties and Update Properties operations for User Pattern objects. - Added the Reorder User Patterns operation for Console objects. - Added the Password Rule object representing a password format and expiration policy specification and associated List, Create, Delete, Get Properties and Update Properties operations for Password Rule objects. - Added the LDAP Server Definition object representing the configuration of an LDAP server used for HMC authentication and associated List, Create, Delete, Get Properties and Update Properties operations for LDAP Server Definition objects. - Added the replication-override-possible properties for Group objects. - Added the enumeration values "user" and "user-role" as possible object class values for the Inventory Service. - Added API support for determining the characteristics of the virtual server network adapter used for GPMP purposes, comprising the following API extensions: - Added the gpmp-network-adapter property to the data model for Virtual Server objects. - Added the adapter-type property to the network-adapter-power and network-adapter-x-hyp nested object for Virtual Server objects. - Added the gpmp-available-version property to the data model for Virtualization Host objects. - Added the management-enablement-level property to the data model for CPC objects. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.0 but primarily apply only to managed system with SE version 2.13.0 or later:	N98841	CPC: N98775 zBX: N98822

Table 6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0) (continued)

Description	HMC MCL	SE MCL
- Added API support for zBX Model 004 ensemble nodes, comprising the following API extensions: - Added the type property to the data model for a zBX object to indicate whether the zBX is a CPC-attached (zBX Model 002/003) or ensemble node (zBX Model 004) zBX. As related extensions, also added the type property to the response for the <code>List zBXs of Ensemble</code> operation and also added it as an optional filtering query parameter for that operation. Note that when a zBX object represents a zBX node, the value of its parent property contains the URI of the Ensemble of which it is a member rather than the URI of the CPC to which it is attached. - Added an optional new input format for the <code>Add Node to Ensemble</code> operation to allow a zBX Model 004 to be specified as the node to be added to the ensemble. - Added the enumeration value "zbx" as a possible value for the type property of a Node element of an Ensemble object. - Added the max-nodes, max-cpc-nodes and max-zbx-nodes properties to the data model for an Ensemble object. - Added many additional properties to the data model for a zBX object when that object represents a zBX node. - Added the <code>Get EC/MCL Description</code>, <code>Activate</code>, <code>Deactivate</code>, <code>Set Power Save</code>, <code>Set Power Capping</code>, <code>List Virtualization Hosts</code> and <code>List Virtual Servers</code> operations for zBX node objects. - Added the <code>Activate</code> and <code>Deactivate</code> operations for BladeCenter objects contained within a zBX node. - Added the <code>List Virtualization Hosts</code> and <code>List Virtual Servers</code> operations for Ensemble nodes in general. - Added the enumeration value "node" as a possible value of the availability status nested object of a Virtual Server object. - Added the zBX (Node) Overview metric group as a metric group that can be requested using the Metric Service. - Added API support for manipulation of hardware messages, comprising the following API extensions: - Added the hardware messages container property and hardware message nested objects to the data model for zBX node and CPC objects and to the data model for the HMC Console object. - Added the <code>List Hardware Messages</code>, <code>Get Hardware Message Properties</code> and <code>Delete Hardware Message</code> operations for zBX node and CPC objects and the HMC Console object. - Added API support for the retrieval of audit and security log information, comprising the following API extensions: - Added a notification topic and Log Entry notification messages to provide for asynchronous push-type delivery of new log entries for the HMC Console object to interested API clients. - Added the <code>Get Notification Topics</code> operation for Session objects to allow API clients a more general way of retrieving the names of notification topics for an API session. - Added the <code>Get Audit Log</code> and <code>Get Security Log</code> operations for zBX node and CPC objects and the HMC Console object. - Added the smt-usage, thread-0-usage and thread-1-usage metrics in the zCPC Processors metric group. - Added the enumeration value "virtio-scsi" as a possible value for the emulation mode property of a virtual disk element of a Virtual Server object.	N98841	CPC: N98775 zBX: N98822

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Service API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: • Increased API version number from 1.6 to 1.7. • Added the following common request validation reason codes: – HTTP status 400 with reason code 18 – HTTP status 404 with reason codes 5 and 6 – HTTP status 409 with reason codes 4, 5, 6, and 9 • Added the following data model property qualifiers: (e) for effective properties, and (p) for pseudo properties. • Added the effective-properties-apply base managed object property for objects which contain an effective property (a property marked with the (e) qualifier in the object's data model). • Added API support for sending commands to and receiving messages from the operating system (OS) executing in Logical Partitions and Partitions, comprising the following API extensions: – Added the operating system notification topic, on which OS messages are received. – Added the operating system message notification message which contains the text from the OS. – Added information about any operating system notification topics associated with the API session to the Get Notification Topics response. – Added the Open OS Message Channel and Send OS Command operations for Logical Partition objects. – Added the Open OS Message Channel and Send OS Command operations for Partition objects. • Added minimal API support for Managed Virtual Machine objects, such that they can be discovered, added to User Roles and removed from User Roles, comprising the following API extensions: – Added the List Managed Virtual Machines of a Logical Partition operation for Logical Partition objects. The URIs returned can be used to add/remove Managed Virtual Machine objects to/from User Roles. – No further API support is provided. That is, there are no List, Create, Delete, Get Properties or Update Properties operations for Managed Virtual Machine objects, and they are not included in the Inventory Service.	P08462.053	P00339.068

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)

Description	HMC MCL	SE MCL
- Added minimal support for unmanaged CPC objects, such that they can be listed, added to User Roles and removed from User Roles. Unmanaged CPCs are those which have been discovered by the HMC but are not configured to be managed by the HMC. This support comprises the following API extensions: - Added the <code>List Unmanaged CPCs</code> operation for the Console object. The URIs returned can be used to add/remove unmanaged CPC objects to/from User Roles. - No further API support is provided. That is, there are no Create, Delete, Get Properties, or Update Properties operations for unmanaged CPC objects, and they are not included in the Inventory Service. - Added HTTP status 409 with reason code 329 on operations that target a CPC object but do not support an undefined CPC object as their target. - Added minimal support for unmanaged zBX node objects, such that they can be listed, added to User Roles, and removed from User Roles. Unmanaged zBX nodes are those which have been discovered by the HMC but are not configured to be managed by the HMC. This support comprises the following API extensions: - Added the <code>List Unmanaged zBX Nodes</code> operation for the Console object. The URIs returned can be used to add/remove unmanaged zBX node objects to/from User Roles. - No further API support is provided. That is, there are no Create, Delete, Get Properties, or Update Properties operations for unmanaged zBX node objects, and they are not included in the Inventory Service. - Added HTTP status 409 with reason code 244 on operations that target a zBX node object but do not support an undefined zBX node object as their target. - Added API support for absolute capping of Logical Partition processor usage, comprising the following API extensions: - Added the effective-capacity, absolute-icf-capping, effective-absolute-icf-capping, absolute-ifl-capping, effective-absolute-ifl-capping, absolute-general-purpose-capping, effective-absolute-general-purpose-capping, absolute-ziip-capping, effective-absolute-ziip-capping and effective-properties-apply properties to the data model for Group Profile objects. - Added HTTP status 409 with reason code 9 on <code>Update Group Profile Properties</code>.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)

Description	HMC MCL	SE MCL
- Added the storage-total-installed, storage-hardware-system-area, storage-customer, storage-customer-central, storage-customer-expanded and storage-customer-available properties to the data model for CPC objects. - Changed the data model qualifier from (w) to (wo) for the zaware-master-pw property of Logical Partition and Image Activation Profile objects. This is not a behavioral change but rather just a data model notation change to document the behavior using current documentation conventions. - Added the storage-central-allocation and storage-expanded-allocation properties to the data model for Logical Partition objects. - Added a limit of 500 Workload Resource Groups per ensemble. An attempt to exceed that limit on a Create Workload Resource Group operation results in HTTP status 409 with reason code 66. - To improve security, the group-profile-capacity property of the Logical Partition object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for Logical Partition objects and it is now a read-only property. The way for an authorized API client to change a Logical Partition's group-profile-capacity is to change the capacity and/or effective-capacity property of the Group Profile with which the Logical Partition is associated. - Added the effective-capacity and effective-properties-apply properties to the data model for Group Profile objects. - Added HTTP status 409 with reason code 9 to Update Group Profile Properties. - New HTTP status 400 with reason code 330 on the Update User Properties operation for User Role objects when the request is to disable the API user's own user ID. - New HTTP status 400 with reason code 300 on the Update Image Activation Profile Properties operation for Image Activation Profile objects. - Added new cp-all-processor-usage, ifl-all-processor-usage, icf-all-processor-usage, and iip-all-processor-usage metrics to the CPC overview metric group. - Added new exhaust-temperature-celsius metric to the zCPC environmentals and power metric group.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)

Description	HMC MCL	SE MCL
- Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. - Added several enumeration values for the name property of Task objects, due to new tasks added to this HMC version. - Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the name property of the corresponding Task objects are part of the programming API and are not changed. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 but primarily apply only to managed systems with SE version 2.13.1 or later: - Added API support for z Appliance Container Infrastructure (zACI) Logical Partitions, comprising the following API extensions, but only for Logical Partitions of the new "zaci" type: - Added "zaci" as a possible enumeration value for the activation-mode property of Logical Partition objects. - Added the zaci-host-name, zaci-master-userid, zaci-master-pw, zaci-network-info, zaci-gateway-info and zaci-dns-info properties to the data models for Logical Partition and Image Activation Profile objects. - Added "zaci" as a possible enumeration value for the operating-mode property of Image Activation Profile objects. - Added API support for energy optimization advice, comprising the following API extensions: - Added the Get Energy Optimization Advice Summary and Get Energy Optimization Advice Details operations for CPC objects. - Added the last-energy-advice-time property to the data model for CPC objects. - Added API support for absolute capping of logical partition processor usage through the new absolute-icf-capping, effective-absolute-icf-capping, absolute-ifl-capping, effective-absolute-ifl-capping, absolute-general-purpose-capping, effective-absolute-general-purpose-capping, absolute-ziip-capping, and effective-absolute-ziip-capping properties of the data model for Group Profile objects.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)

Description	HMC MCL	SE MCL
• Added API support for Dynamic Partition Manager (DPM), comprising the following API extensions (These extensions are only available if the Feature on Demand record for DPM has been installed on the Support Element): – Added the Partition object, representing a partition of a CPC, into which an operating system can be loaded and then started, and corresponding operations on object of this class, including List, Create, Delete, Get Properties, Update Properties, Start, Stop, Dump, PSW Restart, Mount ISO and Unmount ISO operations for Partition objects. – Added the Virtual Function element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties and Update Properties operations for Virtual Function elements of Partition objects. – Added the NIC element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties and Update Properties operations for NIC elements of Partition objects. – Added the HBA element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties, Update Properties and Reassign Storage Adapter Port for HBA elements of Partition objects. – Added the Adapter object representing a network or storage adapter, and corresponding operations on objects of this class, including List, Get Properties, Update Properties, Change Crypto Type, Create Hipersocket, Delete Hipersocket and Get Partitions Assigned to Adapter operations for Adapter objects. – Added the Network Port element of an Adapter and corresponding operations for elements of this class, including Get Properties and Update Properties for Network Port elements of Adapter objects. – Added the Storage Port element of an Adapter and corresponding operations for elements of this class, including Get Properties and Update Properties for Storage Port elements of Adapter objects. – Added the Virtual Switch object representing a CPC's network adapter and port, and corresponding operations on objects of this class, including List, Get Properties, Update Properties and Get Connected VNICS operations for Virtual Switch objects. – Added the Capacity Group element of a CPC and corresponding operations for elements of this class, including List, Create, Delete, Get Properties, Update Properties, Add Partition and Remove Partition for Capacity Group elements of CPC objects.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)		
Description	HMC MCL	SE MCL
(cont'd) - Added the DPM system overview, Partitions, Adapters metric groups. - Added the dpm-enabled, is-cpacf-enabled and maximum-hipersockets properties to the data model for CPC objects. - Added the ability to update the description property of CPC objects when the CPC is enabled for DPM. - Added "dpm" as a possible enumeration value for the iml-mode property of CPC objects. Note that some preexisting CPC properties are not applicable when the CPC is enabled for DPM; such properties are not returned on Get CPC Properties nor are they valid for the Update CPC Properties operation. - Added HTTP status 409 reason code 4 for the following operations for CPC objects, because these operations are not valid when the CPC is enabled for DPM: Activate CPC, Deactivate CPC, Import Profiles, and Export Profiles. - The following related objects are not provided for a CPC enabled for DPM: Logical Partition, Reset Activation Profile, Image Activation Profile, Load Activation Profile and Group Profile. - Added the Start CPC, Stop CPC and Export WWPN List operations for CPC objects. - Added API support for a partition auto-start list of a CPC, comprising the following API extensions: - Added the auto-start-list property to the data model for CPC objects. - Added the Set Auto-Start List operation for CPC objects. - Added the enumeration values "partition" and "adapter" as possible object class values for the Inventory Service. - Added the enumeration value "dpm-resources" as a possible object category value for the Inventory Service. - Added the zaci-boot-selection property to the data model for Image Activation Profile objects.		

Table 8. Summary of updates for API version 2.1 (HMC/SE Version 2.13.1)		
Description	HMC MCL	SE MCL
Increased API version number from 1.7 to 2.1. Note that the change in the major portion of the version number indicates that this version is not compatible with the previous version. The only incompatible changes in this version are due to renaming z Appliance Container Infrastructure to IBM Secure Service Container. Specifically: - The renaming of the zaci-host-name, zaci-master-userid, zaci-master-pw, zaci-network-info, and zaci-gateway-info Logical Partition object properties to ssc-host-name, ssc-master-userid, ssc-master-pw, ssc-network-info, and ssc-gateway-info, respectively. - The change of the "zaci" enum value for the Logical Partition object's activation-mode property to "ssc". - The renaming of the zaci-host-name, zaci-master-userid, zaci-master-pw, zaci-network-info, zaci-gateway-info and zaci-boot-selection Image Activation Profile object properties to ssc-host-name, ssc-master-userid, ssc-master-pw, ssc-network-info, ssc-gateway-info, and ssc-boot-selection, respectively. - The change of the "zaci" enum value for the Image Activation Profile object's operating-mode property to "ssc".	P08462.250	P00339.291
Removed the restriction that the is-cpacf-enabled property of the CPC object is only present when dpm-enabled is true. The is-cpacf-enabled property is now always present. Also corrected the qualifier column for is-cpacf-enabled to indicate that it does not provide property change notifications.	P08462.232	

Table 9. Summary of updates for API version 2.2 (HMC/SE Version 2.13.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.1 to 2.2. - Added the String/Hostname data type. - Provided a more granular value for the sysplex-name property of the Logical Partition to differentiate when the z/OS® logical partition is not in a sysplex. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 but primarily apply only to managed systems with SE version 2.13.1 or later: - Added API support for Secure Service Container (SSC) partitions of DPM-enabled CPCs, comprising the following API extensions: - Added the type, ssc-host-name, ssc-boot-selection, ssc-ipv4-gateway, ssc-dns-servers, ssc-master-userid and ssc-master-pw properties to the data model for Partition objects. - Added the type property to the response for the List Partitions of a CPC operation and also added it as an optional filtering query parameter for that operation. - Added the ssc-management-nic, ssc-ip-address-type, ssc-ip-address, ssc-mask-prefix, vlan-id and mac-address properties to the data model for NIC objects. - Added HTTP status 400 with reason codes 15, 18 and 20 to the Create Partition operation. - Added HTTP status 400 with reason codes 15 and 18 to the Update Partition Properties operation. - Added HTTP status 409 with reason code 120 to the job status for the Start Partition operation. - Added HTTP status 400 with reason code 15 and HTTP status 409 with reason code 8 to the Create NIC operation. - Added HTTP status 400 with reason code 15 to the Update NIC Properties operation.	P08462.261	P00339.304

Table 10. Summary of updates for API version 2.3 (HMC/SE Version 2.13.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.2 to 2.3. - Removed the write qualifier from the crypto-number property of the Adapter object. This property was never in fact writable. - Corrected the default behavior of the Import DPM Configuration operation to assign new object IDs, element IDs and WWPNS to import object so as to avoid the possibility of introducing objects with duplicate IDs and WWPNS. Added the preserve-uris and preserve-wwpns fields to the request body for the Import DPM Configuration operation to override that behavior when appropriate.	P08462.306	

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.2 to 2.20. - Added the following common request validation reason codes: - HTTP status 400 with reason codes 19 and 20 - HTTP status 409 with reason code 11 - Added API support for the HMC mobile app, comprising the following API extensions: - Added the mobile-app-preferences property to the Console object data model. - Added the Get Mobile App Preferences and Set Mobile App Preferences operations for the Console object to manage the console-wide mobile app settings. - Added the Get CPC Notification Preferences for Device and Update CPC Notification Preferences for Device operations for the Console object to manage the mobile app notification preferences for a CPC to a mobile device. - Added the client-tag field to the request body for the Logon operation. - Added API support to manage and interact with the console's multi-factor authentication support, comprising the following API extensions: - Added the multi-factor-authentication-required and force-shared-secret-key-change properties to the User object data model. - Added the multi-factor-authentication-required field to the request body for the Create User operation. - Added the multi-factor-authentication-code field to the request body for the Logon operation. - Added the shared-secret-key and session-credential fields to the response for the Logon operation. - New HTTP status 201 on the Logon operation. - New HTTP status 400 with reason codes 46, 47, 48 and 49 on the Logon operation. - Added the Establish Shared Secret Key operation. - Added the hmc-time field to the response for the Query API Version operation. - Expanded the authentication options when connecting to the API message broker; the session ID and session-specific credential are required for multi-factor authentication users. - Relaxed the definition of a pseudo property to no longer require a pseudo property to also be a container property. - Added the (c) data model qualifier to the connected-vnic-uris pseudo property of Virtual Switch objects to denote that it is a container property. Since psuedo properties were also container properties in previous versions of the API, this is a documentation-only change and not a behavioral change.	P42675.054	P42601.066

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0) (continued)

Description	HMC MCL	SE MCL
- Added API enhancements that allow quicker responses when fetching the properties of certain object types, comprising the following API extensions: - Added a properties query parameter to the following operations to allow an API client to avoid the overhead involved with fetching object properties in which it has no interest: Get CPC Properties, Get Logical Partition Properties. - Added a cached-acceptable query parameter to the following operations to allow an API client to indicate that cached (but potentially out-of-date) property values are acceptable: Get CPC Properties, Get Logical Partition Properties, Get Reset Activation Profile Properties, Get Image Activation Profile Properties, and Get Load Activation Profile Properties. - Added the Delete Partition Asynchronously and Update Partition Properties Asynchronously operations for Partition objects. - Added the fid property to the data model for Virtual Function elements of Partition objects. - Made the mac-address property of NIC elements of Partition objects writable in certain cases. - Added the vlan-type property to the data model for NIC elements of Partition objects. - Relaxed the authorization requirements for access to Load Activation Profile objects via the List Load Activation Profiles and Get Load Activation Profile Properties operations to allow access without requiring CPC object-access permission under certain circumstances. This is consistent with existing HMC UI behavior. - Relaxed the authorization requirements for access to Image Activation Profile objects via the List Image Activation Profiles and Get Image Activation Profile Properties operations to allow access without requiring CPC object-access permission under certain circumstances. This is consistent with existing HMC UI behavior. - Added synchronous interfaces for retrieving operating system messages, comprising the following API extensions: - Added the List OS Messages of a Partition operation for Partition objects. - Added the List OS Messages of a Logical Partition operation for Logical Partition objects. - Added the sequence-number field to the os-message-info object in operating system message notifications. - Added support for more efficient access to certain Partition and Logical Partition information, comprising the following API extensions: - Added the List Permitted Partitions operation. - Added the List Permitted Logical Partitions operation. - Added the has-unacceptable-status, dpm-enabled and se-version properties to the response for the List CPC Objects operation. - Added the classification-text field to the response for the Query API Version operation. - Added the last-used-iocds property to the CPC object data model. - Added the last-used-load-address and last-used-load-parameter properties to the Logical Partition object data model. - Relaxed the requirements on the boot-iso-image-name property of Partition objects and documented the requirements in the Partition object data model. This affects the Mount ISO Image operation for Partition objects. - Corrected the documentation for the Create Partition operation; several fields were missing from the request body definition. - Added the following as a possible enumeration value for the detected-card-type property of an Adapter object: "roce-express-2".	P42675.054	P42601.066

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0) (continued)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 but primarily apply only to managed systems with SE version 2.14.0 or later: - Added support for IBM Virtual Flash Memory, comprising the following API extensions: - Added the storage-vfm-increment-size and storage-vfm-total properties to the CPC object data model. - Added the initial-vfm-storage, maximum-vfm-storage and current-vfm-storage properties to the Logical Partition object data model. - Added the initial-vfm-storage and maximum-vfm-storage properties to the Image Activation Profile object data model. - Added the port property to the ssc-network nested object definition used by the Logical Partition and Image Activation Profile data models. - Extended the set of characters permitted in a load parameter, comprising the following API extensions: - Allow the following three additional characters (@, \$, #) in the load-parameter field of the request body for the Load Logical Partition, SCSI Load, and SCSI Dump operations. - Allow the following three additional characters (@, \$, #) in the ipl-parameter property in the Image Activation Profile and Load Activation Profile data models. - Removed support for controlling whether a Logical Partition entering a wait state causes termination of a time slice: - The does-wait-state-end-time-slice property of the CPC object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for CPC objects and it is now a read-only property whose value is always false. - The end-timeslice-on-wait property of the Reset Activation Profile object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for Reset Activation Profile objects and it is now a read-only property whose value is always false. - New HTTP status 400 with reason code 19 on the Update CPC Properties and Update Reset Activation Profile Properties operations. - Replaced the "esa390" and "esa390-tpf" logical partition activation modes with the new "general" activation mode, comprising the following API changes: - Added the enumeration value "general" as a possible value for the activation-mode property of a Logical Partition object. - Added the enumeration value "general" as a possible value for the operating-mode property of an Image Activation Profile object. - Due to the removal of support for dynamic changes to the SSC configuration, the SSC-related properties of the Logical Partition object are no longer directly writable through the API. The (w) qualifier has been removed from those properties in the data model for Logical Partition objects and they are now read-only properties. The affected properties are ssc-host-name, ssc-master-userid, ssc-master-pw, ssc-network-info, ssc-gateway-info and ssc-dns-info. - Added the ssc-gateway-ipv6-info property to the Image Activation Profile object data model. - Added the last-energy-advice-time property to the CPC object data model (applicable to SE version 2.13.1 or later).	P42675.054	P42601.066

Table 12. Summary of updates for API version 2.21 (HMC/SE Version 2.14.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.20 to 2.21. - Added the following as possible enumeration values for the detected-card-type property of the Adapter objects: "osa-express-6s-1gb", "osa-express-6s-10gb", "osa-express-6s-1000base-t", "crypto-express-6s", "ficon-express-16s", and "ficon-express-16s-plus". - Added more granular controls over which OS message notifications are presented to a mobile device, comprising the following API extensions: - Added the new-os-message-filtered field to the response for the Get CPC Notification Preferences for Device operation. - Added the new-os-message-filtered field to the request body for the Update CPC Notification Preferences for Device operation.	P42675.062	

Table 13. Summary of updates for API version 2.22 (HMC/SE Version 2.14.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.21 to 2.22. - Added API support for accessing the ASCII console of partitions of DPM-enabled CPCs via WebSockets, comprising the following API extension: - Added the Get ASCII Console WebSocket URI operation for Partition objects.	P42675.116	

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.22 to 2.23. - Added the following common request validation reason codes: - HTTP status 409 with reason codes 12 and 13 - Added support for object-specific firmware features, comprising the following API extensions: - Added the available-features-list property to the CPC object data model. - Added the available-features-list property to the Partition object data model. - Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. - Added enumeration values for the name property of User Role objects, due to new system-defined user roles added to this HMC version. - Added enumeration values for the name property of Task objects, due to new tasks added to this HMC version. - Removed enumeration values for the name property of Task objects for tasks that are not provided in this HMC version. - Changed the descriptive name of some classes of managed objects. Those descriptive names are not part of the programming API. The enumeration values for the managed object classes are part of the programming API and are not changed. - Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the name property of the corresponding Task objects are part of the programming API and are not changed. The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 but primarily apply only to managed systems with SE version 2.14.0 or later: - Added the Import DPM Configuration operation for CPC objects to aid in migrating a DPM configuration from a z13® system to a z14 system.	P42675.232	P42601.286

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0) (continued)

Description	HMC MCL	SE MCL
- Added support for managing a FICON configuration for DPM-enabled CPCs and their Partitions, comprising the following API extensions (The majority of these extensions are only available if the dpm-storage-management feature is enabled on the CPC or Partition of interest): - Added the Storage Site object, representing a single storage site in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Update Properties for Storage Site objects. - Added the Storage Fabric object, representing a single storage fabric in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Update Properties for Storage Fabric objects. - Added the Storage Switch object, representing a single storage switch in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Move Storage Switch to Storage Site and Move Storage Switch to Storage Fabric for Storage Switch objects. - Added the Storage Subsystem object, representing a single storage subsystem in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Move Storage Subsystem to Storage Site, Add Connection Endpoint and Remove Connection Endpoint for Storage Subsystem objects. - Added the Storage Control Unit object, representing a single storage control unit in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Add Volume Range, Remove Volume Range for Storage Control Unit objects. - Added the Storage Path element of a Storage Control Unit and corresponding operations on elements of this class, including Create, Delete, Get Properties and Update Properties for Storage Path elements of Storage Control Unit objects. - Added the Storage Group object, representing a single storage group in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Modify Storage Group Properties, Add Candidate Adapter Ports to an FCP Storage Group, Remove Candidate Adapter Ports from an FCP Storage Group, Get Partitions for a Storage Group and Validate LUN Path for Storage Group objects. - Added the Storage Volume element of a Storage Group object and corresponding operations on elements of this class, including List, Get Properties, Fulfill FICON Storage Volume, and Fulfill FCP Storage Volume for Storage Volume elements of Storage Group objects. - Added the Virtual Storage Resource element of a Storage Group object and corresponding operations on elements of this class, including List, Get Properties, and Update Properties for Virtual Storage Resource elements of Storage Group objects. - Added the Start Dump Program, Attach Storage Group to Partition, and Detach Storage Group from Partition operations for Partition objects. - Added the enumeration value "storage-volume" as a possible value for the boot-device property in the Partition object data model. - Added the boot-storage-volume and boot-load-parameters properties to the Partition object data model.		

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0) (continued)		
Description	HMC MCL	SE MCL
(cont'd) – New HTTP status 409 with reason codes 119, 120, 121 and 122 on the Update Partition Properties operation. – New HTTP status 409 with reason code 122 on the Start Partition operation. – New HTTP status 409 with reason code 12 on the following operations when the dpm-storage-management feature is enabled on the targeted CPC: Export WWPN List, Dump Partition and Create HBA. – Added the enumeration values "fc" and "not-configured" as possible values for the type field in the Adapter object data model. – Added the connection-endpoint-uri and connection-endpoint-class properties to the data model for Storage Port elements of Adapter objects. – Added the Change Adapter Type operation for Partition objects. – Added the enumeration values "storage-site", "storage-fabric", "storage-switch", "storage-subsystem", "storage-control-unit", and "storage-group" as possible object class values for the Inventory Service. • Added support for Container Based Processors, comprising the following API extensions: – Added new cpb-shared-processor-usage, cpb-dedicated-processor-usage, and cpb-all-processor-usage metrics to the CPC overview metric group. – Added new cpb-processor-usage metric to the Logical partitions metric group. – Added "cbp" as a possible value for the processor-type metric in the zCPC processors metric group. – Added the processor-count-cbp and processor-count-pending-cbp properties to the CPC object data model. – Added the initial-cbp-processing-weight, initial-cbp-processing-weight-capped, minimum-cbp-processing-weight, maximum-cbp-processing-weight, current-cbp-processing-weight, current-cbp-processing-weight-capped, and absolute-cbp-capping properties to the Logical Partition object data model. – Added the initial-cbp-processing-weight, initial-cbp-processing-weight-capped, minimum-cbp-processing-weight, maximum-cbp-processing-weight, absolute-cbp-capping, number-dedicated-cbp-processors, number-reserved-dedicated-cbp-processors, number-shared-cbp-processors, and number-reserved-shared-cbp-processors properties to the Image Activation Profile object data model. – Added the absolute-cbp-capping and effective-absolute-cbp-capping properties to the Group Profile object data model. – Added the enumeration value "cbp" as a possible value for the type field in the Capacity Record object data model. – Added the enumeration value "cbp" as a possible value for the processor-type field in the request body for the Add Temporary Capacity and Remove Temporary Capacity operations.		

Table 15. Summary of updates for API version 2.24 (HMC/SE Version 2.14.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.23 to 2.24. • Added the vendor field to the response for the Query API Version operation. • Added the maximum-partitions property to the CPC object data model.	P42675.233	

Table 16. Summary of updates for API version 2.25 (HMC/SE Version 2.14.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.24 to 2.25. - Removed the write qualifier from the crypto-number property of the Adapter object. This property was never in fact writable. - Added HTTP status 409 with reason code 496 on the Change Adapter Type operation. - Corrected HTTP status 404 with reason code 2 to reason code 1 on the Undefine Storage Control Unit operation. - Added HTTP status 409 with reason code 497 to the Create Storage Path and Update Storage Path Properties operations. - Added enumeration value "unknown" as a possible value for the status field in the World Wide Port Name Information nested object. - Added HTTP status 409 with reason code 493 to the Create Storage Group, Delete Storage Group, and Modify Storage Group operations. - Added object-access permission on the associated partition for Update Virtual Storage Resource Properties operations that update the adapter-port-uri or device-number properties. - Added HTTP status 403 with reason code 420 to the Update Virtual Storage Resource Properties operation. - Added HTTP status 409 with reason code 498 to the Update Virtual Storage Resource Properties operation.	P42675.283	

Table 17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: - Increased API version number to 2.35. - Added support to delete operating system messages, comprising the following API extensions: - Added the Delete Partition OS Message operation for Partition objects. - Added the Delete Logical Partition OS Message operation for Logical Partition objects. - Added the last-used-world-wide port-name, last-used-logical-unit-number, last-used-disk-partition-id, last-used-operating-system-specific-load-parameters and last-used-boot-record-logical-block-address properties to the Logical Partition object data model. - Added the se-version field to the response for the List Permitted Partitions and List Permitted Logical Partitions operations. - Added more detailed information about hardware messages, comprising the following API extensions: - Added the details property to the data model for Hardware Message elements of the Console object. - Added the details property to the data model for Hardware Message elements of the CPC object. - Added the details property to the data model for Hardware Message elements of the zBX (Node) object. - Added request aggregation services which allow what would otherwise be multiple API requests to be submitted as a single request, with their multiple results likewise returned in a single response, comprising the following API extension: - Added the Submit Requests operation. - Added the enumeration value "storage-template" as a possible object class value for the Inventory Service. - New HTTP status 400 with reason code 453 on the Modify Storage Group operation. - New HTTP status 400 with reason code 452 on the Update Virtual Storage Resource Properties operation. - New HTTP status 400 with reason code 453 on the Create Storage Group operation. - New HTTP status 409 with reason code 478 on the Create Storage Group operation. - Due to the removal of CIM actions from the HMC, the authorization requirements for some operations have changed. The affected operations are Import Profiles, Export Profiles, Add Temporary Capacity and Remove Temporary Capacity operations for CPC objects. - Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. - Added several enumeration values for the name property of Task objects, due to new tasks added to this HMC version. - Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the name property of the corresponding Task objects are part of the programming API and are not changed. - Added the following as possible enumeration values for the detected-card-type property of an Adapter object: "osa-express-7s-25gb", "roce-express-2-25gb", and "fcp-express-32s". - Added Change Object Options task permission for Update Logical Partition Properties operations that update the next-activation-profile-name property.	P41499.052	P41414.060

Table 17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1) (continued)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: - Added the <code>Accept Mismatched FCP Storage Volumes</code> and <code>Reject Mismatched FCP Storage Volumes</code> operations for Storage Group objects. - Added the <code>Zeroize Crypto Domain</code> operation for Partition objects. - Added support for not clearing memory before loading a Logical Partition from a SCSI device, comprising the following API extensions: - Added the clear-indicator field to the request body for the <code>SCSI Load</code> operation for Logical Partition objects. - Allow the clear-indicator property of the Load Activation Profile to be set to false for SCSI loads. - Added support for storage group templates for DPM-enabled CPCs to facilitate the creation of storage groups with certain initial property values, comprising the following API extensions: - Added the Storage Template object, representing a single storage template, and corresponding operations on objects of this class, including <code>List</code>, <code>Create</code>, <code>Delete</code>, <code>Get Properties</code> and <code>Modify Properties</code> for Storage Template objects. - Added the Storage Template Volume element of a Storage Template object and corresponding operations on elements of this class, including <code>List</code> and <code>Get Properties</code> for Storage Template Volume elements of Storage Template objects. - Added the template-uri field in the request body for the <code>Create Storage Group</code> operation. - Changed the adapter-port-uri property of the Virtual Storage Resource element object from a read-only property to a writable property.		

Table 18. Summary of updates for API version 2.36 (HMC/SE Version 2.14.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: - Increased API version number from 2.35 to 2.36. - Removed the write qualifier from the crypto-number property of the Adapter object. This property was never in fact writable. - Added HTTP status 409 with reason code 496 to the <code>Change Adapter Type</code> operation. - Corrected HTTP status 404 with reason code 2 to reason code 1 on the <code>Undefine Storage Control Unit</code> operation. - Added HTTP status 409 with reason code 497 to the <code>Create Storage Path</code> and <code>Update Storage Path</code> operations. - Added enumeration value "unknown" as a possible value for the status field in the World Wide Port Name Information nested object. - Added HTTP status 409 with reason code 493 to the <code>Create Storage Group</code>, <code>Delete Storage Group</code>, and <code>Modify Storage Group</code> operations. - Added object access permission on the associated partition for <code>Update Virtual Storage Resource Properties</code> operations that update the adapter-port-uri or device-number properties. - Added HTTP status 403 with reason code 450 to the <code>Update Virtual Storage Resource Properties</code> operation. - Added HTTP status 409 with reason code 498 to the <code>Update Virtual Storage Resource Properties</code> operation. - Added HTTP status 400 with reason code 451 to the <code>Reject Mismatched FCP Storage Volumes</code> operation.	P41499.108	

Table 19. Summary of updates for API version 2.37 (HMC/SE Version 2.14.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: - Increased API version number from 2.36 to 2.37 - Added Start FCP Storage Discovery, Get Connection Report, and Get Storage Group Histories operations for Storage Group objects - Added a new field, paths to the Storage Volume element object's Data Model.	P41499.161	P41414.177

Table 20. Summary of updates for API version 2.38 (HMC/SE Version 2.14.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: - Increased API version number from 2.37 to 2.38 - Added World Wide Port Name information nested object to the Virtual Storage Resource element object. - Added "incomplete" status to the list of WWPN status values in Table 206 on page 447. - Added object access permission on the referenced storage adapter for Update Storage Path Properties operations that update the adapter-port-uri property. - Added HTTP status 404 with reason code 442 to the Create Storage Path operation. - Added HTTP status 409 with reason code 441 to the Update Storage Port Properties, Define Storage Switch, Create Storage Group, Add Candidate Adapter Ports to an FCP Storage Group, Fulfill FICON Storage Volume, Fulfill FCP Storage Volume, Update Virtual Storage Resource Properties, and Validate LUN Path operations. - Added HTTP status 409 with reason code 457 to the Add Connection Endpoint operation. - Added HTTP status 409 with reason code 499 to the Modify Storage Group Properties operation.	P41499.208	

Table 21. Summary of updates for API version 2.39 (HMC/SE Version 2.14.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: - Increased API version number from 2.38 to 2.39. - Added API support for fulfilling multiple FICON storage volumes with a single request, comprising the following API extension: - Added the Fulfill FICON Storage Volumes operation to the Storage Group object.	P41499.215	

Table 22. Summary of updates for API version 2.40 (HMC/SE Version 2.14.1)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: - Increased API version number to 2.40. - Corrected the handling of the properties query parameter of the Get CPC Properties and Get Logical Partition Properties operations to no longer return HTTP status 400 for properties the HMC supports but the target CPC or Logical Partition does not support. Such properties are simply omitted from the response.	P41499.257	

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: - Increased API version number to 3.1. Note that the change in the major portion of the version number indicates that this version is not compatible with the previous version. The vast majority of the incompatibilities are due to the removal of zManager's support for IBM z BladeCenter Extension (zBX), the blades of a zBX, alternate HMC, Ensembles, and related components. More detailed information is provided below. - Removed support for IBM z BladeCenter Extension (zBX), the blades of a zBX, alternate HMC, Ensembles, and related components. The majority of that support is described in <i>Part 4: "Ensemble and zBX management"</i> of the <i>Hardware Management Console Web Services API</i>, SC27-2637. The changes are: - Removed support for the following managed object and element object classes and types. This includes removal of all operations, inventory classes, inventory categories, and metric groups related to these object classes and types. - ensemble - node - zbx - top-of-rack-switch - rack - bladecenter - blade - virtualization-host - virtual-switch - virtual-server - storage-resource - virtualization-host-storage-resource - virtualization-host-storage-group - network adapter of a virtual server - virtual disk - performance policy - availability policy - Removed all operations described in <i>Part 4: "Ensemble and zBX management"</i> of the <i>Hardware Management Console Web Services API</i>, SC27-2637. As that is a considerable number of operations, they are not listed here. - Removed the following inventory categories and all classes contained therein from the Inventory Service: - zvm-resources - power-vm-resources - x-hyp-resources - prsm-resources - virtual-server-common - zbx-resources - ensemble-wide-resources	P46683.065	P46598.076

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)

Description	HMC MCL	SE MCL
• (cont'd.) – Removed the following metric groups from the Metrics Service: - BladeCenter temperature and power metric group - Blade power - zBX (Node) overview - Blade CPU and memory metric group - Ensemble power - Virtual server CPU and memory metrics group - Virtualization host CPU and memory metrics group - Workload service class data metrics group - Virtualization host (vSwitch) uplink metric group - Virtualization host (vSwitch) by virtual network metric group - Attached virtual server network adapters metric group - Optimizer IEDN virtual network interface metric group - Optimizer IEDN physical network adapter metric group - Top-of-rack switch ports metrics - ESM switch port metrics – Removed the following operations not previously described in the <i>"Ensemble and zBX management"</i> section: - Make Console Primary - List Ensemble CPC Objects - List Unmanaged zBX Nodes – Removed the following object properties: - Removed the paired-role, is-auto-switch-enabled, paired-hmc, and ip-swapping-available properties of the Console object - Removed is-ensemble-member and management-enablement-level properties of the CPC object – Removed several object types from those eligible for membership in pattern-matching groups. The following are no longer valid enumeration values for the types property of a Group object: - "zvm-virtual-machines" - "blade-center" - "data-power-xi50z-blades" - "ibm-smart-analytics-optimizer-blades" - "power-blade" - "system-x-blade" - "power-vm-virtual-server" - "system-x-virtual-server" - "workload" - "defined-zbx-node" - "director-timer-console" - "ibm-fiber-saver"		

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)

Description	HMC MCL	SE MCL
(cont'd.) - Removed several object types from those that can be permitted in user roles. The following are no longer valid enumeration values for the permitted-object property of a User Role object: "alternate-console" "blade-center" "datapower-device" "default-workload" "eckd-storage-resource" "ensemble" "fcp-storage-resource" "p-blade" "p-virtual-server" "undefined-zbx-node" "virtual-network" "workload-element-group" "workload-resource-group" "x-blade" "x-virtual-server" "zbx-node" "zvm-fcp-storage-resource" "zvm-virtual-machine" - Removed several system-defined User Roles. User Roles with the following enumeration values for the name property have been removed: "hmc-bladecenter-objects" "hmc-dpxi50z-blade-objects" "hmc-ensemble-administrator-tasks" "hmc-ensemble-object" "hmc-ibm-blade-objects" "hmc-ibm-blade-virtual-server-objects" "hmc-policy-administrator-tasks" "hmc-policy-operator-tasks" "hmc-storage-resource-administrator-tasks" "hmc-storage-resource-objects" "hmc-virtual-network-administrator-tasks" "hmc-virtual-network-objects" "hmc-virtual-server-administrator-tasks" "hmc-virtual-server-operator-tasks" "hmc-workload-administrator-tasks" "hmc-workload-objects" "hmc-z-vm-virtual-machine-objects" "hmc-z-vm-virtual-machine-tasks" - Removed all tasks related to zBX and Ensemble components. Enumeration values for the name property of those Task objects are not provided in this HMC version. As that is a considerable number of tasks, they are not listed here.		

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)

Description	HMC MCL	SE MCL
• (cont'd.) • Added the Resend Request operation for Storage Group objects. • Added the enumeration value "auto-created" as a possible value for the action field of a storage-group-action-info nested object related to Storage Group histories. • Corrected HTTP status 404 with reason code 2 to be reason code 1 on the Undefine Storage Switch operation. • Added support for more granular action permissions for the HMC Mobile app, comprising the following API extensions: – Added several new action settings properties, one for each action that supports the new granular permission settings, to the mobile-app-preferences nested object of the Console object. These settings are updated with the Set Mobile App Preferences and included in the response to the Get Mobile App Preferences operation. – New HTTP status 400 with reason code 337 and HTTP status 404 with reason code 2 on the Set Mobile App Preferences operation – Added the user-template-uri property to the User object data model • Added the ability for an API client to request cancellation of the Start Partition, Stop Partition and Start CPC operations with the Cancel Job operation.		
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added detailed information about the processors in a logical partition, including the number of allocated and reserved processors of various types and whether they are dedicated or shared. Added the processor-usage, number-general-purpose-processors, number-reserved-general-purpose-processors, number-general-purpose-cores, number-reserved-general-purpose-cores, number-icf-processors, number-reserved-icf-processors, number-icf-cores, number-reserved-icf-cores, number-ifl-processors, number-reserved-ifl-processors, number-ifl-cores, number-reserved-ifl-cores, number-ziip-processors, number-reserved-ziip-processors, number-ziip-cores, and number-reserved-ziip-cores properties to the Logical Partition object data model. • Added the ability to increase the number of allocated and reserved processor cores in a logical partition via the APIs, comprising the following API extensions: – Added writable properties containing the number of allocated or reserved cores for various processor types to the Logical Partition data model. – Add the authorization requirement of action/task permission for the Logical Processor Add task when increasing the number of allocated or reserved cores. • Added support to control whether Elliptic Curve Cryptography (ECC) keys can be used for computing digital signatures, comprising the following API extensions: – Added the permit-ecc-key-import-functions property to the Image Activation Profile object data model • Added support to determine whether System Recovery Boost is enabled for a logical partition, comprising the following API extensions: – Added the is-sub-capacity-boost-active property to the Logical Partition object data model • Added support to determine whether zIIP Boost is enabled a logical partition, comprising the following API extensions: – Added the is-ziip-capacity-boost-active property to the Logical Partition object data model – Added the enumeration value "boost" as a possible value for the record-type property of a Capacity Record element of a CPC object. – Added the maximum-real-hours, maximum-test-hours, remaining-real-hours and remaining-test-hours properties to the data model for Capacity Record elements of CPC objects.	P46683.065	P46598.076

Table 24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 3.1 to 3.2. - Added API support to manage and interact with the console's multi-factor authentication support provided in conjunction with MFA servers, comprising the following API extensions: - Added the MFA Server Definition object representing the configuration of an MFA server used for HMC authentication and associated List, Create, Delete, Get Properties and Update Properties operations for MFA Server Definition objects. - Added the mfa-types, primary-mfa-server-definition-uri, backup-mfa-server-definition, mfa-policy, mfa-userid and mfa-userid-override properties to the User object data model. - Added the mfa-types, primary-mfa-server-definition-uri, backup-mfa-server-definition, mfa-policy, mfa-userid and mfa-userid-override fields to the request body for the Create User operation. - Added the rsa-secupid-passcode field to the request body for the Logon operation. - Added the shared-secret-key and session-credential fields to the response for the Logon operation. - New HTTP status 400 with reason code 337 and HTTP status 404 with reason codes 338 and 339 on the Create User operation. - New HTTP status 400 with reason code 337, HTTP status 404 with reason codes 338 and 339, and HTTP status 409 with reason code 337 on the Update User Properties operation. - New HTTP status 400 with reason code 53, HTTP status 409 with reason codes 50, 51 and 52, HTTP status 500 with reason code 40, and HTTP status 503 with reason code 40 on the Logon operation. - New HTTP status 409 with reason code 51 on the Establish Shared Secret Key operation. - Added the Provide Requested MFA Information operation. - Added the Change Logon Password operation. - Added API support for fulfilling multiple FICON storage volumes with a single request, comprising the following API extension: - Added the Fulfill FICON Storage Volumes operation to the Storage Group object. - Added support for importing a DPM configuration from a z14 system on which the "dpm-storage-management" feature is enabled, comprising the following API extensions: - Added the following fields to the request body for the Import DPM Configuration operation for CPC objects: storage-sites, storage-subsystems, storage-fabrics, storage-switches, storage-control-units, storage-paths, storage-groups, storage-volumes, storage-templates, storage-template-volumes, virtual-storage-resources, storage-ports, and network-ports. - New HTTP status code 200 on the Import DPM Configuration operation. When HTTP status code 200 is returned a response body is provided that contains information about the parts of the configuration that were not restored.	46683.097	P46598.116, P46650.003

Table 24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0) (continued)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: - Added support to control the enablement of IBM Secure Boot for Linux®, comprising the following API extension: - Added the secure-boot property to the Partition object data model - Added the secure-boot property to the Image Activation Profile object data model - Added the secure-boot property to the Load Activation Profile object data model - Added the last-used-secure-boot property to the Logical Partition object data model - Added the secure-boot field to the request body for the SCSI Load operation for Logical Partition objects. - Added the secure-boot field to the request body for the SCSI Dump operation for Logical Partition objects. - Added support to control whether Elliptic Curve Cryptography (ECC) keys can be used for computing digital signatures, comprising the following API extensions: - Added the permit-ecc-key-import-functions property to the Partition object data model. - New HTTP status 400 with reason code 20 on the Update Partition Properties operation.		

Table 25. Summary of updates for API version 3.3 (HMC/SE Version 2.15.0)		
Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: - Increased API version number from 3.2 to 3.3. - New HTTP status 400 with reason code 54 on the Logon operation. - Added a note to the workload-manager-enabled property of Logical Partition object to point out that changing that property may have no effect in some situations.	46683.098	
Removed the checking for unrecognized X-* HTTP request headers, and, therefore, removed the following common request validation reason code: HTTP status 400 with reason code 10.	46683.099	
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: - New HTTP status 409 with reason 130 and HTTP status 500 with reason 130 on the Start Partition operation. - Added the optional bootloader-error-id field to the job-results object in the asynchronous result of the Start Partition operation.	46683.102	46598.119

Table 26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: - Increased API version number to 3.4. - Added the following as possible enumeration values for the vendor field in the response body for the Query API Version operation: "d", "e", "f", "g", "h" and "i". - New HTTP status 403 with reason code 40 on the Provide Requested MFA Information operation. - Added support for cloud network adapters, comprising the following API extensions: - Added the enumeration value "cna" as a possible value for the type property of an Adapter object. - Added the enumeration value "cna" as a possible value for the adapter-family property of an Adapter object. - Added the following as possible enumeration values for the detected-card-type property of an Adapter object: "ficon-express-16sa", and "cloud-network-x5". - Added the enumeration value "cna" as a possible value for the type property of a NIC element of a Partition object. - Added the function-number and function-range properties to the data model for NIC elements of Partition objects. - Added the function-number and function-range fields to the request body for the Create NIC operation. - New HTTP status 409 with reason code 18 on the Update NIC Properties operation. - New HTTP status 409 with reason code 125 on the Start Partition operation. - Documented that the template-uri and type fields of the request body for the Create Storage Group operation are mutually exclusive. - Added support to request service for Console Hardware Messages, comprising the following API extensions: - Added the Request Console Service, Get Console Service Request Information and Decline Console Service operations for the Console object. - Added the service-supported property to the data model for the hardware-message nested object of the Console object. - Added support to request service for CPC Hardware Messages, comprising the following API extensions: - Added the Request CPC Service, Get CPC Service Request Information and Decline CPC Service operations for the CPC object. - Added the service-supported property to the data model for the hardware-message nested object of CPC objects. - Added additional information about degraded adapters, comprising the following API extensions: - The degraded-adapters property of Partition objects may now include Virtual Storage Resource element URIs. - Added the degraded-reasons property to the data model for Virtual Storage Resource elements of Storage Group objects.	P46683.160	P46598.194

Table 26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0) (continued)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: - Added support to determine whether Secure Execution for Linux is enabled for a logical partition or partition, comprising the following API extensions: - Added the is-secure-execution-enabled, is-global-key-installed and is-host-key-installed properties to the CPC object data model. - Added the is-secure-execution-enabled property to the Logical Partition object data model. - Added the secure-execution property to the Partition object data model. - Added the Power Status metric group for CPC objects. - Added the zcpc-minimum-inlet-air-temperature, zcpc-maximum-inlet-air-temperature, zcpc-maximum-inlet-liquid-temperature and zcpc-environmental-class properties to the CPC object data model.		

Table 27. Summary of updates for API version 3.5 (HMC/SE Version 2.15.0)

Description	HMC MCL	SE MCL
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: - Increased API version number to 3.5. - Corrected the handling of the properties query parameter of the Get CPC Properties and Get Logical Partition Properties operations to no longer return HTTP status 400 for properties the HMC supports but the target CPC or Logical Partition does not support. Such properties are simply omitted from the response.	P46683.169	

Chapter 2. Base definitions

This chapter provides basic definitions of data types, representation formats and other fundamental syntactic elements that apply across the Web Services API.

Data types

The following data types are used in the definition of the management data model, input and output parameters and notification message formats in the Web Services API.

Table 28. Primitive data types	
Data type	Description
Boolean	A logical truth value: either the value true or the value false .
Byte	An integer value in the range -2^7 to $(2^7)-1$ (the range of a signed 8-bit integer)
Float	An IEEE 754 floating point number in the range $\pm 4.9\text{E}-324$ to $\pm 3.4028235\text{E}+38$. Note that, although IEEE 754 provides for representations of positive or negative Infinity and NaN, such values are not used within the API.
Long	An integer value in the range -2^{63} to $(2^{63})-1$ (the range of a signed 64-bit integer)
Integer	An integer value in the range -2^{31} to $(2^{31})-1$ (the range of a signed 32-bit integer)
Short	An integer value in the range -2^{15} to $(2^{15})-1$ (the range of a signed 16-bit integer)
String	A sequence of Unicode characters. When the number of characters in the string is bounded, the length or length range is provided in parenthesis, for example String (16) for a 16 character string, or String (0-256) for a string that may range in length from 0 (empty) to 256 characters.
String Enum	A String enumeration, i.e. a String for which the possible values are constrained to be one of a specified set of choices.
String/ URI	A String that contains a URI path used to designate object instances or operations within the API.
String/IPV4 Address	A String that contains an Internet Protocol Version 4 address presented in dotted-decimal notation. Example: "127.0.0.1"
String/IPV6 Address	A string that contains an Internet Protocol Version 6 address presented in colon-separated-hexadecimal notation. Leading and consecutive groups of zeros may be omitted in the representation as is conventional for IPV6 addresses presented in this form. Example: "2001:db8:85a3::8a2e:370:7334"

Table 28. Primitive data types (continued)

Data type	Description
String/Hostname	A string that contains an internet hostname that adheres to the following standard guidelines similar to those in the Internet Engineering Task Force (IETF) RFC 1123: • length is 1-255 characters • valid characters are a-z, A-Z, 0-9, period(.), and hyphen(-) • must not begin or end with a period • must not contain consecutive periods • must not be an IPv4 address in dotted-decimal notation (see the String/IPv4 Address datatype)
Timestamp	A non-negative Long integer quantity where the value represents a date and time expressed as the number of milliseconds since midnight on January 1, 1970 UTC, or the special value -1 to indicate special treatment of the timestamp field.

Table 29. Compound data types

Data type	Description
Array of <T>	A ordered sequence of zero or more elements each of data type <T>. An array may be empty, i.e. have no elements contained within it.
Object	A nested data structure providing a set of fields, each field having a name, data type and value. Object types do not formally have names. However, descriptions of these nested objects will often assign reference names to allow connections to be made in the documentation between points of use and definition for a given nested object.

Input and output representation

Except for a few special cases, the operations provided by the Web Services API expect their input and provide their output using a representation known as JavaScript Object Notation, or JSON for short. The JSON representation is also used within the bodies of notification messages emitted by the API. Unless some different representation is specifically mentioned in the description of an operation or message, all operations and messages should be understood to use JSON notation.

JavaScript Object Notation (JSON) is a lightweight, text-based, language-independent data interchange format that defines a small set of formatting rules for the portable representation of structured data. JSON can represent four primitive types (strings, numbers, booleans, and the value null) and two structured types (objects and arrays) that together provide sufficient expressive power to represent the manageable resource configuration, state, inputs, and outputs that appear in this API.

A JSON string is a sequence of zero or more Unicode characters enclosed in quotes.

A JSON object is an unordered collection of zero or more name/value pairs (sometimes referred to in this document as fields or properties), where a name is a string and a value is a primitive type (string, number, boolean, or null), an array, or a nested object. Each name/value pair is represented in the form **"name": value** and is separated from the next name/value pair by a comma. The collection of name/value pairs comprising the object is enclosed by left and right braces e.g. { ... }.

An array is an ordered sequence of zero or more values separated from each other by commas and enclosed in left and right square brackets e.g. [10, 20, 30]). The values in the array can be primitive or structured types, i.e. arrays of objects or arrays of arrays are permitted.

The precise BNF syntax of JSON notation is not provided in this document, but can be found in the IETF information document RFC 4726, The *application/json Media Type for JavaScript Object Notation (JSON)*, July 2006. This RFC can be found in text format on the World Wide Web at:

<http://www.ietf.org/rfc/rfc4627.txt>

Representing API data types in JSON

The following tables define the mapping between the API data types and their corresponding representation in JSON notation.

Table 30. Primitive data types in JSON notation	
API data type	JSON representation
Boolean	A JSON boolean with keywords true and false
Byte, Integer, Long, Short	A JSON number with a sign and integer component, but no fraction or exponent part.
Float	A JSON number, possibly including fraction or exponent parts. On output, values with a magnitude greater than or equal to 10^{-3} and less than 10^7 are representation in floating-point format with a fraction part but not exponent part (e.g. 1.7, -32.467). Values with magnitudes outside that range are represented in scientific notation with both fraction and exponent parts (e.g. -4.23E127).
String, String Enum	Represented as a JSON string enclosed in quotes.
Timestamp	An unsigned JSON number with integer component, but no fraction or exponent part.

Table 31. Compound data types in JSON notation	
Data type	Description
Array of <T>	A JSON square-bracket-enclosed array with elements represented according to the data type <T>.
Object	A JSON curly-brace-enclosed object, with the fields/properties of the nested object represented as name/value members of the object. The name of a property/field is used directly as the name part of the JSON object member, and the value of the field/property is provided as the value part of the member.

All strings in the JSON representation (object member names, and string values) are encoded in UTF-8.

Chapter 3. Invoking API operations

The Web Services API provides an extensive set of operations that client applications can invoke to obtain information about the manageable resources of the system, to change those resources' characteristics, and to take action on them. Because the API is designed using a web services orientation, these operations are accessed by means of Hypertext Transport Protocol (HTTP) protocol messages flowing across TCP/IP network connections.

Most aspects of HTTP protocol usage required to invoke API operations or receive responses apply universally across all of the operations of the API. Rather than repeat these details in the description of each and every operation, this common information is instead provided in this chapter. The material in this chapter should be considered to apply to each and every operation of the API unless the operation-specific description indicates otherwise. Thus, the information in this chapter should be consulted in conjunction with the operation-specific descriptions elsewhere in this document when determining how to invoke a specific API operation.

HTTP protocol standard

The Web Services API has been designed in accordance with the HTTP version 1.1 protocol, as defined in the W3C internet standards document *RFC 2616, Hypertext Transfer Protocol – HTTP/1.1, June 1999*. This RFC can be found in HTML format on the World Wide Web at: <http://www.w3.org/Protocols/rfc2616/rfc2616.html>

The API requires that all clients interact using the HTTP/1.1 protocol. The API does not support clients that use HTTP/1.0.

Note: While the API does not specifically assume or exclude any particular client user agent, its use and interpretation of HTTP elements has been designed presuming that the client application interacting with the API is a programmatic web application client or HTTP-capable scripting client rather than a standard browser-based application.

Connecting to the API HTTP server

When the Web Services API is enabled, the HMC API HTTP server listens for SSL-based socket connections on TCP port 6794. The HMC is enabled for both the SSL version 3 and TLS version 1 protocols on this SSL port. It does not accept non-SSL connections. The set of cipher suites enabled for the HMC API HTTP server is controlled by the **Certificate Management** task on the HMC. Note, the default set of cipher suites may change with updates to the HMC if one or more of the cipher suites are found to be weak or vulnerable.

The listening port for the API HTTP server is a fixed port number and is not subject to customer reconfiguration. Thus, client applications can treat this as a well-known port number rather than requiring customer input when configuring the networking parameters the client will use to connect to the HMC.

HTTP header field usage

HTTP request and response messages include elements known as header fields (often referred to simply as headers for short) that provide request metadata. Certain headers are required or provided in all HTTP messages, while others are present in selected messages depending on content.

This section describes the use of header fields by the Web Services API.

Required request header fields

The following HTTP request headers are relevant to all request methods (GET, PUT, POST, DELETE) and are required on all API requests (except as indicated for the Logon and Query API Version operations).

HTTP header name	Rqd/Opt	Description
Host	Required	Specifies the Internet host and port number of the HMC to which the request is being directed, as obtained from the original URI given by the client application. The Web Services API enforces that this header is provided as required by the HTTP protocol, but does not check or use the value of the header in any way.
X-API-Session	Required ¹	An opaque string that provides a cryptographically strong identifier of the API session (known as a session id) under which this request is executed. This header is required on all requests that require authentication. The Logon operation begins a new HMC session and includes credentials identifying the HMC user for the session. Upon successful authentication, the Logon operation returns the value to be used in the X-API-Session header for all subsequent requests of the same session. Failure to include this header on a request requiring authentication results in status code 403 (Forbidden) with reason code 4. Specifying an invalid session id results in status code 403 (Forbidden) with reason code 5.
Note: 1. Not required on requests to the Query API Version and Logon operations since these operations can be performed before an API session has been established.		

For requests made using the HTTP PUT or POST methods, the following additional request headers are required if a request body is being provided. If an operation being requested via POST method does not require a request body, these headers can be omitted.

HTTP header name	Rqd/Opt	Description
Content-Length	Required if request body present	When used in a request, specifies the length of the request body. If omitted, the request is presumed to not contain a body. The API limits the size of request bodies in order to control usage of memory resources on the HMC. Unless a different limit is specified for a particular operation, in general the largest request body accepted by the API is 64KB. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.
Content-Type	Required if request body present	When used in a request, specifies the MIME media type of the request body contained in the request. This header is required if the Content-Length header is supplied and specifies a nonzero request body length, otherwise status code 400 (Bad Request) will result.

Optional request headers

The following HTTP request headers are relevant to all request methods (GET, PUT, POST, DELETE) and may be specified on these method requests but are not required. If present, they are interpreted by the API in the indicated way.

HTTP header name	Rqd/Opt	Description
Accept	Optional	Specifies the list of response MIME media types that the client application is prepared to accept for the response to the request. This header is provides for content negotiation between the client and the server in cases where the Web Services API supports multiple possible response media types for a given operation. In the current implementation, the Web Services API supports only a single response media type for each operation. For the majority of operations, that media type is JSON (application/json), but selected operations support a different media type (indicated in the descriptions of those special operations). If this header is omitted, the Web Services API responds using the (single) media type supported for the operation. If the header is included, it must allow for the single media type that the operation supports, otherwise the request will fail with HTTP status code 406 (Not Acceptable). If an operation is extended to support multiple media types, compatibility will be maintained for existing clients that request the operation without specifying an Accept header.
X-Audit-Id	Optional	A string that provides additional client identity information that is included in all audit records created for this request, in addition to the API user's HMC login identity. This header is intended to provide improved audit logging in the case of clients that make requests on behalf of multiple upstream users while logged into the API under a single HMC login identity. Such clients should provide the identity of their upstream user in this header so that the requests of different upstream users can be distinguished in the HMC audit logs. The HMC will use up to the first 64 characters of information from this header if present, and silently ignore the remainder of the header's value if it is longer than 64 characters.
X-Client-Correlator	Optional	A string that provides diagnostic information pertaining to this request that is of significance to the client, such as a client request number or the like. The HMC will record this information in selected diagnostic trace or log data it collects so as to allow better cross-correlation of this information with similar information maintained by the client. This data supplied in this header is intended to assist in product problem determination and does not otherwise affect the operation of the API. The HMC will use up to the first 64 characters of information from this header if present, and silently ignore the remainder of the header's value if it is longer than 64 characters.

Standard response headers

The following HTTP response headers are always provided in the response to all requests.

HTTP header name	Description
Date	The date and time, from the perspective of the HMC's clock, at which the response message was generated. As required by the HTTP protocol specification, this date is an HTTP full date sent in the RFC 1123-defined fixed length format. Example: Sun, 08 Oct 1961 10:08:00 GMT

The following HTTP response headers are provided in the response to all requests except those that result in a 204 (No Content) HTTP status code.

HTTP header name	Description
Content-Length	When used in a response, specifies the length of the response body. If omitted, the response does not contain a body.
Content-Type	When used in a response, specifies the MIME media type of the response body. This response header is provided any time the Content-Length header is provided and specifies a nonzero length.

Additional response headers

Some operations may return additional response headers beyond those described in “[Standard response headers](#)” on page 45. The following table describes these possible additional response headers. Operations that return these additional headers indicate that they do so in the operation description.

HTTP header name	Description
Location	The URI of the resources that was created by the operation. This header is provided for operations that complete successfully with an HTTP status code of 201 (Created).
X-API-Session	An opaque string that provides a cryptographically strong identifier of the API session that was created for the client. This header is provided in the response to a successful Logon operation.
X-Request-Id	A string that provides diagnostic information identifying the request from the perspective of the HMC. This same information is included in the API log entries that are recorded by the HMC for the request. If captured by the client from a response, a client application developer or support technician can use this information to locate the HMC API log entry corresponding to a particular request. The value of this header will be 64 characters or less. This header is provided in the responses to all requests.

Media types

The following media types are applicable to the use of the Web Services API, and thus may appear in the values of **Accept** or **Content-Type** header fields.

MIME media type	Description
application/json	JavaScript Object Notation (JSON), as described by RFC 4627. This media type is used by the Web Services API for both request and response representation for the majority of the operations in the API. The JSON text is encoded using the UTF-8 charset.
application/vnd.ibm-z-zmanager-metrics	Custom output format used for providing the results to the Get Metrics operation of the metrics service. The result text is encoded using the UTF-8 charset.

MIME media type	Description
application/xml	Extensible Markup Language, used for the input and output formats for the Export Performance Policy and Import Performance Policy operations of the workload object. The XML text is encoded using the UTF-8 charset.
application/octet-stream	Binary data. This media type is used by the Web Services API for the request representation for the Mount Virtual Media Image operation of the Virtual Server object.

HTTP status codes

The HMC API provides standard HTTP status codes in the response to requests to indicate the success or failure of the request. Unless stated otherwise in the description of an operation, the following general interpretations of the status code values apply.

HTTP status code	Description/Causes
200 (OK)	The request has succeeded completely. A response body is provided that contains the results of the request.
201 (Created)	The request has succeeded completely and resulted in the creation of a new managed resource/object. The URI for the newly created managed resource is provided in a Location header. (POST methods only)
202 (Accepted)	The request was successfully validated and has been accepted to be carried out asynchronously.
204 (No Content)	The request succeeded completely, and no additional response information is provided.
400 (Bad Request)	The request was missing required input, had errors in the provided input, or included extraneous input. Additional information regarding the error is provided in an error response body that includes a reason code with additional information.
403 (Forbidden)	Multiple error conditions result in this status code: - The request requires authentication but no X-API-Session header was provided, or one was provided but the session ID was invalid. - The user under which the API request was authenticated is not authorized to perform the requested operation.
404 (Not Found)	The URI does not designate an extant resource, or designates a resource for which the API user does not have object-access permission.
405 (Method Not Allowed)	The request specifies an HTTP method that is not valid for the designated URI.
406 (Not Acceptable)	The Accept header for the request does not include at least one content representation supported by the Web Services API.
409 (Conflict)	The managed resource is in an incorrect state (status) for performing the requested operation. Additional information regarding the error is provided in an error response body that includes a reason code with additional information.

HTTP status code	Description/Causes
413 (Request Entity Too Large)	The request includes a request body that is too large. Unless a different limit is specified for a particular operation, in general the largest request body accepted by the API is 64 KB.
415 (Unsupported Media Type)	The Content-Type header for the request specifies a representation that is not supported by the Web Services API.
500 (Server Error)	A server error occurred during processing of the request.
501 (Not Implemented)	The request specifies an HTTP method that is not recognized by the server (for any resource). Note: The response body that accompanies this error is not a JSON response body as defined in “Error response bodies” on page 48 .
503 (Service Unavailable)	The request could not be carried out by the HMC due to some temporary condition.
505 (HTTP Version Not Supported)	The request specifies an HTTP protocol version that is not supported by the Web Services API.

Error response bodies

For most 4xx and 5xx HTTP error status codes, additional diagnostic information beyond the HTTP status code is provided in the response body for the request. The API client can use the **content-type** header to determine the type of information in the response body. If the value of the **content-type** header is **application/json**, the following information is provided in the form of a JSON object containing the following fields:

Field name	Type	Description
request-method	String	The HTTP method (DELETE, GET, POST, PUT) that caused this error response.
request-uri	String	The URI that caused this error response.
request-query-parms	Array of query-parm-info objects	An array of query-parm-info objects (described in the table below) that identify the query parameters specified on the request and their values. Each query-parm-info object identifies a single query parameter by its name and includes its value(s). If the request contains no query parameters, this field is omitted.
request-headers	header-info object	A header-info object (described in the table below) that describes the HTTP headers specified on the request. If the request contains no HTTP headers, this field is omitted.
request-authenticated-as	String	The name of the HMC user associated with the API session under which the request was issued. If the request was issued without an established session or there is no HMC user bound to the session, this field is omitted.

Field name	Type	Description
request-body	String	The request body, in the form of a JSON document. Note that, since it is in the form of a JSON document, this may not be exactly what was submitted by the API client program, but it is semantically equivalent. If the request body could not be parsed or some other error prevented the creation of a JSON document from the request body, this field is omitted and the request body is instead available in the request-body-as-string field.
request-body-as-string	String	The complete request body, or some portion of the request body, exactly as it was submitted by the API client program. The request-body-as-string-partial field indicates whether the complete request body is provided. If the request-body field is present, this field is omitted.
request-body-as-string-partial	Boolean	When the request-body-as-string field is present, this boolean indicates whether the request-body-as-string field contains only part of the request body (true) or the entire request body (false). If the request-body-as-string field is not present, this field is omitted.
http-status	Integer	HTTP status code for the request.
reason	Integer	Numeric reason code providing more details as to the nature of the error than is provided by the HTTP status code itself. This reason code is treated as a sub-code of the HTTP status code and thus must be used in conjunction with the HTTP status code to determine the error condition. Standard reason codes that apply across the entire API are described in “Common request validation reason codes” on page 50. Additional operation-specific reason codes may also be documented in the description of the specific API operations.
message	String	Message describing the error. This message is not currently localized.
stack	String	Internal HMC diagnostic information for the error. This field is supplied only on selected 5xx HTTP status codes.
error-details	Object	A nested object that provides additional operation-specific error information. This field is provided by selected operations, and the format of the nested object is as described by that operation.

Each query-param-info object contains the following fields:

Field name	Type	Description
<i>{query_parm_name}</i>	Array of Strings	The value of the <i>{query_parm_name}</i> query parameter. If this query parameter was specified multiple times, there will be multiple entries in this array, one for each instance of this query parameter.

The header-info object contains the following field(s), one for each header present on the request:

Field name	Type	Description
<i>{header_name}</i>	String or Array of Strings	The value of the <i>{header_name}</i> HTTP request header. It will be either a single string or an array of strings.

Usage notes:

- The message provided in the **message** field is primarily intended as a convenience for use by developers when developing and testing client applications. Because it is not localized, it may not be appropriate for client applications to simply pass this message on to their clients when reporting errors to those upstream clients. Instead, client applications can use the value in the **reason** field as a key in obtaining a client-provided message that may be more appropriate to use.

- Because the reason code is treated as a sub-code of the HTTP status code, the same reason code value is often defined for multiple different HTTP status codes and has a different meaning in each case. For example, reason code 1 when considered for a 400 (Bad Request) status code has a different meaning than when considered for a 403 (Forbidden) status code. For this reason, client applications that make decisions based on the reason codes should always include checking the HTTP status code as part of the relevant logic (e.g. test for status code == 400 AND reason code == 1, not just reason code == 1 alone).

Common request validation reason codes

The Web Services API performs request validation on each request it receives to ensure the request is correctly formed and appropriate before it begins processing the request. Many errors of basic request syntax can occur on all or a large number of the operations provided by the API. Validation for these kinds of errors is done in a common way across all of the operations and results in a common (not request-specific) reason code being reported if errors are detected. Other validation is operation-specific by nature, and results in operation-specific reason codes when errors are detected.

The following table provides the HTTP status codes and reason codes for common request validation. These status and reason codes may be reported on any of the operations of the API.

HTTP status code	Reason code	Description
400 (Bad Request)	1	The request included an unrecognized or unsupported query parameter.
	2	A required request header is missing or invalid.
	3	A required request body is missing.
	4	A request body was specified when not expected.
	5	A required request body field is missing.
	6	The request body contains an unrecognized field (i.e. one that is not listed as either required or optional in the specification for the request body format for the operation).
	7	The data type of a field in the request body is not as expected, or its value is not in the range permitted.
	8	The value of a field does not provide a unique value for the corresponding data model property as required.
	9	The request body is not a well-formed JSON document.
	11	The length of the supplied request body does not match the value specified in the Content-Length header.
	13	The maximum number of logged in user sessions for this user ID has been reached; no more are allowed.
	14	Query parameters on the request are malformed or specify a value that is invalid for this operation. Common causes include the inability to successfully decode a parameter element, the presented parameters are not in the expected key=value format, the value is not a valid regular expression, a required parameter is missing, multiple instances of a parameter are present on an operation that does not permit multiple instances of that parameter, or the value is not a valid enum for the operation.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body. A prerequisite condition or dependency among request body fields is not met.
	18	The request body contains a field whose presence or value is inconsistent with the type of the object. Such a requirement is often described in an object's data model as the field having a prerequisite condition on a "type", "family", or similar property that identifies an object as being of a particular type. Such a property is typically, but not necessarily, immutable.
	19	The request body contains a field whose corresponding data model property is no longer writable. In certain earlier HMC and/or SE versions the property is writable, but later versions do not allow changing the property through the Web Services APIs. This could be due to a change in the underlying system-management model, or the property may have become obsolete.
	20	The request body contains a field or value that is directly or indirectly dependent on the version of the SE that is targeted by or associated with the request operation, and that SE is not at a version that supports or provides the field or value.

HTTP status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
	4	The request requires authentication but no X-API-Session header was specified in the request.
	5	An X-API-Session header was provided but the session id specified in that header is not valid.
	301	The operation cannot be performed because it targets a CPC that does not support Web Services API operations.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. For URIs that contain object ID and/or element ID components, this reason code may be used for issues accessing the resource identified by the first (leftmost) such ID in the URI.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. For URIs that contain object ID and/or element ID components, this reason code may be used for issues accessing the resource identified by the first (leftmost) such ID in the URI.
	4	The object designated by the request URI does not support the requested operation.
	5	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. More specifically, this reason code indicates issues accessing the resource identified by the element ID component in the URI. Such an element ID is typically the second (counting left to right) ID component in the URI.
	6	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. More specifically this reason code indicates issues accessing the resource identified by the element ID component in the URI. Such an element ID is typically the second (counting left to right) ID component in the URI.

HTTP status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	3	The operation cannot be performed because the object designated by the request URI is currently locked to prevent disruptive changes from being made.
	4	The operation cannot be performed because the CPC designated by the request URI is currently enabled for DPM.
	5	The operation cannot be performed because the CPC designated by the request URI is currently not enabled for DPM.
	6	The operation cannot be performed because the object hosting the object designated by the request URI is not in the correct state.
	8	The operation cannot be performed because the request would result in the object being placed into a state that is inconsistent with its data model or other requirements. The request body contains a field whose presence or value is inconsistent with the current state of the object or some aspect of the system, and thus a prerequisite condition or dependency would no longer be met.
	9	The operation cannot be completed because it is attempting to update an effective property when the object is not in a state in which effective properties are applicable. More specifically, the request body contains one or more fields which correspond to a property marked with the (e) qualifier in the data model, and the object's effective-properties-apply property is false .
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	11	The operation cannot be performed because it requires a fully authenticated session and the API session that issued it is only partially authenticated.
	12	The operation cannot be performed because a feature that prohibits the operation is currently enabled. The error-details field of the response body contains an error-feature-info object identifying the feature whose current enablement status is invalid for the operation. The error-feature-info object is described in the next table.
	13	The operation cannot be performed because a feature required by the operation is currently disabled. The error-details field of the response body contains an error-feature-info object identifying the feature whose current enablement status is invalid for the operation. The error-feature-info object is described in the next table.

The error-feature-info object contains the following fields:

Table 32. error-feature-info object properties

Field name	Type	Description
scope	String Enum	The scope of the feature requirement. The valid values are: • "cpc" - The feature enablement requirement applies to the CPC object involved in the operation. • "partition" - The feature enablement requirement applies to the Partition object involved in the operation.
name	String Enum	The name of the feature requirement. • When the scope is "cpc", this is a feature name from the CPC's available-features-list property. • When the scope is "partition", this is a feature name from the Partition's available-features-list property.

Common request processing reason codes

Certain common error conditions can be encountered during the processing of many of the operations of the API. When they are encountered they are reported using the same HTTP status and reason code by any operation of the API that may encounter them.

These common request processing reason codes are listed in the following table:

HTTP status code	Reason code	Description
500 (Server Error)	0 - 39	An internal processing error has occurred and no additional details are documented.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.
	3	This request would exceed the limit on the number of concurrent API requests allowed.

Use of chunked response encoding

For most API operations, the size of the response data is modest and therefore standard HTTP response payload transfer encoding is used. In this encoding, the length of the entire payload of the response message is provided in the response before any of the contents of the response payload are written to the socket connection. But some operations, such as the Get Inventory operation of the Inventory service and the Get Metrics operation of the Metrics service, can produce very large responses. Use of standard transfer encoding for these kinds of operations is inefficient for the HMC because it requires the entire response be generated and buffered before any of it is sent in order to compute and send the total length of the response body before sending any of the contents of the response data.

To avoid the need for the buffering the entire response, and to instead allow the response to be transmitted in smaller segments as they are prepared, operations that return large responses use HTTP chunked response encoding instead. Chunked transfer encoding is an HTTP V1.1 data transfer feature that allows the payload of the response message to be split into a sequence of smaller parts known as chunks, with the size of each chunk transmitted as part of the chunk rather than requiring the transmission of the size of the full response payload.

Chunked transfer encoding is defined in the HTTP/1.1 protocol standard, RFC 2616, cited earlier in this section.

The HTTP protocol standard requires that all HTTP/1.1 applications (client or server) be capable of receiving and handling chunked transfer-encoded messages, so the use of this encoding by the API HTTP server is within the options allowed by the protocol standard. However, since this format may be unexpected to naively-written applications, its use is limited by the API HTTP server to the special circumstances that warrant its use to improve performance or efficiency. Therefore, a client application can safely assume that an operation will not use chunked transfer encoding for its responses unless the use of this encoding is specifically mentioned in the description of the operation.

Filter query parameters

Some operations allow for the (optional) use of designated query parameters for conveying additional request parameters. Although query parameters can be used to convey various kinds of additional request information, most operations that make use of query parameters do so for the purpose of filtering the response entries to a subset of what would otherwise be returned. For example, this kind of filtering is typically provided on operations that are described as List operations (e.g. `List Logical Partitions of a CPC`). This section describes the interpretation/handling of filter-type query parameters across all of the operations of the API.

As would be expected, if an operation is invoked without specifying any of its possible filter-type query parameters, the operation returns all of the result entries applicable to the request. For example, the `List Logical Partitions of a CPC` operation invoked with no filtering query parameters returns all of the Logical Partition objects in the CPC to which the API user has access.

If one or more filter-type query parameters are specified, the combination of those parameters specifies a logical match expression that is evaluated against each entry that is a candidate for inclusion in the result to determine if the entry is included or not. Within that expression, there may be multiple occurrences of the same-named query parameter and/or there may be occurrences of differently-named query parameters. The query parameters are interpreted as a logical expression using the following rules:

- Multiple occurrences of the same-named query parameter are interpreted as a group that is connected by a logical OR operation among all of query parameters with the same name. An entry remains a candidate for inclusion in the result as long as it matches at least one of the values specified for this particular query parameter.
- Occurrences of differently-named query parameters are first organized into OR'ed groups as mentioned above, and then these groups are interpreted as being connected by logical AND operations. Thus an entry is included in the result only if it matches at least one value from each of the differently-named groups of parameters.
- As an example, a query string of `"name=fee&type=fie&name=foe&type=fum"` is interpreted as specifying the expression `(name=fee OR name=foe) AND (type=fie OR type=fum)`. Note that the order in which the query parameters appear in the string is not important.

As a filter-type query parameter is applied against a candidate entry, it is determined to match or not as follows:

- If the query parameter is of data type String, the parameter's value is interpreted as a regular expression pattern and is considered to match if the corresponding String property of the candidate entry matches the pattern.
- If the query parameter is of data type String Enum, the parameter's value is compared against the corresponding Enum property of the candidate entry and is considered to match if they are exactly the same value.
- If the query parameter is of data type Boolean, the parameter's value is compared against the corresponding Boolean property of the candidate entry and is considered to match if they are both true or both false.
- If the query parameter is of data type String/ URI, the parameter's value is interpreted as a canonical URI path and is considered to match if the corresponding property of the candidate entry is exactly the same value.

Regular expression syntax

The values of String-type filtering query parameters are interpreted as regular expressions. The regular expression syntax used is the same as that used by the Java programming language, as specified for the `java.util.regex.Pattern` class. Documentation on that syntax can be found at on the following web page: <http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html>

Chapter 4. Asynchronous notification

The Web Services API includes an asynchronous notification facility by which client applications may subscribe to and receive notification messages regarding a set of predefined management events. These events include:

- Addition and removal of managed objects to/from the inventory of resources that are managed by the HMC.
- Changes to specified properties of managed object instances.
- Changes to the operational status of managed objects.
- Completion of asynchronously processed jobs.
- Addition of entries to the HMC's audit log.
- Addition of entries to the HMC's security log.
- New and refreshed operating system messages.

The zManager notification facility is based on the Java Message Service (JMS) architecture and API for exchanging messages among two or more applications.

JMS basics

In the JMS model, message-based communication between producing and consuming applications is coordinated by an intermediate component known as a message broker that acts as the clearinghouse for message exchange. The message broker provides a registry of the available destinations to which messages can be posted, and a store for messages that have been posted but not yet consumed.

Applications acting in the role of message producer create messages and post them (via the broker) to the destination appropriate for the type of message. The messages are associated with the destination and retained by the broker until they have been consumed by interested applications.

Applications acting in the role of message consumer connect to a message destination (again, via the broker) in order to express interest in receiving messages posted to it. As messages are posted to the destination by producers, the broker makes the messages available to interested consumers which then receive and process the message.

In the Web Services API notification facility, the HMC acts both as the message broker and the message producer for API notification messages. API client applications act as message consumers.

For the broker function, the HMC includes an integrated JMS message broker implementation based on Apache ActiveMQ, an open, standards-based implementation of JMS. This integrated broker is configured to allow internal HMC function to act as message producers, and to allow external applications to connect as message consumers. However, external applications cannot produce and send messages using the HMC integrated broker.

Connecting to the API message broker

As part of the Web Services API, the HMC provides an integrated JMS message broker based on Apache ActiveMQ Version 5.2.0. This message broker is active on the HMC whenever the Web Services API is enabled.

When active, the integrated broker listens for client connections using the following transports supported by ActiveMQ:

- OpenWire flowing over SSL connections, listening port: 61617.

- STOMP (Streaming Text Oriented Messaging Protocol) flowing over SSL connections, listening port: 61612.

The broker is enabled for the SSL version 3 and TLS version 1 protocols on these SSL ports.

The listening ports for the message broker are fixed port numbers and are not subject to customer reconfiguration. Thus, client applications can treat them as well-known port numbers rather than requiring customer input when configuring the networking parameters the client will use to connect to the HMC.

In order to connect to the integrated message broker, clients must provide certain authentication information for the HMC user making the connection. If the HMC user is configured to use multi-factor authentication, then the client must provide information about the HMC user's API session that is to be associated with the message broker connection. The user name field in the connection request must contain the API session ID, and the password field must contain the session-specific authentication credential. Both of these are available in the response body of the Logon operation that created the API session: the **api-session** and **session-credential** fields, respectively.

If the HMC user is not configured to use multi-factor authentication, then the client may provide the authentication information as described immediately above, or it may provide a valid HMC user name and logon password in order to identify the HMC user making the connection. The user name and password are validated using the standard HMC user authentication mechanisms before allowing the connection to succeed.

The integrated message broker does not allow any anonymous or unauthenticated connections.

Per-session notification topics

As part of its access control enforcement, the Web Services API limits the distribution of notification messages to clients that have object-access permission to the managed object for which the notification was generated.

In order to accomplish this, the API does not define a single (or fixed number of) notification topics to which all messages are posted and from which any or all clients can receive messages. Rather, the API uses per-session notification topics.

In this approach, each API session is associated with a set of JMS destinations that are created by the HMC when the session is created or other actions are performed using the session, and are used for providing notifications destined to that session. The names of the object notification and job completion destinations are returned as part of the results from the Logon operation. The names of all destinations available to a session are returned by the `Get Notification Topics` operation. Each session has the following per-session notification destinations:

- An object notification topic, used by the HMC to send notifications that pertain to the inventory and status of managed objects that this session has permission to access.
- A job notification topic, used by the HMC to send notifications that pertain to the status of asynchronous operations that are initiated by this session.
- An audit notification topic, used by the HMC to send notifications that pertain to the HMC's audit log, but only if the client has permission to the **Audit and Log Management** task. Without the required task permission, there is no such destination available to the session.
- A security notification topic, used by the HMC to send notifications that pertain to the HMC's security log, but only if the client has permission to the **View Security Logs** task. Without the required task permission, there is no such destination available to the session.
- An operating system message notification topic, used by the HMC to send notifications that pertain to new and refreshed operating system messages. More than one topic of this type can exist for a single API session.

Unlike the other topic types, topics of this type are not created by default when the session is created. They are created when the user targets the `Open OS Message Channel` operation at a Logical Partition or Partition object. Operating system messages begin to flow after the user first connects to

the topic. If refresh messages are desired, they are published to the topic immediately following a connection being established to the topic. New messages are sent as they are received. When there are no connections remaining to the topic, the flow of messages stops. The topic will persist, allowing for a reconnect. If the user reconnects, messages will flow as if it was the first connect. The topic is destroyed only when the user performs the Logoff operation or the session is otherwise destroyed.

The session is also associated with an HMC user (identified during API session logon) that in turn has a set of object-access permissions defined for it that determine the managed resources that it is authorized to access. The HMC user also has a set of task permissions defined for it that determine the tasks that it is authorized to perform.

As notification messages are created for managed resource changes or other events, they are distributed to sessions according to the object-access permissions of those sessions. More specifically, when a notification is generated pertaining to some managed resource, it is published to the object notification topics of all sessions for which the related API user has object-access permission to that managed resource, and is omitted from the object notification topics of sessions for which the user does not have object-access permission. As a result, a particular API session will have access to all notifications for all managed resources to which its API user has access permission, but will not have access to notifications for managed resources that it does not.

Notification messages for job completion are sent only to the job notification topic of the API session that initiated the asynchronous processing represented by the job.

Notification message formats

Six types of notification messages are provided by the API. The JMS messages created for all types of notifications share a common set of message characteristics and header fields, which are extended with additional header fields and message body formats that vary by the type of notification.

Common message characteristics

The characteristics described in this section apply to all notification messages published by the Web Services API.

Message format

The following JMS message characteristics apply to all notification messages sent by the Web Services API. These characteristics are echoed in the message themselves in the values of the standard JMS message header fields.

Characteristic	Description
Destination	The per-session object or job notification topic as indicated below for each type of notification.
Message type	Text message. The contents of the body vary by the type of notification.
Delivery mode	Nonpersistent.
Expiration	No expiration.
Priority	4 (highest normal priority)
Message ID	A unique message ID for the message.
Correlation ID	Not set by the API.
Timestamp	The time, from the HMC's perspective, that this message was sent.

In addition to the standard JMS message headers, the following additional message properties are provided in all notification messages to allow for message selection:

Message Property Name	Description
notification-type	The type of notification contained in this message, varies by notification type.
session-sequence-nr	The sequence number of this notification within the session. This number starts at 0 when the API session is created, and is incremented each time a notification message is published to this session.
global-sequence-nr	The sequence number of this notification from the HMC. This number starts at 0 when the HMC is started, and is incremented each time a notification message is published to any API session.
object-uri	The current value of the object-uri property (canonical URI) of the managed object for which the notification is being emitted.
object-id	The current value of the object-id property (durable unique identifier) of the managed object.
element-uri	The current value of the element-uri property of the element object for which the notification is being emitted. This message property is included only when the message pertains to an element object of a managed object.
element-id	The current value of the element-id property (local identifier) of the element object. This message property is included only when the message pertains to an element object of a managed object.
class	The current value of the class property (kind of object) of the object, i.e. the kind of object for which the notification is being emitted.
name	The current value of the name property (display name) for the object to which the notification pertains. Note: Note: In some circumstances the name property may be unavailable, in which case this field is set to an empty string. This may occur, for example, if a property change occurs and is to be reported on very shortly before (essentially concurrent with) the removal of that object from the inventory.

When a notification message pertains to an element object, the message includes **element-uri** and **element-id** fields in addition to **object-uri** and **object-id** fields. The element-* fields identify the element object instance while the object-* fields identify the containing managed object instance. In this case, the **class** field provides the class of the element object, and the **name** field provides the name of the element object.

When a notification message pertains to a managed object, the message contains **object-uri** and **object-id** fields but not the element-* fields and the **class** field provides the class of the managed object and the **name** field provides the name of the managed object.

Grouping of notifications

A particular managed resource may often experience a series of changes that occur in rapid succession. This might occur, for example, when a user uses an object's Details task in the HMC UI to make a set of changes to the object's properties and then selects the Finish button to complete the task. All of the pending property changes collected by the task will be made on the managed object in very quick succession in response to the Finish button being selected, rather than before as the user was interacting with the task.

In order to reduce the notification traffic in these situations, the notification messages have been designed to allow the HMC to report multiple changes of the same type (e.g. property changes, status changes) for the same managed resource in a single message rather than generating a distinct message for each change. In order to do such grouping, the HMC may delay generation of a notification message

for a change for a brief period of time in order to allow coalescing of that change report with others that occur for the same managed resource within the coalescing time window. This optimization will be performed while maintaining the following characteristics:

- Grouping of notifications may be done for property change, status change, operating system message, and log entry notifications, but will not be done for other notification types.
- Notifications will be buffered for a maximum of 1 second.
- The grouping of change reports will not obscure a client's ability to correctly observe the temporal ordering of the individual changes made to a particular object or between objects based on the messages sent to a session. That is, notification messages will always be generated so that the ordering of the messages as determined by their session sequence numbers reflects the temporal order in which the changes were actually made. All of the changes reported to a session in a message with a lower session sequence number will have occurred before any of the changes reported in a message with a higher session sequence number. Further, the ordering of change reports within a particular message reflects the order in which they occurred to that object as well.
- Coalescing of multiple changes into a single notification message will occur for at most a single pending notification message (thus, of a single type) at a time. If a need arises to report a change of a different type than is currently pending (for example, a need to report a status change when there is currently a set of pending property change reports), coalescing will end for that pending message and it will be posted to notification topics as appropriate. This is necessary in order to maintain the API client's ability to correctly observe temporal ordering.
- The grouping of change reports into notification messages occurs independently for each session, so that one session may receive a different distribution of change reports across notification messages than another session.

The degree to which message grouping occurs or not depends on the timing of changes and possibly other considerations and thus is to be strictly considered an optimization and not guaranteed behavior. Client applications should infer no particular semantic significance to change reports being delivered in a single message vs. a sequence of messages.

Status change notification

A Status Change notification is emitted by the API to report changes to the **status** property of a managed object.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided for this type of notification:

Message property name	Description
notification-type	Contains the value "status-change" .

The body of a Status Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
change-reports	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
old-status	String	The old (previous) value of the status property for the object. The value of this field will be one of the possible enumeration values for the status property as defined for this class of object.
old-additional-status	String	The old (previous) value of the additional-status property for the object. The value of this field will be one of the possible enumeration values for the additional-status property as defined for this class of object.
new-status	String	The new (current) value of the status property for the object. The value of this field will be one of the possible enumeration values for the status property as defined for this class of object.
new-additional-status	String	The new (current) value of the additional-status property for the object. The value of this field will be one of the possible enumeration values for the additional-status property as defined for this class of object.
has-unacceptable-status	Boolean	The value of the has-unacceptable-status property of the object, based on its new status. If true, the object is now considered to have unacceptable status because its current status is not one of the configured acceptable status values for this object.

Property change notification

A Property Change notification is emitted by the API to report changes to the properties of a managed object where the data model description indicates that modification notification support (qualifier "pc") is available for that property.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	Contains the value "property-change" .
property-names	Blank-separated list of the names of the properties for which change reports are provided in the body of this notification message. The list always includes a leading and trailing blank so that a property name can be specified as a blank-delimited word in a message selector to avoid matching unintended properties that have the desired property name as a substring.

The body of a Property Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
change-reports	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
property-name	String	The name of the property (as specified in the object's data model) that has changed.
old-value	Based on model	If the property is not a container-type or write-only property, this field contains the old (previous) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete previous value. Rather, it provides an array of entries that have been removed from the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been removed, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null.
new-value	Based on model	If the property is not a container-type or write-only property, this field contains the new (current) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete new value. Rather, it provides an array of entries that have been added to the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been added, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null.

Inventory change notification

An Inventory Change notification is emitted by the API to report the addition or removal of a managed object to/from the current inventory of resources that are being managed by zManager. This occurs when managed resources are created or deleted, but also may occur in other situations, such as when zManager reestablishes its inventory of (already-existing) managed resources upon restart of the HMC.

For some kinds of managed objects, an Inventory Change notification is also emitted by the API to report the addition or removal of an element of a managed object. Such notifications do not occur for all elements, but rather only when specifically described in the documentation for a class of managed object.

Because an Inventory Change notification may be generated more than once for the same conceptual object, these notifications cannot be interpreted as designating a resource creation action.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	Contains the value "inventory-change" .
name	Not provided for this notification. Always an empty string.
action	The value "add" when the object has been added to the inventory, or "remove" when it is being removed.

The body of an inventory change notification is null.

Job completion notification

A Job Completion notification is emitted by the API to report that the processing of an operation that runs asynchronously to the client application has ended.

Asynchronous operations are those that complete with an HTTP status code of 202 (Accepted) when requested by the client. A Job Completion Notification message is sent to the API session that initiated the job when such an operation completes or is canceled, and provides to the client application the URI of the job that has completed or been canceled so the client application can use the Query Job Status operation to obtain results for the job.

Characteristic	Description
Destination	The per-session job notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	The value "job-completion" .
job-uri	The URI identifying the asynchronous job that has just completed execution or has been canceled.

The body of a job completion notification is null.

Log entry notification

A Log Entry notification is emitted by the API to report the addition of a log entry to its corresponding console log.

Characteristic	Description
Destination	The audit notification topic or security notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message property is provided to allow for message selection:

Message property name	Description
notification-type	Contains the value "log-entry" .

The body of a Log Entry notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
log-entries	Array of objects	An array of nested log-entry-info objects, the format of which is described in Table 320 on page 593 . The order in which these objects appear in this array reflects the temporal order in which the log entries were created.

Operating system message notification

An operating system message notification is emitted by the API to report new or refreshed operating system messages.

Characteristic	Description
Destination	One of the os-message-notification topics associated with the API session.

In addition to the common JMS message headers described above, the following additional message property is provided to allow for message selection:

Message property name	Description
notification-type	Contains the value "os-message" .

The body of an operating system message notification message is a JSON representation of an object that contains the following field and value:

Field name	Type	Description
os-messages	Array of objects	An array of nested os-message-info objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the messages were created.

Each nested os-message-info object has the following fields and values:

Field name	Type	Description
sequence-number	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message.
message-text	String	The text of the new or refreshed operating system message.
message-id	String	The message identifier of the operating system message.
timestamp	Timestamp	The timestamp represents the date and time when the operating system message was created. A value of -1 is returned if this information is not available from the corresponding operating system.
sound-alarm	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (true) or not (false).
is-priority	Boolean	Specifies whether the operating system message is a priority message (true) or not (false). A priority message indicates a critical condition that requires immediate attention.
is-held	Boolean	Specifies whether the operating system message is a held message (true) or not (false). A held message is one that requires a response.

Field name	Type	Description
prompt-text	String	Specifies the prompt text that is associated with this operating system message or null indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.
os-name	String (1-8)	Specifies the name of the operating system that generated this operating system message or null indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the partition in which the operating system is running.
is-refresh	Boolean	Specifies whether the message is a new (false) or a refresh message (true). When the user connects to an os-message-notification topic, operating system messages that already exist are sent as refresh messages, if desired by the user.

Chapter 5. Data model definitions

This chapter covers data model concepts and shared data model schema elements.

Data model concepts

zManager provides resource management and control functions for the various resources known to the HMC. In performing these functions, zManager establishes a separation between those aspects of resource management that are handled entirely by system firmware, and the other aspects for which customer or installation visibility, configuration and control is appropriate.

In order to specify the external aspects in a succinct way, the Web Services API is described in this document in terms of a conceptual data model that it offers for the resources that it manages. This data model is an information structuring technique that conceptually defines the kinds of resources that are managed by zManager and for each, the information that is available for and the operations that can be performed on resources of that kind. This data model is intended to provide the complete perspective that clients of the API can have regarding the logical resources of the system while insulating them from implementation details.

Objects in the data model

The manageable resources of the environment are represented in the management system as entities referred to as objects. Each distinct manageable resource is represented by a separate object instance, and the life cycle of an instance corresponds with the lifecycle of the manageable resource it represents. For example, for physical entities, such as a CPC, the object that represents it is created implicitly when the physical entity is attached to and configured to be part of the system. This object continues to exist so long as the CPC is managed by the HMC.

There are different kinds of manageable physical or logical resources in the system, and each kind manifests different observable characteristics. As a result, there are different classes of objects in the data model. Objects of the same class represent the same kind of resource and provide a defined set of *properties* that capture the attributes of that kind of resource that the Web Services API exposes.

Managed objects and element objects

The object classes defined in the data model fall into one of two main categories: managed object classes (or simply managed objects), and element object classes (element objects).

These two categories are very similar in that they are both, ultimately, unordered collections of named properties that capture the key attributes of a resource instance. The categories differ primarily in how prominently they are handled in the API: the way that instances of them are designated to perform operations on them, and the degree to which API facilities such as inventory and change notification can be offered for objects in that category.

Managed objects are the first-class entities in the data model and the API. They represent the primary manageable resources of the system, such as CPCs, logical partitions, adapters and partitions. These kinds of objects typically appear prominently in the main displays of the HMC user interface.

Instances of managed objects are registered and indexed in the zManager managed object registry, and thus can be directly referenced by URIs that form a relatively "flat" namespace. The URI of a managed object designates its object instance based on its class and a unique, durable, UUID-based identifier called an object ID. For example, the URI of a CPC is of the form `/api/cpcs/{cpc-object-id}` where the identifier at the end of the URI is globally unique. Inventory change, property change, and status change notifications can be generated for managed objects.

In comparison, element objects represent the secondary or more-detailed aspects of the system. Examples include the Load Activation Profiles of a CPC, or the NICs of a Partition. These kinds of entities

do not generally appear in the main displays of the HMC user interface, but rather are displayed only within particular management tasks offered by the UI.

Instances of element objects are not directly registered in the zManager object registry, but rather are associated with or “attached to” some containing or related managed object instance. As a result, access to these elements is indirect, through the containing managed object. The URIs that designate element objects are hierarchical in nature, with the leftmost part of the URI identifying the managed object to which the element is attached. For example, the URI for a HBA of a Partition is of the form `/api/partitions/{partition-object-id}/hbas/{hba-id}` in which the `{hba-id}` at the end is only necessarily unique within the context of the related partition. Inventory, property and status change notifications are not always directly provided for element objects. In such cases, when changes to elements are reported, those changes are done via property change notifications emitted for the associated managed object.

Properties in the data model

Objects in the management system contain, fundamentally, unordered collections of name/value pairs called properties that capture the key characteristics of the manageable resources they represent. The defined set of named properties that are maintained for a particular kind of resource constitutes the specification of the data model class for that kind of resource.

As a result, in the chapters that follow, the description of the management interfaces for a class of resource begins with a data model section that specifies the properties that are exposed by the API for that kind of resource.

Each property has a name, a data type, and a semantic description in prose.

The property name is the programmatic identifier of the property. This identifier is used within requests and responses to indicate that a field represents a particular property of the data model. It is the “name” part of the name/value pair that is the property.

The property data type indicates the kind of information that can be represented by the property, just as a variable's data type indicates the kind of information that can be stored in a variable. The data type provides information on the nature of the “value” part of the name/value pair that is the property.

Property characteristics

Properties are classified as being either writable or read-only from the perspective of an API client application.

Writable properties are ones that can have their values read by `Get <class> Properties` or similar operations and can also have their values directly changed by `Update <class> Properties` operations. Properties that are classified as write-only can have their values directly changed by `Update <class> Properties` operations, but cannot have their values read by using `Get <class> Properties` or similar operations. Properties that are classified as read-only can have their values read by using `Get <class> Properties` or similar operations, but cannot have their values changed directly.

Although properties that are classified as read-only cannot have their values changed directly, their values may nonetheless be affected by other operations supported by a class of object. For example, a class of object might include an **is-enabled** property that is classified as read-only because the enabled state of the resource cannot be affected by a simple `Update <class> Properties` operation on that **is-enabled** property. However, this object might also define a `Change State` or `Enable` operation that can perform this enabling, and as a side effect will alter the value of the **is-enabled** property.

In addition to the read-only vs. writable classification, properties defined for a managed object also can differ in whether changes to them result in property change or status change notifications being emitted for the managed object or not. For properties that have property or status change support, these notifications are emitted asynchronously by the API any time the value of the property changes, whether that change was made via this API, the HMC UI, or implicitly by the system. Changes to the values of properties for which change notification support is not provided do not result in such notifications.

Most objects have properties that are primarily configuration data. However, for some objects, certain of those configuration-related properties may at times also have a transient, runtime counterpart property in

effect whose value can be different than the preserved configuration or base value. That transient, runtime counterpart property is known as an "effective" property and is identified as such in the object's data model by the (e) qualifier. The name of an effective property is formed by prepending "effective-" to the name of its corresponding base property. Whether an object's effective properties are applicable at any point in time is determined by the state of the object or the state of the system or related objects and is described in that object class' section of this document. The object's **effective-properties-apply** property indicates whether effective properties are currently applicable. If **effective-properties-apply** is **false**, the value of an effective property is the same as its corresponding base property and the effective property may not be altered by an Update <class> Properties operation.

In some cases, an object in the management system may have properties whose values are unwieldy to provide or expensive to obtain and further may not be of general interest in typical API client use cases. To allow the handling of such properties to be optimized they are represented by a special kind of property termed a pseudo property. A pseudo property is conceptually one of the characteristics of a class of object and is thus documented in the data model for the object (with a (p) qualifier). However, the name and value of a pseudo property is not included in the response to a Get <class> Properties operation or in the inventory service data for that class of object. Instead, class-specific operations are provided in order to obtain the current value of the property when needed by an API client. As for normal properties, property change notifications may be generated for changes to a pseudo property if indicated in the data model.

In the tables of properties that appear within this document, the characteristics of properties are indicated by qualifier annotations in parenthesis following the property name. The qualifiers have the following meanings:

Qualifier notation	Description
(w)	The property is a writable property. Any property that lacks this qualifier and the (wo) qualifier is considered read-only and thus is not directly modifiable.
(wo)	The property is a write-only property. Any property that lacks this qualifier and the (w) qualifier is considered read-only and thus is not directly modifiable.
(ro)	Although this property is writable when present in other managed object classes, it is read only in this class. This qualifier is only used when a managed object class specializes the definition of a base managed object property and overrides the writable characteristic of the base definition.
(p)	The property is a pseudo property. Its current value is omitted from the response to a Get <class> Properties operation or inventory service data for the object, but can be obtained by a class-specific operation.
(pc)	Change to this property's value will result in Property Change notifications.
(c)	The property is a container-type property for which deltas (changes) are reported in Property Change notifications rather than complete old and new values.
(sc)	Change to this property will result in Status Change notifications.
(mg)	This property represents a performance or utilization metric of the object that is included in a metric group available via the Metrics Service of this API. The value of this property may change very frequently and, therefore, property change notifications are not emitted for changes to this property. Client applications interested in obtaining metric information frequently should obtain this information through use of the Metrics Service of this API.
(e)	The property is an effective property.

Shared data model schema elements

The data-model schema fragments in this section define groups of properties that are used in common ways in specifying the data models for the managed object classes defined in the API.

The description of the data model for a specific object class specifies the shared schema elements it is incorporating within the data model section of that description, if any. It will also include a description of the specializations that apply to that class's use of the shared schema, such as additional constraints on properties, class-specific values for properties, etc.

Base managed object properties schema

This data-model fragment contains the basic properties that are present in the representation of many of the managed object types that represent manageable resources.

Name	Qualifier	Type	Description
object-uri	—	String (1-255)	The canonical URI path that designates this managed object instance and serves as the primary reference and retrieval key for this instance. The URI path is formed based on a unique and permanently-assigned object ID (see the object-id property in the next row of this table), and as result, an object's URI path will not change as a result of changes to properties of the object. Further, this canonical URI path is independent of the containment hierarchy and thus will not change if this object instance is moved within the hierarchy.
object-id	—	String (36)	The object identifier for the managed object instance. This value is unique in space and time, and is permanently associated with this instance while it is managed by this HMC. (If the instance is removed from this HMC and later managed by another HMC, it will have a new and different object identifier when managed by that other HMC.) It is generated by zManager and assigned when the managed resource is created or first discovered, and is immutable thereafter. As example, a managed object's object ID will not change as a result of changes to display name, changes in the location of this resource in the containment hierarchy, or across restarts of the HMC.
parent	—	String (1-255)	The canonical URI path of the managed object that is conceptually the parent of this object in the containment hierarchy. This property is null for objects that do not have a parent.
class	—	String (1-32)	The class of resource represented by this managed object. Each distinct class of resource has a different type name, while all instances of the same type share the same type name. The specific value used for a class of object is specified in the data model section for that object type. Example: "logical-partition" .
name	(w)(pc)	String (1-64)	The current display name of the managed object as defaulted or specified for the object. This is the simple name of this object, not qualified by containment hierarchy. This name must consist only of alphanumeric characters and the following special characters: period (.), hyphen (-), at sign (@), underscore (_), and space. It must not begin or end with a space. Some resource types do not support the setting of a user-assigned display name. For such objects, this property is not settable, and instead always provides a name assigned by the HMC or SE.

Name	Qualifier	Type	Description
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about this managed resource. This information is retained for the resource and may be shown as part of the object's details on the user interface, but is otherwise not generally used by zManager. This property may be null.
is-locked	(pc)	Boolean	The object is locked and thus disruptive actions or tasks cannot be performed on it.
effective-properties-apply	—	Boolean	The object is currently in a state in which effective properties are applicable. As this property is only meaningful for object classes whose data model includes effective properties, it is only included for those object classes.

Operational status properties

Many (but not all) classes of managed objects support the concept of operational status. That is they maintain information about the current functional state (Not Communicating, Not Operating, etc.) of the managed resource and whether that current functional state is considered acceptable (not alert causing) or not. If a class of object supports the operational status concept, it provides the standard properties defined in the following table (referred to as the operational status properties) in addition to those defined earlier in this section.

Unless stated to the contrary, any object class data model that includes the base managed object properties schema should be understood to also provide these operational status properties as well. For object classes for which that is not the case, the data model description will specifically point out that operational status and thus these operational-status-related properties are not provided for that object.

The operational status properties are as follows:

Name	Qualifier	Type	Description
status	(sc)	String Enum	The current operational status of the managed resource. The possible status values vary by managed object class and are specified in the description of each managed object class that provides this property.
additional-status	(sc)	String Enum	A qualifier to the status property used by selected object classes to provide finer grained operational status tracking.
acceptable-status	(w)(pc)	Array of String Enum	The set of operational status values that the managed resource can be in and be considered to be in an acceptable (not alert causing) state.
has-unacceptable-status	(sc)	Boolean	If true, the current operational status of the managed resource is not one of the acceptable statuses for this resource.

Chapter 6. Firmware features

This chapter describes firmware features.

Firmware feature concepts

Starting with HMC version 2.14.0 and API version 2.23, features can be enabled for specific objects. To indicate this, an available-features-list property is introduced to the objects that are affected. These features may be enabled by default from a specific HMC or SE version onwards or enabled by using standard feature enablement mechanisms.

These features affect certain API operations as documented here. API clients should query the enablement of these features on a system and choose the operations accordingly.

The enablement of a feature might mean that the API clients have to use new API operations or new properties in existing API operations. It might also indicate that some of the existing API operations and properties will not be supported.

If an API operation is not supported when a feature is enabled, invoking the API operation on an object where the feature is enabled would result in a standard status code 409 (Conflict) with a standard reason code 12.

If an API operation is supported only when a feature is enabled, invoking the API operation on an object where the feature is disabled would result in a standard status code 409 (Conflict) with a standard reason code 13.

If an API operation targeting an object whose existence is controlled by a feature is invalid given the enablement of the feature, a standard status code 404 (Not Found) with standard reason code 1 (for managed objects) or 5 (for element objects) is returned.

The following section describes the features that are currently available.

dpm-storage-management

This feature is applicable for the CPC and Partition objects. The `Get CPC Properties` and `Get Partition Properties` operations can be used to query if the feature is enabled or disabled.

When this feature is enabled, management of FICON storage is available. FCP and FICON virtual storage resources are defined in Storage Groups which are then attached to Partitions. A Partition that has this feature enabled has no HBAs visible to an API client or on the UI.

When this feature is disabled, FICON storage is not available and FCP virtual storage resources (HBAs) are attached directly to Partitions.

The following API operations are affected and will return status code 409 (Conflict) with reason code 12 when they are invoked on an object on which the **"dpm-storage-management"** feature is enabled.

- Create HBA
- Export WWPN List
- Dump Partition

The following API operations are affected and will return status code 404 (Not Found) with reason code 5 when they are invoked on an HBA object on which the **"dpm-storage-management"** feature is enabled.

- Delete HBA
- Update HBA Properties
- Get HBA Properties
- Reassign Storage Adapter Port

There are new API operations introduced with the feature. They are defined in the following sections under [Chapter 10, “Dynamic Partition Manager \(DPM\),” on page 147](#):

- [“Storage Site operations summary” on page 153](#)
- [“Storage Fabric operations summary” on page 153](#)
- [“Storage Switch operations summary” on page 153](#)
- [“Storage Subsystem operations summary” on page 154](#)
- [“Storage Control Unit operations summary” on page 155](#)
- [“Storage Group operations summary” on page 155](#)

In addition, the following new API operations are introduced with the feature and are defined in [“Partition operations summary” on page 149](#):

- Start Dump Program
- Attach Storage Group to Partition
- Detach Storage Group from Partition

These operations will return status code 409 (Conflict) with reason code 13 when they are invoked on an object on which the **"dpm-storage-management"** feature is disabled.

Part 2. General services

Topics in this part describe the general services available for the Web Services API.

Topics covered in this part are:

- [Chapter 7, “General API services,” on page 77](#)
- [Chapter 8, “Inventory and metrics services,” on page 111](#)
- [Chapter 9, “Metric groups,” on page 129](#)

Chapter 7. General API services

This chapter describes the services that are provided by the Web Services API for creating and deleting API sessions and performing other general functions.

General API services operations summary

Table 33. General API services: operations summary	
Operation name	HTTP method and URI path
“Query API Version” on page 78	GET /api/version
“Logon” on page 80	POST /api/sessions
“Establish Shared Secret Key” on page 86	POST /api/sessions/operation/establish-shared-secret-key
“Provide Requested MFA Information” on page 88	POST /api/sessions/operations/provide-more-mfa-information
“Change Logon Password” on page 91	POST /api/sessions/operations/change-logon-password
“Logoff” on page 93	DELETE /api/sessions/this-session
“Get Notification Topics” on page 94	GET /api/sessions/operations/get-notification-topics
“Submit Requests” on page 96	POST /api/services/aggregation/submit
“Query Job Status” on page 103	GET /api/jobs/{job-id}
“Delete Completed Job Status” on page 106	DELETE /api/jobs/{job-id}
“Cancel Job” on page 108	POST /api/jobs/{job-id}/operations/cancel

Table 34. General API services: URI variables	
Variable	Description
{job-id}	The identifier of an asynchronous job associated with this user, as returned in the response of the operation that initiated the job.

Session management services

Almost all operations of the Web Services API are requested and carried out in the context of an API session that is used for determining the client's authority to access managed resources and perform requested operations. It is also used to scope the delivery of asynchronous notifications and manage WebSocket instances. An API session is an HMC concept that is independent of any layers on top of

network-related considerations such as a TCP/IP socket connection. As a result, a single API session may span multiple TCP/IP socket connect/disconnect sequences from the same client.

Sessions are created upon request from a client by using the Logon operation, and may be explicitly terminated by a client using the Logoff operation. Sessions may also be terminated by the HMC due to inactivity when no requests are made using the session over a certain period of time. (The default session timeout is 6 hours, but it is configurable on a per-user basis.) However, termination of a session due to inactivity will not occur as long as a client application uses the API's notification facility to maintain a JMS subscription to one or more of the session's JMS notification topics. The existence of such a subscription is considered by the HMC to indicate that a client is still using the session and thus it is not terminated even if no requests are made using it. The existence of an open WebSocket associated with the API session will also prevent it from being considered inactive.

Sessions are identified by clients using a **session-id**, which is a string of up to 64 characters in length that is returned to the client in the results from a successful Logon operation. This string is generated in a cryptographically-secure manner. A **session-id** string is a form of authentication credentials for a user equivalent in power to a user's user ID and password. Because of this, a **session-id** should be transmitted only within SSL connections.

In order to indicate that subsequent requests are to be performed in the context of a designated session, the client supplies the appropriate **session-id** to the HMC in each such subsequent request. This is done by supplying the **session-id** as the value of the **X-API-Session** HTTP header which is an application-specific header defined by and recognized by the HMC.

The Logon and Query API Version operations are the only two operations in the Web Services API that can be performed without an API session so requests for these operations do not need to provide the **X-API-Session** HTTP header. All other operations are valid only in the context of an API session and thus requests for all other operations must supply an **X-API-Session** header with a valid **session-id** in order to be successfully executed.

Query API Version

The Query API Version operation returns information about the level of Web Services API supported by the HMC.

HTTP method and URI

```
GET /api/version
```

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
api-major-version	Integer	The major-version part of the API version in effect for this session
api-minor-version	Integer	The minor-version part of the API version in effect for this session
hmc-version	String (5-8)	The version number of the HMC firmware. This is a string of the form <i>v.r.m</i> , where each of <i>v</i> , <i>r</i> and <i>m</i> can be one or two digits. Example: "2.11.1".
hmc-name	String (1-16)	The name assigned to the HMC
hmc-time	Timestamp	The current time, according to the HMC.
classification-text	String (0-50)	The HMC's classification text. A 0-length string or null is returned if the classification is not set. If the classification text is more than 50 characters, only the first 50 are returned.

Field name	Type	Description
vendor	String Enum	The vendor that supplied the HMC. Non-IBM vendors are identified in an abstract fashion rather than by company name. The valid values are: • "ibm" - IBM • "a" - vendor a • "b" - vendor b • "c" - vendor c • "d" - vendor d • "e" - vendor e • "f" - vendor f • "g" - vendor g • "h" - vendor h • "i" - vendor i

Description

This operation returns name and version information for the HMC and the HMC API.

This operation can be requested without an API session being open, i.e. no **X-API-Session** header, and **session-id** is required on input.

For more information about the function included in each API version, see [“Summary of API version updates”](#) on page 7.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 78.

Under normal conditions, no error response codes are returned by this request. (HTTP Status code 500 could possibly result if internal HMC errors occur.)

Example HTTP interaction

```
GET /api/version HTTP/1.1
```

Figure 1. Query API Version: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 07 Aug 2018 20:02:57 GMT
content-type: application/json; charset=UTF-8
content-length: 167
{
  "api-major-version":2,
  "api-minor-version":20,
  "classification-text":"Top Secret",
  "hmc-name":"HMCTR117",
  "hmc-time":1504814577975,
  "hmc-version":"2.14.0",
  "vendor":"ibm"
}

```

Figure 2. Query API Version: Response

Logon

The Logon operation establishes an API session with the Web Services API.

HTTP method and URI

POST /api/sessions

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
userid	String	Required	The name of the HMC user to be associated with the new API session. This name may be of arbitrary length, i.e. the HMC does not have a defined maximum length.
password	String	Required	The password used to authenticate the HMC user identified by the userid field. The required length and valid characters are determined by the password policy in effect for the user ID.
new-password	String	Optional	A new password to be established for the user defined by the userid field. The required length and valid characters are determined by the password policy in effect for the user ID.
multi-factor-authentication-code	String (1-12)	User-dependent	The current multi-factor authentication code (time-based one-time password) used to authenticate the HMC user identified by the userid field. This field is required for HMC users that are configured to use multi-factor authentication and must be omitted for all other HMC users.
client-tag	String Enum	Optional	A tag string supplied by the API client program that issued the Logon request. Valid values are: • "mobile" - the client asserts that it is the mobile app. This value is intended for use only by the Mobile HMC app. The last logon time for each user of this client tag is displayed in the HMC Mobile Settings task.

Field name	Type	Rqd/Opt	Description
rsa-securid-passcode	String (1-64)	User-dependent	The current RSA SecurID passcode used to authenticate the HMC user identified by the userid field. This field is required for certain HMC users and is ignored for all others.

The largest request body accepted by this operation is 512 bytes. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
api-session	String (1-64)	The session-id of the newly created session. The client must specify this value in the X-API-Session header of all subsequent requests that are to be performed in the session.
notification-topic	String (1-128)	The name of the JMS topic the HMC will use to send object-related notification messages to this session.
job-notification-topic	String (1-128)	The name of the JMS topic the HMC will use to send job-related notification messages to this session.
api-major-version	Integer	The major-version part of the API version in effect for this session
api-minor-version	Integer	The minor-version part of the API version in effect for this session
password-expires	Integer	The time interval, in days, until the user's current password expires. A value of 0 indicates that the password will expire within the next 24 hours. A value of -1 indicates that the HMC does not enforce password expiration for this user, however, if this user is authenticated with an external authentication mechanism (e.g. LDAP) such expiration might be enforced by that mechanism.
shared-secret-key	String (32)	The proposed shared secret key for the user identified in the request body. This field is only included if the user is required to establish a shared secret key; in that case, the Logon operation completes with HTTP status code 201 (Created).
session-credential	String (32)	The session-specific authentication credential for this session. This token can be used to connect to the API message broker on behalf of the HMC user associated with this session. See “Connecting to the API message broker” on page 57 for more information.

Description

This operation opens a new API session with the Web Services API. Authentication is performed as part of this process.

The characteristics and permissions of an HMC user are specified in an HMC User or User Template definition. The user name provided in the **userid** field of the request body is used to select a corresponding User or User Template based on the name. If such a User or User Template is found, the client's authority to operate as this HMC user is authenticated by validating the password provided in the **password** field using the authentication method specified in the User or User Template.

If the HMC user is configured for multi-factor authentication, additional authentication processing is required. If the HMC user currently has an established shared secret key, the user's current multi-factor authentication code provided in the **multi-factor-authentication-code** field is validated. If the HMC user does not currently have an established shared secret key, HTTP status code 201 (Created) is returned.

If the authentication described above is successful, a new API session is created and the session-id for the new session is provided in the **api-session** field in the response from this operation. This same value is also provided by an **X-API-Session** HTTP header field in the response. If all required authentication is successful, the newly created API session is fully authenticated. If HTTP status code 201 (Created) is to be returned, a partially-authenticated API session is created. In this case an Establish Shared Secret Key operation, using the key supplied in the **shared-secret-key** field in the response body, must be issued to establish a shared secret key and complete the API logon sequence, thereby converting the partially-authenticated session into a fully-authenticated session. The operations available to a partially-authenticated session are limited to the Establish Shared Secret Key and Logoff operations.

If the request specifies an **X-API-Session** HTTP header field on input (indicating that this operation be performed under some designated session), the logon request fails and status code 400 (Bad Request) is returned.

If an HMC User or User Template corresponding to the user ID field does not exist, or if the password or multi-factor authentication code validation fails, the logon request fails and status code 403 (Forbidden) is returned. There is no reason code to distinguish these reasons for the failure. If the User or User Template is marked as disabled or the associated password has expired, or if the User or User Template is not configured to allow use of the API, the logon request also fails with status code 403 (Forbidden) and a reason code identifying the specific cause.

If all required user authentication is successful and the request body contains the optional **new-password** field, the password associated with the user is changed to the specified new value as part of the Logon operation. If the new password does not meet the requirements of the password policy in effect for this user or if the password is not changeable because it is managed by an external authentication mechanism, the request fails with status code 400 (Bad Request) and a reason code indicating the cause of the failure.

As part of establishing the new API session, names are assigned for the JMS topics that will be used by the HMC to send object-related and job-related notification messages to this session and the names of these topics are provided in fields of the response body. The name of the topic used for object-related notifications is provided in the **notification-topic** field of the response, and the name of the topic used for job-related notifications is provided in the **job-notification-topic** field.

Authorization requirements

This operation has the following authorization requirement:

- The HMC User Profile or User Template selected by the **userid** field must be configured to allow use of the Web Services API.

HTTP status and reason codes

On success, HTTP status code 200 (OK) or 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 81.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 35. Logon: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	12	The request specified an X-API-Session header, which is interpreted as an attempt to unnecessarily logon again when already logged on.
	13	The maximum number of logged in user sessions for this user ID has been reached; no more are allowed.
	43	The password for this user cannot be changed, for example because it is managed in an external authentication mechanism such as LDAP.
	44	The new password does not conform to the requirements of the password policy in effect for this user.
	45	The user's password has expired and no new-password field was specified. This reason code is only applicable for users who are not required to use MFA.
	47	The user is required to include a multi-factor authentication code during logon but the request body did not include the multi-factor-authentication-code field.
	48	The user is not required to include a multi-factor authentication code but the multi-factor-authentication-code field was included in the request body.
	49	The specified multi-factor authentication code is correct, but it has already been used. A code may only be used once. Wait until a new code is available and try again.
	53	Password changes cannot be requested when the user is required to use MFA.
	54	The user is required to include an RSA SecurID passcode during logon but the request body did not include the rsa-securid-passcode field.
403 (Forbidden)	0	User authentication failed.
	40	The user is disabled.
	41	The user is not authorized to use the HMC Web Services interface.

Table 35. Logon: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	50	The MFA Server requires more information in order to authenticate the API user. Details are provided in a JSON object in the response body. See the MFA server documentation for a description of that JSON object and the format of the JSON object required to contain the additional information. That additional information must be provided via the Provide Requested MFA Information operation. The error-details field of the response body contains an mfa-info-request object containing additional details, found in Table 36 on page 84.
	51	The user's password has expired. A new password must be set via the Change Logon Password operation. The error-details field of the response body contains a change-password-info object containing additional information, described in Table 37 on page 84. This code is only applicable for users who are required to use MFA.
	52	The MFA policy assigned to the API user contains one or more unsupported MFA factors.
500 (Server Error)	40	Unexpected error returned by the MFA server.
503 (Service Unavailable)	40	The request could not be processed because the HMC is unable to contact the MFA server.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

On completion where the HTTP status code is 409 (Conflict) with reason code 50, the standard error response body contains an error-details field that contains a description of the additional information requested by the MFA server. The value of the **error-details** field is a nested mfa-info-request object with the following fields:

Table 36. mfa-info-request nested object properties

Field name	Type	Description
resume-id	String (4-128)	A string which identifies the authentication request to which this response applies.
api-session	String (1-64)	The session-id of the partially-authenticated API session to which this response applies.
information-request	Object	A JSON object describing the additional information requested by the MFA server. See the MFA server documentation for details.

On completion where the HTTP status code is 409 (Conflict) with reason code 51, the standard error response body contains an error-details field that contains information for use on the required **Change Logon Password** operation. The value of the **error-details** field is a nested change-password-info object with the following field:

Table 37. change-password-info nested object properties

Field name	Type	Description
api-session	String (1-64)	The session-id of the partially-authenticated API session to which this response applies.

Usage notes

- The Logon operation checks for and prevents requests that specify an **X-API-Session** header on input in order to detect client applications that unnecessarily log on again when already logged on. It is valid to have multiple sessions, but in order to more explicitly indicate that this is desired, the client application needs to request each logon without referencing any existing session.
- Some of the information returned by this operation is also present in the response body of a successful Get Notification Topics request. Specifically, the information contained in the **notification-topic** and **job-notification-topic** fields is also included in the Get Notification Topics response. That operation identifies all JMS topics available to the API user, possibly including topics other than those identified in the Logon response.
- The multi-factor authentication code is a time-based one-time password as defined in RFC 6238, *TOTP: Time-Based One-Time Password Algorithm*, May 2011, (available at: <https://tools.ietf.org/html/rfc6238> from the Internet Engineering Task Force. The TOTP algorithm uses a shared secret key and the current time of day to calculate the TOTP for the current 30-second interval.
- When using multi-factor authentication the user must establish a shared secret key. This key is sensitive security information much like a password and is to be known only by the user and the HMC. This key can be established via the Establish Shared Secret Key operation or by logging on to the HMC via the local GUI interface or a remote web browser. The HMC presents this key to the user only once (during the first logon after being required to use multi-factor-authentication or having their shared secret key be invalidated by an administrator), and it is the user's responsibility to have it available for use during subsequent API and GUI logons, which will require the user's current multi-factor authentication code.
- The sequence of operations to establish a shared secret key via the APIs is as follows:
 - Issue a Logon operation with a valid logon password.
 - The Logon completes with HTTP status code 201 (Created) and returns an API session ID for a partially-authenticated session and a proposed shared secret key.
 - Issue an Establish Shared Secret Key operation with the API session ID and calculated multi-factor authentication code.
 - The Establish Shared Secret Key operation completes with HTTP status code 204 (No Content), the partially-authenticated session is converted into a fully-authenticated session and the proposed key is now the user's officially established shared secret key.
- When using multi-factor authentication it is important for the API client's time of day clock to be reasonably in sync with the HMC's clock, because the current time of day is used when calculating the user's current multi-factor authentication code. To determine if the clocks are reasonably in sync, the Query API Version operation may be used to obtain the HMC's current time, which can then be compared to that of the API client.
- If an Establish Shared Secret Key operation is required to complete a logon sequence that includes a **client-tag** field, the last mobile app logon time for the API user is not updated until the Establish Shared Secret Key operation completes successfully.

Example HTTP interaction

```
POST /api/sessions HTTP/1.1
content-type: application/json
content-length: 58
{
  "password": "12345678",
  "userid": "APIUSER"
}
```

Figure 3. Logon: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Wed, 02 Aug 2017 18:41:27 GMT
x-api-session: 4hy7c4nogldz4b59ajegzb1dulec641ziyv6uf73zs43205edv
content-type: application/json;charset=UTF-8
content-length: 281
{
  "api-major-version": 20,
  "api-minor-version": 2,
  "api-session": "4hy7c4nogldz4b59ajegzb1dulec641ziyv6uf73zs43205edv",
  "job-notification-topic": "APIUSER.229job",
  "notification-topic": "APIUSER.229",
  "password-expires": 29,
  "session-credential": "un8bu462g37aw9j0o8plntontz3szt35jh4b1qe2toxt6fkh14"
}

```

Figure 4. Logon: Response

Establish Shared Secret Key

The Establish Shared Secret Key operation completes the authentication of a partially-authenticated API session and establishes the user's multi-factor authentication shared secret key.

HTTP method and URI

POST /api/sessions/operations/establish-shared-secret-key

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
multi-factor-authentication-code	String (1-12)	Required	The current multi-factor authentication code (time-based one-time password) to be used to authenticate the HMC user associated with the partially-authenticated API session specified on the X-API-Session request header.

Description

This operation establishes a shared secret key for a user that is configured for multi-factor authentication. It completes the authentication of a partially-authenticated API session created by a previous Logon operation that completed with HTTP status code 201 (Created). That API session's ID must be specified on the **X-API-Session** request header. The request body must contain the current multi-factor authentication code calculated using the proposed shared secret key returned in the response body of the aforementioned Logon operation.

If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the multi-factor authentication code is not correct for the proposed shared secret key and current time of day, HTTP status code 403 (Forbidden) is returned. If the requirement for the user to establish a shared secret key no longer exists, HTTP status code 409 (Conflict) is returned with a reason code that indicates what has changed.

If the operation does not complete with status code 204 (No Content) or status code 409 (Conflict) with reason code 51, the partially-authenticated session is destroyed.

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of the partially-authenticated API session and the current multi-factor authentication code for that session.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	46	The X-API-Session request header does not identify a partially-authenticated session.
403 (Forbidden)	0	The multi-factor authentication code is not valid for the API session identified by the X-API-Session request header.
409 (Conflict)	12	The user associated with the API session identified by the X-API-Session request header already has an established shared secret key. That key could have been established by another API session or a GUI logon.
	13	The user associated with the API session identified by the X-API-Session request header is no longer configured for multi-factor authentication.
	51	The user's password has expired. A new password must be set via the Change Logon Password operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage notes

See the usage notes for the Logon operation for more information on using multi-factor authentication with the APIs.

Example HTTP interaction

```
POST /api/sessions/operations/establish-shared-secret-key HTTP/1.1
x-api-session: 5ql67thiw2og8ysixzljv8pwwmb4exfp85h9lu23a2irjxaq0w
content-type: application/json
content-length: 46
{
  "multi-factor-authentication-code": "314159"
}
```

Figure 5. Establish Shared Secret Key: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 16 Dec 2016 21:21:09 GMT

<No response body>
```

Figure 6. Establish Shared Secret Key: Response

Provide Requested MFA Information

The Provide Requested MFA Information operation satisfies a request from an MFA server for more user authentication information.

HTTP method and URI

POST /api/sessions/operations/provide-more-mfa-information

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
resume-id	String (4-128)	Required	A string which identifies the authentication request to which this operation applies.
requested-information	Object	Required	A JSON object containing the requested information in the format required by the MFA server.

Response body contents

On successful completion, the response body contains a JSON object with the same format as the response body for a Logon operation that completed with HTTP status code 200 (OK). See the Logon “Response body contents” on page 81.

Description

This operation supplies the additional user authentication information requested by an MFA server during a previous authentication attempt. For example, the API user may be required to change their MFA PIN or provide their next MFA token code. The previous attempt could have been via a Logon operation or a Provide Requested MFA Information operation.

If the provided information is successfully validated, the MFA server may require further information. In that case, HTTP status code 409 (Conflict) with reason code 50 is returned, and the response body describes the additional required information. The **error-details** field of the response body contains an mfa-info-request object described in [Table 36 on page 84](#).

If the provided information is successfully validated and the MFA server requires no further information, this operation completes the MFA authentication of a partially-authenticated API session created by a previous Logon operation that completed with HTTP status code 409 (Conflict) with reason code 50. That partially-authenticated API session's ID must be specified on the **X-API-Session** request header. If the user's logon password has expired, HTTP status code 409 (Conflict) with reason code 51 is returned, and the password must be changed via the Change Logon Password operation to complete the logon sequence.

The request body is validated against the schema described in “Request body contents” on page 88. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body must contain the **resume-id** returned in the response from a Logon or prior Provide Requested MFA Information operation as well as the information

specifically requested by that response. If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the MFA server did not authenticate the credentials for any reason, HTTP status code 403 (Forbidden) is returned and additional error details may be available.

If the operation does not complete with status code 200 (OK) or status code 409 (Conflict) with reason code 50 or 51, the partially-authenticated session is destroyed.

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of the partially-authenticated API session and the **resume-id** from a prior request for that API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, the response body is provided as described in [“Response body contents”](#) on page 88.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 38. Provide Requested MFA Information: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	40	The resume-id does not identify the currently outstanding request for additional information for the specified API session.
	46	The X-API-Session request header does not identify a partially-authenticated session that requires more MFA information.
403 (Forbidden)	0	The additional information provided in the request body is invalid.
	40	The MFA server failed to authenticate the credentials or requires the user to re-authenticate. The error-details field of the response body contains an mfa-requested-information-failures object. The mfa-requested-information-failures object is described in the next table.
409 (Conflict)	50	The MFA Server requires more information in order to authenticate the API user. Details are provided in a JSON object in the response body. See the MFA server documentation for a description of that JSON object and the format of the JSON object required to contain the additional information. That additional information must be provided via the Provide Requested MFA Information operation.
	51	The user's password has expired. The Change Logon Password operation must be issued to change the password and continue the logon sequence, possibly completing it.
500 (Server Error)	40	Unexpected error returned by the MFA server.
503 (Service Unavailable)	40	The request could not be processed because the HMC is unable to contact the specified MFA server.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Table 39. mfa-requested-information-failures object

Field name	Type	Description
failing-factors	Array of mfa-requested-information-failing-factor objects	The array of objects describing the failures, one per failing factor, as returned by the MFA server.

Table 40. mfa-requested-information-failing-factor object

Field name	Type	Description
name	String	The name/key of the failing factor
reason	String	The reason code of the failure, in the form of xx.yy, as returned by the MFA server.
message	String	The message describing the failure, as returned by the MFA server. May be an empty string.

Example HTTP interaction

```
POST /api/sessions/operations/provide-more-mfa-information HTTP/1.1
x-api-session: 12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3
content-type: application/json
content-length: 214
```

```
{
  "resume-id": "TgJI8gbkFYveL+DXToSM79gwI5Dx+Vs",
  "requested-information": {
    "resumeID": "TgJI8gbkFYveL+DXToSM79gwI5Dx+Vs",
    "apiVersion": 2,
    "factors": [
      {
        "factorName": "AZFSIDP1",
        "credentialObject": {
          "passCode": "12345678"
        }
      }
    ]
  }
}
```

Figure 7. Provide Requested MFA Information: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 01 Nov 2019 16:33:15 GMT
x-api-session: 12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3
content-type: application/json
content-length: 301
{
  "api-major-version": 3,
  "api-minor-version": 2,
  "api-session": "12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3",
  "job-notification-topic": "apiuser1.13job",
  "notification-topic": "apiuser1.13",
  "password-expires": 28,
  "session-credential": "3q2dbw5xbrzgjcnn1r13shuv3k00vcy1gjhqtf2fogpor6aa"
}
```

Figure 8. Provide Requested MFA Information: Response

Usage notes

See the usage notes for the Logon operation for more information on using multi-factor authentication with the APIs.

Change Logon Password

The Change Logon Password operation changes the logon password during a logon sequence for an MFA-enabled user.

HTTP method and URI

POST /api/sessions/operations/change-logon-password

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
current-password	String	Required	The password used to authenticate the HMC user associated with the partially-authenticated API session identified by the X-API-Session HTTP request header.
new-password	String	Required	A new password to be established for the user. The required length and valid characters are determined by the password policy in effect for the user ID.

Response body contents

On successful completion, the response body contains a JSON object with the same format as the response body for a Logon operation that completed with HTTP status code 200 (OK). See the Logon [“Response body contents”](#) on page 81.

Description

This operation changes the password used to authenticate the user associated with a partially-authenticated API session when that user's password has expired. The API session is identified by the **X-API-Session** HTTP request header. If the new password does not meet the requirements of the password policy in effect for this user or if the password is not changeable because it is managed by an external authentication mechanism, the request fails with status code 400 (Bad Request) and a reason code indicating the cause of the failure.

Upon successful completion the user's password is changed and this operation completes the authentication of a partially-authenticated API session created by a previous Logon operation that completed with HTTP status code 201 (Created) or HTTP status code 409 (Conflict) with reason code 50 or 51. That partially-authenticated API session's ID must be specified on the **X-API-Session** request header.

The request body is validated against the schema described in [“Request body contents”](#) on page 91. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the specified current password is not correct, HTTP status code 403 (Forbidden) is returned.

If the operation does not complete with status code 200 (OK), the partially-authenticated session identified by the **X-API-Session** HTTP request header is destroyed.

Authorization requirements

The request must contain the session ID of a partially-authenticated API session for a user whose logon password has expired and the current password for the user associated with that API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, the response body is provided as described in [“Response body contents”](#) on page 91.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 41. Change Logon Password: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	44	The new password does not conform to the requirements of the password policy in effect for this user.
	46	The X-API-Session request header does not identify a partially-authenticated session that requires a logon password change.
403 (Forbidden)	0	User authentication failed.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/sessions/operations/change-logon-password HTTP/1.1
x-api-session: 4uyp0d2o7q0495p4ktmw9mt4ef4w2owsrqjjgch7rbhds17u73
content-type: application/json
content-length: 60
{
  "current-password":"12345678",
  "new-password":"87654321"
}
```

Figure 9. Change Logon Password: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Sep 2019 19:51:25 GMT
x-api-session: 18l8cbysndoabvzdy1gq6oscd77hjmsyy4j4ezvf0tptf5vp
content-type: application/json
content-length: 301
{
  "api-major-version":3,
  "api-minor-version":2,
  "api-session":"18l8cbysndoabvzdy1gq6oscd77hjmsyy4j4ezvf0tptf5vp",
  "job-notification-topic":"apiuser1.12job",
  "notification-topic":"apiuser1.12",
  "password-expires":12,
  "session-credential":"2q2dbw5xbrzgjcnn1r13shuv3k00vcy1gjhtfg2fogpor6ii"
}
```

Figure 10. Change Logon Password: Response

Usage notes

See the usage notes for the Logon operation for more information on using multi-factor authentication with the APIs.

Logoff

The Logoff operation closes an API session with the Web Services API.

HTTP method and URI

```
DELETE /api/sessions/this-session
```

Description

This operation closes an API session with the Web Services API.

The session to be closed is indicated by the **session-id** in the **X-API-Session** header of the request. If the **session-id** designates an open session, the API session is closed and status code 204 (No Content) is returned. Closing of the API session includes closing/deleting any Metrics Service retrieval contexts or JMS notification topics associated with the session. However, asynchronous actions initiated by the session continue to run.

Once a session is closed, its **session-id** is no longer valid for use in subsequent Web Services API requests. Attempts to do so will result in the same errors as any other attempt to use a session-requiring operation without providing a valid **session-id**.

Authorization requirements

This operation has the following authorization requirement:

- No explicit authorization is required, however the client application must possess and present a valid **session-id** of the session to be closed.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/sessions/this-session HTTP/1.1
x-api-session: zkspmapxgtcasy5uixmtwuudqe8ha6fy0006bzm xsm2bd8yo
```

Figure 11. Logoff: Request

```
204 No Content
date: Wed, 20 Jul 2011 18:33:56 GMT
x-request-id: Sx32 Rx0
server: zSeries management console API web server / 1.0
cache-control: no-cache
```

Figure 12. Logoff: Response

Get Notification Topics

The Get Notification Topics operation returns a structure that describes the JMS notification topics associated with the API session. These topics allow the user to receive various types of asynchronous notifications from the HMC.

HTTP method and URI

GET /api/sessions/operations/get-notification-topics

Response body contents

On successful completion, the response body is a JSON object with the following field:

Field name	Type	Description
topics	Array of objects	Array of nested topic-info objects as described in the next table

Each nested topic-info object contains the following fields:

Table 42. topic-info object		
Field name	Type	Description
topic-type	String Enum	The type of notification topic, which provides an indication of the type of data found on the topic. Except for os-message-notification, a given value will only be represented at most once within a single response. One of the following values: • "object-notification" - The object notification topic. This topic is consistent with the information returned in the notification-topic field in the response body of a successful Logon request. It is used by the HMC to send object-related notifications to this session. • "job-notification" - The job notification topic. This topic is consistent with the information returned in the job-notification-topic field in the response body of a successful Logon request. It is used by the HMC to send job-related notifications to this session. • "audit-notification" - the audit notification topic. This topic is used by the HMC to send audit-related events to this session. • "security-notification" - the security notification topic. This topic is used by the HMC to send security-related events to this session. • "os-message-notification" - an operating system message notification topic. Topics of this type are used by the HMC to send notifications that pertain to new or refreshed messages generated by the operating system running in a partition. More than one of these might exist for this session. Additional fields specific to this topic type are described later in this table.

Table 42. topic-info object (continued)		
Field name	Type	Description
topic-name	String (1-128)	The name of the notification topic. API users can connect using this name to receive notifications for the topic.
object-uri	String/ URI	When the topic-type is " os-message-notification ", this field is the canonical URI path of the partition object for which this topic exists. This field does not exist for the other topic types.
include-refresh-messages	Boolean	When the topic-type is " os-message-notification ", this field indicates whether refresh operating system messages will be sent on this topic. A value of true indicates that refresh messages will be sent. A value of false indicates that no refresh messages will be sent. This field does not exist for the other topic types.

Description

This operation returns a list of all JMS topics to which the API user is authorized to connect. As there exists at least one JMS topic available to any authenticated user, the returned JSON array will never be empty.

Authorization Requirement

This operation has the following authorization requirement:

- No explicit authorization is required; however, the response to this request is limited to the topics to which the user is authorized to connect.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 94](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/sessions/operations/get-notification-topics HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4qymqa8ktzjep7dbyr1l0k
```

Figure 13. Get Notification Topics: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json;charset=UTF-8
{ "topics" :
  [
    { "topic-type": "object-notification", "topic-name": "mikeuser.1" },
    { "topic-type": "job-notification", "topic-name": "mikeuser.1job" },
    { "topic-type": "audit-notification", "topic-name": "mikeuser.1aud" },
    { "topic-type": "security-notification", "topic-name": "mikeuser.1sec" },
    { "topic-type": "os-message-notification",
      "topic-name": "mikeuser.1osmsg.cpc1.lpar1",
      "object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
      "include-refresh-messages": true },
    { "topic-type": "os-message-notification",
      "topic-name": "mikeuser.1osmsg.cpc2.par7",
      "object-uri": "/api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd",
      "include-refresh-messages": false }
  ]
}

```

Figure 14. Get Notification Topics: Response

Usage notes

Some of the information returned by this operation is also present in the response body of a successful Logon request. This operation is intended to provide a superset and will contain all JMS topics available to the API user including the two topics indicated in the Logon response.

Request aggregation services

Request aggregation services allow what would otherwise be multiple API requests to be submitted as a single request, with their multiple results likewise returned in a single response.

Submit Requests

The Submit Requests operation carries out multiple Web Services API requests as specified in its request body, and returns response information in a single API response.

HTTP method and URI

POST /api/services/aggregation/submit

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
requests	Array of api-request objects	Required	Array of api-request objects, each representing one Web Services API operation for the service to run.

Field name	Type	Rqd/Opt	Description
req-headers	Array of api-header objects	Optional	Array of api-header objects, each defining an HTTP header to add or modify in the header list for every api-request in the requests array. The headers associated with the <code>Submit Requests</code> operation are the basis for the headers used when carrying out each api-request. Any api-header objects specified here are processed in the order provided to modify those headers, and the resulting header set is used as the basis for every api-request. If a same-named header already appears in the base set of headers, the value supplied in the api-header object will override that existing value. If a same-named header does not already appear, then the header will be added with the specified value. An api-header object with only a name field and no value field will cause that header to be omitted. Note that an HTTP request header with an empty value can be set by supplying the empty string ("") for the api-header value field. Supplying null for the value field is not permitted. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC". The request header names Content-Type and Accept may not be specified. Only operations which expect JSON (if they have a request body), and generate JSON (if they have a response body) are supported. Therefore, the Content-Type and Accept headers associated with <code>Submit Requests</code> suffice. The request header name Content-Length may not be specified. The <code>Submit Requests</code> operation handles any needed calculation of Content-Length for the contained requests. Note that further header modification can be specified in each api-request. Such an individual request modification overrides any modification for same-named headers specified here.

Field name	Type	Rqd/Opt	Description
resp-headers	Array of String	Optional	HTTP header names. For each api-request, if the corresponding HTTP response contains any of these header names, those headers will be returned in the headers field of the corresponding api-response object in the Submit Requests response body. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC". Values may repeat in this array, and repeated values do not cause the headers field of the api-response objects to contain duplicates. However if a header name appears multiple times in an api-request's response, it will appear multiple times with each respective value in the headers field of the api-response. Note that, in general, response headers which are generated by the HMC API HTTP server itself, such as the Server response header, should not be expected to be seen for an individual api-response. That is because each individual api-request is not routed through the HTTP server but instead is driven as part of the Submit Requests HMC API HTTP server request. An exception to this guidance for server-generated response headers is the Date response header. If Date is requested in resp-headers it will be generated for each api-request by Submit Requests itself so that it may be returned.
threads	Integer (1-10)	Optional	The maximum number of threads the HMC may use to run the operations in the requests array. The default value is 1. A threads value of 1 specifies single-threaded execution. All requests will be issued on the HMC on a single thread in the same order they are specified in the requests array. A request will not be started until the previous request has completed. Single-threaded execution guarantees that the order of api-response objects in the response will match the order of the corresponding api-request objects in the requests array. A threads value greater than 1 specifies multi-threaded execution. The requests will be carried out on the HMC on multiple threads, up to the number specified. There will be no guarantee of the sequence of execution and no guarantee that any given request will be complete before another request is begun. The order of api-response objects in the response body will likewise not be predictable.

The api-request nested object contains the following fields:

Field name	Type	Rqd/Opt	Description
method	String	Required	The request method of the desired operation. For example, "GET", "POST", or "DELETE" (case is not significant).
uri	String/ URI	Required	The URI of the desired operation, including query parameters, if any.
body	Object	Optional	The JSON request body, if any, of the desired operation.
req-headers	Array of api- header objects	Optional	Array of api-header objects, each defining an HTTP header to add or modify in the header list used for this api-request only. The req-headers specified generally in the Submit Requests request body, if any, are applied first in the manner described for that field. Then these request-specific req-headers are applied to that modified base set in the same manner for this api-request only. The same guidance and restrictions given for the general req-headers field apply to this request-specific req-headers field.
resp-headers	Array of String	Optional	HTTP header names. For this api-request, if the corresponding HTTP response contains any of these header names, that header will be returned in the headers field of the corresponding api-response object in the Submit Requests response body. This array supplements any array specified generally in the Submit Requests request body.
id	String (1-64)	Optional	If an ID is provided, it will be returned in the id field of the corresponding api-response object in the Submit Requests response. Clients may use this ID to correlate api-request and api-response objects.

The api-header nested object contains the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	An HTTP header name. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC".
value	String	Optional	An HTTP header value.

The largest request body accepted by this operation is 256KB. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.

Response body contents

On successful completion, the response body is a JSON array of api-response objects as described below. The response body is sent using HTTP chunked transfer encoding.

If **threads** is not specified in the request body, or is specified with a value of 1, then the order of the api-response objects in the response corresponds to the order of the api-request objects in the **requests** array of the request body. So the first api-response object represents the response to the first api-request object, and so on.

If **threads** is specified in the request body with a value greater than 1, then no order of the api-response objects is guaranteed.

Regardless of the value of **threads**, the optional **id** field in the api-request and api-response objects may be used by the client to correlate requests and responses.

Each api-response object contains the following fields:

Field name	Type	Description
body	Object	The response body of the corresponding api-request. This field is always included but if there is no response body for the corresponding api-request then the value is null .
status	Integer	The HTTP status code for the corresponding api-request.
headers	Array of api-header objects	HTTP response headers for the corresponding api-request. This array contains only headers requested using the resp-headers field (global or specific to the api-request) in the Submit Requests request body. If a requested response header did not actually appear in the api-request's set of response headers, then it is omitted from this array. If no response headers fit this criterion then this field is omitted.
id	String	The id of the corresponding api-request object. If no id is provided in the api-request, this field is omitted.

Description

The Submit Requests request body specifies the Web Services API operations that are to be run in aggregation as an array of api-request objects. An api-request specification includes the operation **method** (for example, "GET", "POST", or "DELETE"), the operation URI, and the operation request body (if any). It may also include HTTP request header overrides in a **req-headers** field (in the event that those provided with Submit Requests itself are not satisfactory) and HTTP response header names in a **resp-headers** field (in the event that the client needs to know the value of a response header for an individual operation). Note that while you may specify these HTTP header behaviors for an individual operation in an api-request, you may also alternatively or additionally specify these HTTP header behaviors for every operation being aggregated. So, for example, if the client needs to receive the value of the **Location** response header for every operation, the **resp-headers** field does not need to be specified repeatedly in each api-request; the **resp-headers** field may be specified once as a peer to the **requests** array and it will apply to every api-request.

The Submit Requests response is an array of JSON objects, one for each API request specified in the request body. These JSON objects include the individual request's response body (if any) and status code. Optionally, they also include HTTP response headers associated with the individual request and an id that can be used to correlate that JSON object with the matching submitted request.

Note that only operations which expect JSON (if they have a request body), and generate JSON (if they have a response body) are supported for aggregation by Submit Requests. So operations that have request bodies that are not JSON (such as Mount ISO Image) and operations that have response bodies that are not JSON (such as Get Metrics) are not supported.

Single-threaded execution may be specified with a value of 1 for the **threads** field in the request body, or by omitting the **threads** field from the request body. Under single-threaded execution, Submit

Requests processes the requests sequentially in the order provided. It ensures each request has completed before beginning the next request (although some requests may initiate asynchronous jobs if they are so-specified). In other words, all of the requests are processed by the HMC on a single thread. Therefore the JSON objects in the response are in the same order as the requests in the request body.

Multi-threaded execution may be specified with a value greater than 1 for the **threads** field in the request body. Under multi-threaded execution, `Submit Requests` may process the requests in any order and their processing may overlap on up to the number of threads specified. Therefore, multi-threaded execution should not be specified if the sequence of execution of the requests is significant, or if there is some other reason why two or more of the requests should not be run simultaneously. Since the response order is not guaranteed, the client must use the **id** fields of `api-request` and `api-response`, or some other field unique to each expected response, to correlate the requests and responses.

Under both single-threaded and multi-threaded execution, requests are processed without regard to the result of any other request specified in the `Submit Requests` request body. Therefore, exercise caution aggregating requests with inter-dependencies. Even under single-threaded execution where it is guaranteed that request N has completed before request N+1 is initiated, it is *not* guaranteed that request N has completed *successfully* before request N+1 is initiated.

The response is returned using HTTP chunked transfer encoding, which allows the HMC to stream the response to each request as it completes, rather than having to accumulate the responses to all aggregated requests before returning data.

Authorization requirements

This operation has no authorization requirements of its own and will not directly report any authorization error.

The operation in each `api-request` is subject to that operation's authorization requirements as if it had been issued individually from the same API session. Any authorization problem is reflected in the status and/or body fields of the corresponding `api-response` object with the same content that would have been in the HTTP status code and response body had the request been issued directly and individually.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 99](#).

It is important to recognize that a successful status code from `Submit Requests` does not indicate that every (or, indeed, any) submitted request was successful. Each `api-response` object must be examined to determine the outcome of each submitted `api-request`.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 43. <i>Submit Requests: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The request body attempted to use a req-headers field to modify one of the following request headers : Content-Type , Accept , or Content-Length .

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/services/aggregation/submit HTTP/1.1
x-api-Session: 59ott72ftvzxkj7fz7g4jl48gh6csq7a7q3l98lm6gicq8wrl5
content-type: application/json
content-length: 257
{
  "requests":
  [
    {
      "id": "100000",
      "uri": "/api/console/users?name=stevef",
      "method": "get",
      "method": "DELETE",
      "uri": "/api/services/metrics/context/no-such-animal",
      "id": "200000"
    }
  ],
  "threads": 1,
  "req-headers": [{"name": "new-hdr", "value": "new-value"}]
}
```

Figure 15. Submit Requests: Request

```
200 OK
transfer-encoding: chunked
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 06 Apr 2018 15:08:04 GMT
content-type: application/json
[
  {
    "body": {
      "users": [
        {
          "name": "stevef",
          "object-uri": "/api/users/a8ce7ac6-39ab-11e8-b4d7-00106f0d5d80",
          "type": "standard"
        }
      ]
    },
    "status": 200,
    "id": "100000"
  },
  {
    "body": {
      "http-status": 404,
      "reason": 1,
      "request-uri": "/api/services/metrics/context/no-such-animal",
      "request-method": "DELETE",
      "message": "Context specified is not found, id is not a number",
      "request-headers": {
        "new-hdr": "new-value",
        "content-length": 0,
        "host": "127.0.0.1:6794",
        "content-type": "application/json",
        "x-api-session": "59ott72ftvzxkj7fz7g4jl48gh6csq7a7q3l98lm6gicq8wrl5",
        "accept-encoding": "identity",
        "accept": "application/json"
      },
      "request-body": {
        "request-authenticated-as": "acsadmin"
      },
      "status": 404,
      "id": "200000"
    }
  }
]
```

Figure 16. Submit Requests: Response

Usage notes

Note that specifying a higher number for **threads** in the request body will not necessarily lead to better performance. One reason is that a larger number of threads may increase the contention for access to the connection which delivers completed results to the client. The relative performance of different numbers of threads is largely dependent on attributes of the individual requests, making it impossible to recommend an ideal number of threads for all applications.

For example, it has been observed that for some large collections of quick-running requests, a thread count of 2 consistently provided better overall performance than either higher thread counts or single-threaded operation. Longer running requests that use different resources may see different results. Other processing load will of course influence performance as well. This variation is why Submit Requests allows the client to tune the maximum number of threads used.

Note further that increasing the number of threads for Submit Requests may increase the possibility of one or more of the constituent requests reporting a 503 (Service Unavailable) error with reason code 3.

This error is reported when exceeding the number of concurrent API requests that are allowed, either for a given user or overall for an HMC. On its own, a single `Submit Requests` request will not exceed these limits, even if the maximum value of **threads** is specified. However, if there is other concurrent API activity, from the same client or other clients, then recognize that each thread running the constituent requests for `Submit Requests` counts individually toward the concurrent API request limits.

In summary, if multi-threaded behavior is desired, then the client application should specify the smallest number for **threads** that is observed to provide the desired performance most of the time. If in doubt, or if performance is not critical, defaulting to the single-threaded behavior is recommended.

Asynchronous job processing

Some of the operations that are provided in the Web Services API may take a significant amount of elapsed time to complete. In order to optimize the usage of HMC session resources and to allow the client application the opportunity to perform other processing, such long-running operations are structured to be executed asynchronously (rather than synchronously) from the perspective of the client application.

In a synchronous operation, the Web Services API does not respond to the client application's request until all of the processing associated with the request is complete (successfully or in error) and the API can provide a final result status for the operation. The client application thread is typically blocked (not running) during this time.

By contrast, in a function that operates asynchronously by starting a job, the Web Services API performs just the minimal front-end validation and set up work needed to accept the request to perform the indicated operation, and then quickly returns an HTTP 202 (Accepted) result to the client indicating that the operation request has been started but is not yet finished. Along with the HTTP 202 (Accepted) result, the client application is provided with a URI that represents the asynchronous job that is in progress. This URI is of the form `/api/jobs/{job-id}`.

At any point after receiving the HTTP 202 (Accepted) result, the client application can invoke the `Query Job Status` operation described in this section to determine if the job has ended or not. A job is considered ended if it runs to completion or is canceled. If the job has not yet ended, the `Query Job Status` request returns an indication that the job is still running or cancellation has been requested. If the job has ended, the `Query Job Status` request returns an indication of how the job ended along with the final status code, reason code and result data associated with the now-finished asynchronous processing. Once a job ends, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely.

Since the major reason an API operation is structured to be asynchronous is that it will take significant time to complete, very frequent polling for completion via calls to `Query Job Status` can lead to significant unproductive use of client application and HMC resources. In order to eliminate the need to poll at all, the Web Services API also provides asynchronous notifications of job completion or cancellation via its JMS notification capability. IBM recommends that client applications use this notification facility to determine when a job has ended rather than polling. See [“Job completion notification”](#) on page 64 for details on this notification mechanism.

If it is not practical for a client application to use asynchronous notification of job completion, the application should introduce elapsed-time delays between successive `Query Job Status` requests to poll the current job status in order to reduce unproductive use of resources.

Query Job Status

The `Query Job Status` operation returns the status associated with an asynchronous job.

HTTP method and URI

```
GET /api/jobs/{job-id}
```

In this request, the URI variable `{job-id}` is the identifier of an asynchronous job associated with the API user, as returned in the response of the operation that initiated the job.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
status	String Enum	An indication of the current disposition of the job. The possible values are as follows: • "running" - indicates that the job was found and it has not ended at the time of the query. • "cancel-pending" - indicates that the job was found and it has not ended but cancellation has been requested. • "canceled" - indicates that the job's normal course of execution was interrupted by a cancel request. The successful or unsuccessful completion of the job is indicated by the job-status-code and job-reason-code fields. • "complete" - indicates that the job was found and has completed running. The successful or error completion of the job is indicated by the job-status-code and job-reason-code fields.
job-status-code	Integer; Field provided only if status is "complete" or "canceled"	The job completion status code. This field is provided only if the status field is set to "complete" or "canceled". This field provides the major status code describing the success or failure completion of the asynchronous action represented by the job. It is expressed in terms of an HTTP status code (i.e. the HTTP status code that would have been returned for the operation had it been performed synchronously). The values provided here and their meaning depend on the particular action that is being performed asynchronously. The description of these values is provided as part of the description for the operation that initiated the asynchronous job.
job-reason-code	Integer; Field provided only if status is "complete" or "canceled"	The job completion reason code. This field is provided only if the status field is set to "complete" or "canceled" and only if the job-status-code field indicates an error completion (status code other than 2XX). When present, this field provides a more detailed reason code describing the success or failure completion of the asynchronous action represented by the job. It is expressed in terms of the API reason code as are returned in standard error response bodies provided by the API. The values provided here and their meaning depend on the particular action that is being performed asynchronously. The description of these values is provided as part of the description or the operation that initiated the asynchronous job.

Field name	Type	Description
job-results	Object; Field provided only if status is "complete" or "canceled"	A nested object that provides results for the job. This field is provided only if the status field is set to "complete" or "canceled" and the asynchronous operation is documented to provide job results. If the status field is set to some other value, or the asynchronous operation provides no result information (beyond the job status and reason codes) then this field is omitted. The structure of the nested object provided by this field and its meaning depends on the particular action that is being performed asynchronously. The description of this object's structure is provided as part of the description of the operation that initiated the asynchronous job.

Description

The Query Job Status operation returns the status associated with an asynchronous job.

If the job designated by the URI is still running, the operation sets the **status** field in the response body to **"running"** and provides no other information about the job. If cancellation has been requested for the job designated by the URI but the cancellation action has not yet caused the job to end, the operation sets the **status** field in the response body to **"cancel-pending"** and provides no other information about the job. The client application may repeat the query at a later time, but should avoid frequent polling since that can lead to unproductive use of client and HMC resources. In order to eliminate the need to poll at all, the client application can (and should) use the asynchronous notifications facility provided by the API to receive notification that the job has ended via a JMS-based message. See ["Job completion notification"](#) on page 64 for details on this notification mechanism.

If the job is complete, the operation sets the **status** field in the response body to **"complete"** and provides the other completion-related fields defined in the response body contents section above to report the results to the client application. If the job's normal execution sequence was interrupted by a cancel action, the operation sets the status field in the response body to **"canceled"** and provides the other related fields defined in the ["Response body contents"](#) on page 104 to report the results to the client application. Once a job ends, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely.

If the URI does not designate a job associated with the API user, HTTP status code 404 (Not Found) is returned to the client.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in ["Response body contents"](#) on page 104.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/jobs/86e44546-107f-11e1-bde0-0010184c8334 HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4rymq8ktzjep7dbyrl10k
```

Figure 17. Query Job Status: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Wed, 16 Nov 2011 18:19:35 GMT
content-type: application/json;charset=UTF-8
content-length: 63
{
  "job-reason-code": 0,
  "job-status-code": 200,
  "status": "complete"
}
```

Figure 18. Query Job Status: Response

Delete Completed Job Status

The Delete Completed Job Status operation deletes the job status and results associated with a job that has ended.

HTTP method and URI

```
DELETE /api/jobs/{job-id}
```

In this request, the URI variable *{job-id}* is the identifier of an asynchronous job associated with the API user, as provided by the operation that initiated the job.

Description

The Delete Completed Job Status operation deletes the job status and results associated with a job that has ended.

If the job designated by the request URI has completed or has been canceled, its ending status and results are deleted from the HMC and status code 204 (No Content) is returned to the client.

If the job has not yet ended (i.e. is still running, or cancellation has been requested but is still pending), the operation fails and HTTP status code 409 (Conflict) is returned to the client.

If the URI does not designate a job associated with the API user, or if the job's status has already been deleted (either explicitly, or due to expiration of the status retention interval), HTTP status code 404 (Not Found) is returned to the client.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.
409 (Conflict)	40	The URI designates an asynchronous job that has not ended.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage notes

- This operation is defined to operate only on jobs that have ended, i.e. have a **status** field with value **"complete"** or **"canceled"**. As a result, this operation cannot be used to cancel an in-progress asynchronous operation. See [“Cancel Job” on page 108](#) for information pertaining to job cancellation.
- Once an asynchronous job has ended, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely. At the expiration of the retention interval, job status is deleted as if the Delete Completed Job Status operation were called.

Example HTTP interaction

```
DELETE /api/jobs/86e44546-107f-11e1-bde0-0010184c8334 HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo14rymq8ktzjep7dbyrl10k
```

Figure 19. Delete Completed Job Status: Request

```
204 No Content
date: Wed, 16 Nov 2011 18:19:35 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 20. Delete Completed Job Status: Response

Cancel Job

The Cancel Job operation attempts to cancel an asynchronous job.

HTTP method and URI

```
POST /api/jobs/{job-id}/operations/cancel
```

In this request, the URI variable *{job-id}* is the identifier of an asynchronous job associated with the API user, as provided by the operation that initiated the job.

Description

The Cancel Job operation attempts to cancel the specified job. The specific nature of the asynchronous job and its current state of execution when the request is received can affect the success of the cancellation action.

Not all asynchronous jobs support job cancellation. If a particular type of job supports cancellation, the description of the operation that initiates that type of job will explicitly specify that cancellation is supported and may describe other cancellation characteristics as well. If the description of the operation that initiates that type of job does not specify that cancellation is supported, then cancellation of that type of job is not possible.

If the specified job exists, but is of a type that does not support cancellation, status code 404 (Not Found) is returned and the job is allowed to continue processing without interruption.

If the specified job exists and has not yet completed (that is, the value of its **status** property is **"running"**), the cancellation request is made pending for the job, the **status** of the job is changed to **"cancel-pending"**, and HTTP status code 202 (Accepted) is returned. The processing of the pending cancellation request occurs asynchronously to the completion of the Cancel Job operation.

If the specified job exists and supports cancellation but either already has a cancellation request pending or has already ended (that is, has a **status** property with values **"cancel-pending"**, **"complete"** or **"canceled"**), HTTP status code 409 (Conflict) is returned with a reason code that more specifically indicates the particular error condition.

Once a cancellation request is made pending, the HMC will take steps to interrupt the processing of the job in order to cause the processing to end as quickly as is possible considering the nature of the processing done by the job. The conditions under which a running job can be interrupted vary depending on the type of job. For some types of jobs the console may be able to interrupt the processing very quickly while for others the console may be able to do so only as processing crosses selected interruption points. Thus, acceptance of a cancellation request does not guarantee that the processing of the job will either be immediately or eventually interrupted. As a consequence, it is possible that a job may proceed to normal completion (end with a **status** of **"complete"**) even after a cancellation request was accepted for the job.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user.

HTTP status and reason codes

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.
	4	The URI designated an asynchronous job that can not be canceled.
409 (Conflict)	41	The URI designates an asynchronous job that has ended. It can not be canceled.
	42	The URI designates an asynchronous job that already has a cancellation request pending. A second cancel request is not allowed.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/jobs/86e44546-107f-11e1-bde0-0010184c8334/operations/cancel HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4rymq8ktzjep7dbyrl10k
```

Figure 21. Cancel Job: Request

```
202 Accepted
date: Wed, 10 Jun 2015 18:19:35 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache
<No response body>
```

Figure 22. Cancel Job: Response

Chapter 8. Inventory and metrics services

The functions described in this chapter are termed "services" because unlike the interfaces described in many of the other chapters of this document, the functions described here are service-oriented rather than object-oriented in nature. That is, the functions of these services operate across multiple instances of managed objects rather than being directed at particular managed object instances.

The Inventory Service provides an efficient mechanism for retrieving identify and configuration information about all of the manageable resource instances that are managed by zManager. It provides this information in bulk form via a single request, and thus is expected to be a much more efficient means of determining this information than walking the entire resource tree one object at a time. It is anticipated that this service supports the requirements of a "discovery" phase of a client application that is interested in configuration information about all resources managed by zManager.

The Metrics Service provides a mechanism to retrieve performance metric data for resources that are managed by zManager. This data is captured periodically and buffered on the HMC. The data may include snapshots of performance data at a moment in time, or accumulated performance data, or both, as appropriate. This service is designed to support client applications that provide monitoring function for zManager managed resources.

Inventory services operations summary

The following operation is provided by the Inventory service:

Table 44. Inventory service: operations summary	
Operation name	HTTP method and URI path
"Get Inventory" on page 112	POST /api/services/inventory

Metrics service operations summary

The following operations are provided by the Metrics service:

Table 45. Metrics service: operations summary	
Operation name	HTTP method and URI path
"Create Metrics Context" on page 121	POST /api/services/metrics/context
"Get Metrics" on page 124	GET /api/services/metrics/context/{metrics-context-id}
"Delete Metrics Context" on page 127	DELETE /api/services/metrics/context/{metrics-context-id}

Table 46. Metrics service: URI variables	
Variable	Description
{metrics-context-id}	Identifier of a metrics context object. Metrics contexts are associated with API sessions. Thus, this identifier is assigned by the metrics service so that it is unique within an API session and has a lifetime scoped to that session.

Inventory service

The Inventory Service is an API which allows the client application to fetch a list of managed resources and their properties.

This service is intended to support clients that need to determine the inventory and properties of all of the managed resources known to the HMC (or at least a large portion of those resources). Retrieving this information in bulk form using this service is expected to be much more efficient than walking the resource tree one object at a time using the object-oriented operations of the Web Services API.

The ability to filter the results to only certain classes of resources is provided.

A response to an inventory request is a series of JSON objects returned using HTTP chunked transfer encoding. These objects will be in a format specified in the corresponding resource class's inventory service data sections.

Resources returned are those to which the API client has object-level authorization.

Get Inventory

The `Get Inventory` operation fetches managed resources and associated properties.

HTTP method and URI

```
POST /api/services/inventory
```

Request body contents

The request body can include a specification of the classes of resources that should be returned. It contains the following field:

Field name	Type	Description
resources	Array of String Enum	An array of String values. Each element specifies a category or class of resource that should be returned. A category is simply a grouping of classes, so specifying a category is functionally equivalent to specifying all of its member classes. The request may include both categories and classes. Omitting the resources field, or providing an empty array, is equivalent to specifying an array listing all of the supported classes. Categories and associated class values: • Category: "dpm-resources" – Class: "adapter" – Class: "partition" – Class: "virtual-switch" – Class: "storage-site" – Class: "storage-fabric" – Class: "storage-switch" – Class: "storage-subsystem" – Class: "storage-control-unit" – Class: "storage-group" – Class: "storage-template" • Category: "core-resources" – Class: "cpc" – Class: "logical-partition" • Category: "console-resources" – Class: "console" – Class: "custom-group" – Class: "user" – Class: "user-role"

Response body contents

On successful completion, the response body is a JSON array of JSON objects sent using HTTP chunked transfer encoding. The order in which these objects are returned is unspecified.

The array element documents are of 2 types:

- For resources that were successfully inventoried, the document will be as specified in the corresponding resource's inventory service data.
- For resources that were found but not successfully fully inventoried (i.e. the Object URI can be determined but not the properties), an inventory error document will be returned. Note that, even if one or more of these inventory error documents is contained in the response, an HTTP status code of 200 (OK) is still returned. The fields in the inventory error document are:

Field name	Type	Description
uri	String/ URI	Canonical URI of the resource which could not be fully inventoried.
class	String	The class for these error documents is "inventory-error" .

Field name	Type	Description
inventory-error-code	Integer	A reason code for the inventory failure. Note that all of these reasons indicate success in locating a resource, but some sort of failure in gathering its properties during inventory collection. A subsequent call to get the properties for the URI in this error document may succeed. • 1: Resource not found on target. Although the resource's URI was located on the HMC, its properties were subsequently not located on the HMC or SE on which the property data for the managed object is to be gathered. • 2: Communication problem. Communication problems were experienced with the SE on which the property data for the managed object is to be gathered. • 3: Plugin load error. The code responsible for capturing the properties of a resource class experienced an unexpected problem loading. • 4: Unknown plugin error. The code responsible for capturing the properties of a resource returned an unrecognized error. • 5: Unexpected plugin error. The code responsible for capturing the properties of a resource returned an unexpected error. • 6: Timeout error. The code responsible for capturing the properties of a resource did not respond within the time allocated to it.
inventory-error-text	String	An error description for the inventory failure.
inventory-error-details	inventory-error-info Object	A nested inventory-error-info object that provides additional diagnostic information for unexpected inventory plugin errors. This field is provided if the inventory-error-code field is 5 (indicating unexpected plugin error). It is not provided for other inventory-error-code values. The format of the inventory-error-info object is defined in the next table.

The inventory-error-info object contains the following fields:

Field name	Type	Description
http-status	Integer	HTTP status code for the request.
request-uri	String	The URI that caused this error response.
reason	Integer	Numeric reason code providing more details as to the nature of the error) than is provided by the HTTP status code itself. This reason code is treated as a sub-code of the HTTP status code and thus must be used in conjunction with the HTTP status code to determine the error condition. Standard reason codes that apply across the entire API are described in “Common request validation reason codes” on page 50 . Additional operation-specific reason codes may also be documented in the description of the specific API operations.
message	String	Message describing the error. This message is not currently localized.
stack	String	Internal HMC diagnostic information for the error. This field is supplied only on selected 5xx HTTP status codes.

Field name	Type	Description
error-details	Object	A nested object that provides additional operation-specific error information. This field is provided by selected operations, and the format of the nested object is as described by that operation.

Description

The Get Inventory operation returns information on managed resources and associated properties.

A resource is included in the response if it matches any one of the list of resource classes in the request body. Specifying a category is equivalent to specifying its member classes. If a class is repeated on the request, either explicitly or effectively via categories, the operation will behave as if the class were only specified once. If no resources are specified in the request body, all resources are returned. The inclusion of a resource may cause objects of certain related classes to also be included in the response. See the resource's Inventory Service Data section for the information about which, if any, related classes will be included.

Furthermore, a resource is included in the response only if the API user has object-access permission for that resource. If an HMC is a manager of a resource but the API user does not have permission to it, that resource is simply omitted from the response. A success status code is still returned.

If the HMC does not manage any resources to which the user has access, or if no resources are found that match the request body specification, an empty response is returned with a 204 (No Content) status code.

In addition to objects for inventoried resources, the response may include objects for resources whose URIs could be determined, but whose properties could not, for some reason, be obtained. Rather than treat these resources as completely non-inventoried and omit them, the URI and an error reason are returned.

The order in which the objects are returned is unspecified.

The Get Inventory implementation may choose to limit the number of simultaneous in-process inventory requests. If such a limit is reached, further requests will return an HTTP 503 (Service Unavailable) error status code until previous requests complete and the number of in-process inventory requests falls back below the limit.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to any object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 113](#). If there are no resources to provide, HTTP status code 204 (No Content) is returned, along with an empty response body.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
503 (Service Unavailable)	200	The request could not be processed because of the number of currently pending inventory requests. The request can be reissued at a later time.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage notes

The `Get Inventory` results represent a snapshot of inventory results as viewed from the HMC. The actual inventory can change, even as the results are being streamed back to the API client. Therefore, if the client wishes to stay informed about changes to the inventory and not risk missing any inventory changes, it should use the API event mechanisms to subscribe to inventory-related events before even issuing a `Get Inventory` request.

The `Get Inventory` results do not reflect all properties at a single moment in time. During the overall inventory collection process multiple resource's states and other properties may change. Therefore, states (or other properties) among two or more resources that might normally be expected to match (or have some other expected relationship) at one moment in time may instead return apparently inconsistent results in the `Get Inventory` response.

Example HTTP interaction

The following example illustrates a typical response for a `Get Inventory` request for the **logical-partition** class of resources. Responses for other classes will differ significantly from this because the data differs on a class by class basis. The format of the data returned by the Inventory Service for each class of object is described in a section entitled "Inventory service data" within the documentation for that object class.

```
POST /api/services/inventory HTTP/1.1
x-api-session: 2hatu4672fai3jicdxpeamt173qpqkxglg8vyz0wfmnu2fx32
content-type: application/json
content-length: 36
{
  "resources": [
    "logical-partition"
  ]
}
```

Figure 23. Get Inventory: Request

```

200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 21:51:05 GMT
content-type: application/json; charset=UTF-8
[
  {
    "absolute-aap-capping": {
      "type": "none"
    },
    "absolute-cf-capping": {
      "type": "none"
    },
    "absolute-ifl-capping": {
      "type": "none"
    },
    "absolute-processing-capping": {
      "type": "none"
    },
    "absolute-ziip-capping": {
      "type": "none"
    },
    "acceptable-status": [
      "operating"
    ],
    "activation-mode": "not-set",
    "additional-status": "",
    "class": "logical-partition",
    "cluster-name": null,
    "current-aap-processing-weight": null,
    "current-aap-processing-weight-capped": null,
    "current-cf-processing-weight": null,
    "current-cf-processing-weight-capped": null,
    "current-ifl-processing-weight": null,
    "current-ifl-processing-weight-capped": null,
    "current-processing-weight": null,
    "current-processing-weight-capped": null,
    "current-ziip-processing-weight": null,
    "current-ziip-processing-weight-capped": null,
    "defined-capacity": null,
    "description": "LPAR Image",
    "group-profile-capacity": null,
    "group-profile-uri": null,
    "has-operating-system-messages": null,
    "has-unacceptable-status": true,
    "initial-aap-processing-weight": null,
    "initial-aap-processing-weight-capped": null,
    "initial-cf-processing-weight": null,
    "initial-cf-processing-weight-capped": null,
    "initial-ifl-processing-weight": null,
    "initial-ifl-processing-weight-capped": null,
    "initial-processing-weight": null,
    "initial-processing-weight-capped": null,
    "initial-ziip-processing-weight": null,
    "initial-ziip-processing-weight-capped": null,
    "is-locked": false,
    "last-used-activation-profile": "",
    "last-used-boot-record-logical-block-address": "C8",
    "last-used-disk-partition-id": 0,
    "last-used-load-address": "00007",
    "last-used-load-parameter": "",
    "last-used-logical-unit-number": "1000000000000",
    "last-used-operating-system-specific-load-parameters": "SYSG",
    "last-used-world-wide-port-name": "50017380EB0B0141",
    "maximum-aap-processing-weight": null,
    "maximum-cf-processing-weight": null,
    "maximum-ifl-processing-weight": null,
    "maximum-processing-weight": null,
    "maximum-ziip-processing-weight": null,
    "minimum-aap-processing-weight": null,
  }
]

```

Figure 24. Get Inventory: Response (Part 1)

```

    "minimum-cf-processing-weight":null,
    "minimum-ifl-processing-weight":null,
    "minimum-processing-weight":null,
    "minimum-ziip-processing-weight":null,
    "name":"VM137",
    "next-activation-profile-name":"VM137",
    "object-id":"d6641179-b8e0-3980-9d22-32cdd967c774",
    "object-uri":"/api/logical-partitions/d6641179-b8e0-3980-9d22-32cdd967c774",
    "os-ipl-token":null,
    "os-level":null,
    "os-name":null,
    "os-type":null,
    "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
    "partition-identifier":null,
    "partition-number":null,
    "program-status-word-information":null,
    "status":"not-activated",
    "storage-central-allocation":null,
    "storage-expanded-allocation":null,
    "sysplex-name":null,
    "workload-manager-enabled":null
  },
  {
    "absolute-aap-capping":{
      "type":"none"
    },
    "absolute-cf-capping":{
      "type":"none"
    },
    "absolute-ifl-capping":{
      "type":"none"
    },
    "absolute-processing-capping":{
      "type":"none"
    },
    "absolute-ziip-capping":{
      "type":"none"
    },
    "acceptable-status":[
      "operating"
    ],
    "activation-mode":"not-set",
    "additional-status":"",
    "class":"logical-partition",
    "cluster-name":null,
    "current-aap-processing-weight":null,
    "current-aap-processing-weight-capped":null,
    "current-cf-processing-weight":null,
    "current-cf-processing-weight-capped":null,
    "current-ifl-processing-weight":null,
    "current-ifl-processing-weight-capped":null,
    "current-processing-weight":null,
    "current-processing-weight-capped":null,
    "current-ziip-processing-weight":null,
    "current-ziip-processing-weight-capped":null,
    "defined-capacity":null,
    "description":"LPAR Image",
    "group-profile-capacity":null,
    "group-profile-uri":null,
    "has-operating-system-messages":null,
    "has-unacceptable-status":true,
  }
]

```

Figure 25. Get Inventory: Response (Part 2)

```

    "initial-aap-processing-weight":null,
    "initial-aap-processing-weight-capped":null,
    "initial-cf-processing-weight":null,
    "initial-cf-processing-weight-capped":null,
    "initial-ifl-processing-weight":null,
    "initial-ifl-processing-weight-capped":null,
    "initial-processing-weight":null,
    "initial-processing-weight-capped":null,
    "initial-ziip-processing-weight":null,
    "initial-ziip-processing-weight-capped":null,
    "is-locked":false,
    "last-used-activation-profile":"",
    "last-used-boot-record-logical-block-address":"0",
    "last-used-disk-partition-id":0,
    "last-used-load-address":"00000",
    "last-used-load-parameter":"",
    "last-used-logical-unit-number":"0",
    "last-used-operating-system-specific-load-parameters":"",
    "last-used-world-wide-port-name":"0",
    "maximum-aap-processing-weight":null,
    "maximum-cf-processing-weight":null,
    "maximum-ifl-processing-weight":null,
    "maximum-processing-weight":null,
    "maximum-ziip-processing-weight":null,
    "minimum-aap-processing-weight":null,
    "minimum-cf-processing-weight":null,
    "minimum-ifl-processing-weight":null,
    "minimum-processing-weight":null,
    "minimum-ziip-processing-weight":null,
    "name":"MOBILEAU",
    "next-activation-profile-name":"MOBILEAU",
    "object-id":"3fd82f1a-ef7e-3e41-9a74-a5ce63089561",
    "object-uri":"/api/logical-partitions/3fd82f1a-ef7e-3e41-9a74-a5ce63089561",
    "os-ipl-token":null,
    "os-level":null,
    "os-name":null,
    "os-type":null,
    "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
    "partition-identifier":null,
    "partition-number":null,
    "program-status-word-information":null,
    "status":"not-activated",
    "storage-central-allocation":null,
    "storage-expanded-allocation":null,
    "sysplex-name":null,
    "workload-manager-enabled":null
  },
  {
    "absolute-aap-capping":{
      "type":"none"
    },
    "absolute-cf-capping":{
      "type":"none"
    },
    "absolute-ifl-capping":{
      "type":"none"
    },
    "absolute-processing-capping":{
      "type":"none"
    },
    "absolute-ziip-capping":{
      "type":"none"
    },
    "acceptable-status":[
      "operating"
    ],
  },

```

Figure 26. Get Inventory: Response (Part 3)

```

    "activation-mode":"not-set",
    "additional-status": "",
    "class":"logical-partition",
    "cluster-name":null,
    "current-aap-processing-weight":null,
    "current-aap-processing-weight-capped":null,
    "current-cf-processing-weight":null,
    "current-cf-processing-weight-capped":null,
    "current-ifl-processing-weight":null,
    "current-ifl-processing-weight-capped":null,
    "current-processing-weight":null,
    "current-processing-weight-capped":null,
    "current-ziip-processing-weight":null,
    "current-ziip-processing-weight-capped":null,
    "defined-capacity":null,
    "description":"LPAR Image",
    "group-profile-capacity":null,
    "group-profile-uri":null,
    "has-operating-system-messages":null,
    "has-unacceptable-status":true,
    "initial-aap-processing-weight":null,
    "initial-aap-processing-weight-capped":null,
    "initial-cf-processing-weight":null,
    "initial-cf-processing-weight-capped":null,
    "initial-ifl-processing-weight":null,
    "initial-ifl-processing-weight-capped":null,
    "initial-processing-weight":null,
    "initial-processing-weight-capped":null,
    "initial-ziip-processing-weight":null,
    "initial-ziip-processing-weight-capped":null,
    "is-locked":false,
    "last-used-activation-profile": "",
    "last-used-boot-record-logical-block-address":"0",
    "last-used-disk-partition-id":0,
    "last-used-load-address":"00000",
    "last-used-load-parameter": "",
    "last-used-logical-unit-number":"0",
    "last-used-operating-system-specific-load-parameters": "",
    "last-used-world-wide-port-name":"0",
    "maximum-aap-processing-weight":null,
    "maximum-cf-processing-weight":null,
    "maximum-ifl-processing-weight":null,
    "maximum-processing-weight":null,
    "maximum-ziip-processing-weight":null,
    "minimum-aap-processing-weight":null,
    "minimum-cf-processing-weight":null,
    "minimum-ifl-processing-weight":null,
    "minimum-processing-weight":null,
    "minimum-ziip-processing-weight":null,
    "name":"T257",
    "next-activation-profile-name":"T257",
    "object-id":"130e0c22-97ec-3a37-a109-93e1e2a4f5df",
    "object-uri":"/api/logical-partitions/130e0c22-97ec-3a37-a109-93e1e2a4f5df",
    "os-ipl-token":null,
    "os-level":null,
    "os-name":null,
    "os-type":null,
    "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
    "partition-identifier":null,
    "partition-number":null,
    "program-status-word-information":null,
    "status":"not-activated",
    "storage-central-allocation":null,
    "storage-expanded-allocation":null,
    "sysplex-name":null,
    "workload-manager-enabled":null
  }
]

```

Figure 27. Get Inventory: Response (Part 4)

Metrics service

The zEnterprise® (or later) Central Processing Complexes (CPCs) and their associated system resources are instrumented at key points to collect performance and utilization data. The data is forwarded by the metric data providers to a buffer on the HMC where it is made available to consumers of this API.

The data collection instrumentation organizes and formalizes the data it collects into a series of named metric groups. The Metrics Service API allows specification of the metric groups the client wishes to query. The API returns some information about the format of the metrics that are being fetched. Specifically, a structure called a metrics context is associated with any metrics retrieval, and that structure includes metric group names, individual metric field names, and the associated individual metric data types.

The full metric group documentation, however, including descriptions of the data collected and the frequency of collection, can be found in [Chapter 9, “Metric groups,” on page 129](#).

Create Metrics Context

The `Create Metrics Context` operation creates a context under which metrics can be repeatedly retrieved. This context will be associated with the API session under which it was created.

HTTP method and URI

```
POST /api/services/metrics/context
```

Request body contents

A request body must be specified. It has the following fields:

Field name	Type	Description
anticipated-frequency-seconds	Integer	The number of seconds the client anticipates will elapse between <code>Get Metrics</code> calls against this context. The minimum accepted value is 15.
metric-groups	Array of Strings	Optional. Array of metric group names. If specified, then results from future <code>Get Metrics</code> requests associated with this context will be limited to only metrics with group names matching one of the specified values. If not specified, or if an empty array is specified, then results will not be limited with respect to metric group names.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
metrics-context-uri	String/ URI	Canonical URI path of the metrics context object created by this operation. This includes the metrics-context-id. E.g. <code>/api/services/metrics/context/1</code> , where <code>"1"</code> is the metrics-context-id.
metric-group-infos	Array of objects	Array of metric-group-info objects (described in the next table) that describe the data format for each metric group that may be returned by future GETs associated with this metric context.

Each nested metric-group-info object contains the following fields:

Field name	Type	Description
group-name	String	The name of the metric group for which we are providing descriptive information.
metric-infos	Array	Array of metric-info objects (described in the next table). These describe each metric for the group in the order that they will appear in future GETs associated with this context.

Each nested metric-info object contains the following fields:

Field name	Type	Description
metric-name	String	The name of the metric.
metric-type	String Enum	One of the following, describing the type of the metric: "boolean-metric", "byte-metric", "double-metric", "long-metric", "integer-metric", "short-metric", "string-metric" See the Get Metrics “Response body contents” on page 124 for further information on these metric types.

Description

This operation establishes a context for future Get Metrics operations that is valid for the current API session. Because of the high frequency of invocation and large volume of data expected, the metrics service interface has been structured to optimize the transmission of data on each Get Metrics request. Thus, rather than use a self-describing representation for the results returned by each Get Metrics, the metrics service instead provides the descriptive metadata as results from this Create Metrics Context operation. It then returns the metric data in a compact format each time Get Metrics is invoked.

At a high level, the Create Metrics Context response communicates two primary pieces of information back to the client. One is the metrics-context-uri, which includes the ID of the metrics context that must be referenced on future GETs to associate them with this context. The other is the metric-groups description data. That data provides the metric type and metric name information for each metric group whose metrics may be returned by this context. This may be useful to the client for determining how to parse future Get Metrics responses for this context, although the full documentation on metric group formats is found in [Chapter 9, “Metric groups,” on page 129](#).

The anticipated-frequency-seconds specification which is required on the request body tells the metrics service how frequently the client anticipates issuing Get Metrics requests against this context. The metrics service may take no action based on this frequency, but reserves the right to invalidate and delete the metrics context if 4 times the specified frequency passes without receipt of an associated Get Metrics operation.

Optional result filtering is provided by field metric-groups on the request body. If a non-empty metric-groups arrays is specified, then future Get Metrics operations associated with this context will return only groups with names listed there.

Additionally, if a metric-groups array of group names is specified on the Create Metrics Context request, then the response JSON document will include only matching metric-group-info fields. If one or more names in the metric-groups array does not represent a metric group registered on the HMC, then HTTP error status code 400 (Bad Request) will be returned and the context will not be established.

Although the POST response fully describes and guarantees the ordering of metric-infos within a metric group for that context, as a matter of policy the HMC will further guarantee that, for a given metric group, any additions of new metrics to the group will be to the end of the list for the group.

Authorization requirements

There are no authorization restrictions on creating a metrics context. However any future metric results returned by `Get Metrics` queries against that context will be restricted to managed objects accessible according to the permissions associated with the API session under which the metrics context was established.

Note that there is no indication via an HTTP status or reason code that future results may be restricted due to authorization restrictions. Rather, success is indicated and future `Get Metrics` responses behave just as if any restricted objects did not exist.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 121. The URI for the newly created context is also provided in the **Location** header of the response.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/services/metrics/context HTTP/1.1
x-api-session: 3dajvxld29sk9zi584isbiguj4rl4gfkrsuxjprnsymb44e4vd
content-type: application/json
content-length: 73
{
  "anticipated-frequency-seconds":30,
  "metric-groups":[
    "channel-usage"
  ]
}
```

Figure 28. Create Metrics Context: Request

```

201 Created
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
location: /api/services/metrics/context/1
cache-control: no-cache
date: Wed, 18 Sep 2019 22:45:15 GMT
content-type: application/json;charset=UTF-8
{
  "metric-group-infos":[
    {
      "group-name":"channel-usage",
      "metric-infos":[
        {
          "metric-name":"channel-name",
          "metric-type":"string-metric"
        },
        {
          "metric-name":"shared-channel",
          "metric-type":"boolean-metric"
        },
        {
          "metric-name":"logical-partition-name",
          "metric-type":"string-metric"
        },
        {
          "metric-name":"channel-usage",
          "metric-type":"integer-metric"
        }
      ]
    }
  ],
  "metrics-context-uri":"/api/services/metrics/context/1"
}

```

Figure 29. Create Metrics Context: Response

Get Metrics

The Get Metrics operation retrieves the current set of metrics associated with an established metrics context.

HTTP method and URI

```
GET /api/services/metrics/context/{metrics-context-id}
```

In this request, the URI variable *{metrics-context-id}* is the identifier of the metrics context object for which metrics are to be obtained.

Response body contents

On successful completion, the response body contains the set of metrics associated with the metrics context. The response is sent using HTTP chunked transfer encoding and UTF-8 character encoding. A MIME media type of **application/vnd.ibm-z-zmanager-metrics** is used and is specified in the **Content-Type** header on the response.

Because performance and scalability are a major concern for the metrics service, the response body is in a terse custom format, rather than being presented as a JSON object. The data type, name, and order information required to parse and interpret the response is provided in a previous Create Metrics Context response.

Data in this format will be delimited by newlines and commas.

Using a partial Extended Backus-Naur Form, where a comma (,) indicates concatenation and curly braces ({}) indicate 0 or more repetitions, we can express the format this way:

```

MetricsResponse = {MetricsGroup},NL
MetricsGroup = MetricsGroupName,NL,{ObjectValues},NL
MetricsGroupName = StringValue
NL = "\n"
ObjectValues = ObjectURI,NL,Timestamp,NL,ValueRows,NL
Timestamp = LongValue
ObjectURI = StringValue
ValueRows = ValueRow,{ValueRow}
ValueRow = Value,{",",Value},NL
Value = BooleanValue | ByteValue | DoubleValue | LongValue | IntegerValue | ShortValue |
StringValue

```

The MetricsGroupName is the name of the metrics group, as a StringValue as defined below.

The Timestamp is the time when the associated values were buffered (i.e. "cached") on the HMC. It is expressed as an "epoch" timestamp: a LongValue giving the milliseconds since January 1, 1970, 00:00:00 GMT (just as is expected, for example, by the constructor of a java.util.Date object).

The ObjectURI is the canonical URI of the object, as a StringValue as defined below.

NL is a single newline character (Unicode U+000A).

All the varieties of Value will be represented as strings according to the following rules and limits:

- BooleanValue
 - Either the string true or the string false.
- ByteValue
 - A string representation of a signed decimal integer in the range -128 to 127 (i.e. the range of a signed 8 bit integer).
- DoubleValue
 - A string representation of a 64 bit IEEE 754 floating point number in the range +/-4.9E-324 to +/-3.4028235E+38. Note that, although IEEE 754 provides for representations of positive or negative Infinity and NaN, such values are not allowed in the metric data feed and thus will not appear in a metrics service result. For results with a magnitude greater than or equal to 10⁻³ and less than 10⁷, the string representation will be a dotted decimal (e.g. 1.7, -32.467). For results with magnitudes outside that range, the string representation will be computerized scientific notation (e.g. -4.23E127).
- LongValue
 - A string representation of a signed decimal integer in the range -9223372036854775808 to 9223372036854775807 (i.e. the range of a signed 64 bit integer).
- IntegerValue
 - A string representation of a signed decimal integer in the range -2147483648 to 2147483647 (i.e. the range of a signed 32 bit integer).
- ShortValue
 - A string representation of a signed decimal integer in the range -32768 to 32767 (i.e. the range of a signed 16 bit integer).
- StringValue
 - A string starting with a double-quote, ending with a double-quote, and with any embedded double-quotes or backslashes escaped with a preceding backslash (i.e. escaped as \" and \\).

Description

On successful execution status code 200 (OK) is returned, with a response body as described above.

The URI path on the request must designate an existing metrics context for the current API session. If the URI designates an unrecognized context for the API session, then status code 404 (Not Found) is returned.

Note that under some circumstances the metrics service may delete a metrics context, requiring the client to establish a new context in order to resume metric retrievals. For example, the metrics service may choose to delete a given context if the time since the last associated `Get Metrics` request has exceeded 4 times the anticipated frequency specified when the context was created. In addition, the client may explicitly delete a metrics context with the `Delete Metrics Context` operation. If the URI designates a context that was once valid for the current API session, but no longer is, then status code 409 (Conflict) is returned.

Authorization requirements

Only metrics referencing managed objects accessible according to the permissions associated with the API session under which the `Get Metrics` is being issued will be returned. Note that there is no indication via an HTTP status or reason code that results may have been restricted due to authorization restrictions. Rather, success is indicated and the responses are just as if any restricted objects did not exist.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 124](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The metrics context ID in the URI (<i>{metrics-context-id}</i>) does not designate a metrics context for the associated API session.
409 (Conflict)	1	The metrics context ID in the URI (<i>{metrics-context-id}</i>) designates a metrics context for the associated API session that is no longer valid.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage notes

- Repeated metrics retrievals, even consecutive retrievals against a common metrics context, will not necessarily yield metrics for the exact same set of objects. Objects can come and go from the system's inventory due to various circumstances unrelated to the metrics data. The metrics feed for a particular metric group may stop for some reason, and the metrics data may therefore expire from the HMC's buffer (i.e. the metrics cache). The access permissions of a user associated with a metrics context may change, giving the user access to a smaller or larger set of objects (and therefore, perhaps, a smaller or larger set of metrics data).
- It is possible that there may be no metrics to return for one or more metric groups associated with the specified metrics context. For example, data for a metric group may not be buffered on the HMC at the time of the `Get Metrics` invocation, or authorization restrictions related to objects in a requested metric group may prevent any metrics for that group from being returned. If there is no metric data to be returned for a metric group name, then that group name does not appear in the response body. Furthermore, if there is no metric data to return for any metric group name associated with a context, the response body format above specifies that the body will consist only of a single newline character.

This is nonetheless considered a successful response. In other words, an HTTP status code 200 (OK) will still be returned with such a response.

Example HTTP interaction

```
GET /api/services/metrics/context/1 HTTP/1.1
x-api-session: 4rtgkzmzk62b6kz0351uhz5fytb8e0o9weagn4rpj4adv52pf0s
```

Figure 30. Get Metrics: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 22:46:52 GMT
content-type: application/vnd.ibm-z-zmanager-metrics; charset=UTF-8
"channel-usage"
"/api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5"
1568847192154
"0.00",true,"Shared",1
"0.01",true,"Shared",1
"0.04",true,"Shared",1
"0.05",true,"Shared",1
"0.08",true,"Shared",1
"0.09",true,"Shared",1
"0.0C",true,"Shared",1
"0.0D",true,"Shared",1
"0.10",true,"Shared",1
"0.11",true,"Shared",1
"0.14",true,"Shared",1
"0.15",true,"Shared",1
"0.18",true,"Shared",1
"0.19",true,"Shared",1
"0.1C",true,"Shared",1
"0.1D",true,"Shared",1
"0.24",true,"Shared",1
"1.25",false,"LP22",1
"0.28",true,"Shared",1
"0.29",true,"Shared",1
"0.2C",true,"Shared",1
"0.2D",true,"Shared",1
"0.30",true,"Shared",1
"0.31",true,"Shared",1
"0.34",true,"Shared",1
"0.35",true,"Shared",1
"0.E2",true,"Shared",10
"0.E3",true,"Shared",10
"0.FC",false,"CF01",1
"0.FD",false,"LP01",1
"0.FE",false,"CF02",1
"1.FF",false,"LP17",1
<3 blank lines here (consecutive new lines)>
```

Figure 31. Get Metrics: Response

Delete Metrics Context

The Delete Metrics Context operation deletes a metrics context.

HTTP method and URI

```
DELETE /api/services/metrics/context/{metrics-context-id}
```

In this request, the URI variable `{metrics-context-id}` is the identifier of the metrics context object for which metrics are to be obtained.

Description

This operation deletes the metrics context ID. That is, it disassociates it from the API session and cleans up any data associated with it. Further `Get Metrics` requests against this context will result in status code 409 (Conflict).

The URI path must designate an existing valid metrics context for the current API session. If the URI path represents an already invalidated metrics context for the current API session, status code 409 (Conflict) is returned. If the URI path does not represent a recognized metrics context for the current API session, status code 404 (Not Found) is returned.

Authorization requirements

There are no authorization requirements for deleting a metrics context. The association with the API session is implicit, so there is no possibility of deleting a context that was created by a different API session. In other words, only the session which created a metrics context can delete it.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The metrics context ID in the URI (<i>{metrics-context-id}</i>) does not designate a metrics context for the associated API session.
409 (Conflict)	1	The metrics context ID in the URI (<i>{metrics-context-id}</i>) designates a metrics context for the associated API session that is no longer valid.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/services/metrics/context/1 HTTP/1.1
x-api-session: 6a9oz3ymut6rvjijrft0loqhfzgp0rnu4mjishwh6d39jh31q
```

Figure 32. Delete Metrics Context: Request

```
204 No Content
date: Wed, 07 Dec 2011 04:01:59 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 33. Delete Metrics Context: Response

Chapter 9. Metric groups

This chapter provides a description of the metric groups that can be retrieved using the Metrics Service. For each metric group provided by the HMC, the material in this chapter describes the purpose and general characteristics of the metric group, and then defines the content of the metric group via a table that specifies the metric fields provided by the group. The order in which metric fields appear within these tables corresponds to the order in which the data items appear in a value row returned by the `Get Metrics` operation. For example, the metric field appearing in the first row of a metric group table (and identified in a table below as being in position 1) will be the first data item provided in a value row for that metric group; the metric field appearing in the second row (position 2) will be the next data item in a value row, and so on. Thus, the order in which metric fields are documented here is considered semantically significant and can be relied upon by client applications in order to simplify parsing of the data retrieved using the `Get Metrics` operation.

The contents of metric groups may be extended in future versions of this API. If a metric group is extended, new metric fields will be added to the end so as to not alter the relative positions of any of the existing fields. Such new fields would not be understood by a client application designed for an earlier version of the API. Therefore, applications must be developed using the philosophy of "ignore what you don't understand/expect" when processing metric group data in order to tolerate such possible future extensions. See ["Compatibility"](#) on page 5 for more discussion on API compatibility principles.

Monitors dashboard metric groups

The Monitors Dashboard task is the current system monitoring interface on the HMC. It gives a dashboard display to monitor system resources, such as power consumption, environmental data, processor usage, etc.

In order to provide programmatic access to this same data, the utilization and environment data that is displayed on the user interface is also provided via the Metrics Service in the following metric groups.

Channels

This metric group reports the channel usage for each channel on the system. An instance of this metric group is created for each channel of a CPC.

Metric Group Name
"channel-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 47. Channels metric group				
Pos	Metric field name	Type	Units	Description
1	channel-name	String	—	The name of the channel in the form CSS.Chpid
2	shared-channel	Boolean	—	True if the channel is shared among logical partitions, and false if it is not

Table 47. Channels metric group (continued)				
Pos	Metric field name	Type	Units	Description
3	logical-partition-name	String	—	For channel types for which logical partition names are available, the name of the owning logical partition or the value "shared" if the channel is shared. For other channel types for which the name is not available (for example, coupling channels), the value is always an empty string.
4	channel-usage	Integer	%	The channel percent usage (0 – 100%)

CPC overview

This metric group reports the aggregated processor usage and channel usage, the ambient temperature, and total system power consumption for each system. The **cpc-processor-usage** is the average of the percentages of processing capacity for all the physical processors in the CPC. The **channel-usage** is the average of the percentages of I/O capacity for all the channels and adapters in the CPC.

Metric Group Name

"cpc-usage-overview"

Collection Interval

15 seconds

Applicable Managed Object Class

"cpc"

The following metrics are provided in each entry of this metric group:

Table 48. CPC overview metric group				
Pos	Metric field name	Type	Units	Description
1	cpc-processor-usage	Integer	%	The processor percent usage.
2	channel-usage	Integer	%	The channel percent usage.
3	power-consumption-watts	Integer	Watts	The total system power consumption.
4	temperature-celsius	Double	Degrees Celsius	The ambient temperature.
5	cp-shared-processor-usage	Integer	%	The processor percent usage for all CP shared processors. Set to -1 if there are no processors of this type.
6	cp-dedicated-processor-usage	Integer	%	The processor percent usage for all CP dedicated processors. Set to -1 if there are no processors of this type.
7	ifl-shared-processor-usage	Integer	%	The processor percent usage for all IFL shared processors. Set to -1 if there are no processors of this type.
8	ifl-dedicated-processor-usage	Integer	%	The processor percent usage for all IFL dedicated processors. Set to -1 if there are no processors of this type.
9	icf-shared-processor-usage	Integer	%	The processor percent usage for all ICF shared processors. Set to -1 if there are no processors of this type.

Table 48. CPC overview metric group (continued)

Pos	Metric field name	Type	Units	Description
10	icf-dedicated-processor-usage	Integer	%	The processor percent usage for all ICF dedicated processors. Set to -1 if there are no processors of this type.
11	iip-shared-processor-usage	Integer	%	The processor percent usage for all zIIP shared processors. Set to -1 if there are no processors of this type.
12	iip-dedicated-processor-usage	Integer	%	The processor percent usage for all zIIP dedicated processors. Set to -1 if there are no processors of this type.
13	aap-shared-processor-usage	Integer	%	The processor percent usage for all zAAP shared processors. Set to -1 if there are no processors of this type.
14	aap-dedicated-processor-usage	Integer	%	The processor percent usage for all zAAP dedicated processors. Set to -1 if there are no processors of this type.
15	all-shared-processor-usage	Integer	%	The processor percent usage for all shared processors. Set to -1 if there are no processors of this type.
16	all-dedicated-processor-usage	Integer	%	The processor percent usage for all dedicated processors. Set to -1 if there are no processors of this type.
17	cp-all-processor-usage	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
18	ifl-all-processor-usage	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
19	icf-all-processor-usage	Integer	%	The processor percent usage for all ICF processors. Set to -1 if there are no processors of this type.
20	iip-all-processor-usage	Integer	%	The processor percent usage for all zIIP processors. Set to -1 if there are no processors of this type.
21	cbp-shared-processor-usage	Integer	%	The processor percent usage for all CBP shared processors. Set to -1 if there are no processors of this type.
22	cbp-dedicated-processor-usage	Integer	%	The processor percent usage for all CBP dedicated processors. Set to -1 if there are no processors of this type.
23	cbp-all-processor-usage	Integer	%	The processor percent usage for all CBP processors. Set to -1 if there are no processors of this type.

DPM system overview

This metric group reports the aggregated processor usage, network usage, storage usage, accelerator usage, crypto usage, power consumption and temperature for each DPM enabled system.

Metric Group Name

"dpm-system-usage-overview"

Collection Interval

15 seconds

Applicable Managed Object Class

"cpc"

The following metrics are provided in each entry of this metric group:

Table 49. DPM system overview metric group				
Pos	Metric field name	Type	Units	Description
1	processor-usage	Integer	%	The processor percent usage.
2	network-usage	Integer	%	The network percent usage. Set to -1 if there are no adapters of this type.
3	storage-usage	Integer	%	The storage percent usage. Set to -1 if there are no adapters of this type.
4	accelerator-usage	Integer	%	The accelerator percent usage. Set to -1 if there are no adapters of this type.
5	crypto-usage	Integer	%	The crypto percent usage. Set to -1 if there are no adapters of this type.
6	power-consumption-watts	Integer	Watts	The power consumption in watts.
7	temperature-celsius	Double	Degrees Celsius	The ambient temperature.
8	cp-shared-processor-usage	Integer	%	The processor percent usage for all CP shared processors. Set to -1 if there are no processors of this type.
9	cp-all-processor-usage	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
10	ifl-shared-processor-usage	Integer	%	The processor percent usage for all IFL shared processors. Set to -1 if there are no processors of this type.
11	ifl-all-processor-usage	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
12	all-shared-processor-usage	Integer	%	The processor percent usage for all shared processors. Set to -1 if there are no processors of this type.

Logical partitions

This metric group reports the processor usage and z/VM paging rate for each active logical partition (aka Image, LPAR Image, Zone, PR/SM virtual server) on the system.

Metric Group Name
"logical-partition-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"logical-partition"

The following metrics are provided in each entry of this metric group:

<i>Table 50. Logical partitions metric group</i>				
Pos	Metric field name	Type	Units	Description
1	processor-usage	Integer	%	The processor percent usage.
2	zvm-paging-rate	Integer	Pages Per Second	The z/VM paging rate. Only returned for logical partitions running z/VM level 6.1 or higher that have the appropriate agent running in them.
3	cp-processor-usage	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
4	ifl-processor-usage	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
5	icf-processor-usage	Integer	%	The processor percent usage for all ICF processors. Set to -1 if there are no processors of this type.
6	iip-processor-usage	Integer	%	The processor percent usage for all IIP processors. Set to -1 if there are no processors of this type.
7	cbp-processor-usage	Integer	%	The processor percent usage for all CBP processors. Set to -1 if there are no processors of this type.

Partitions

This metric group reports the processor usage, network usage, storage usage, accelerator usage, and crypto usage for each active partition on a DPM enabled system.

Metric Group Name
"partition-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"partition"

The following metrics are provided in each entry of this metric group:

<i>Table 51. Partitions metric group</i>				
Pos	Metric field name	Type	Units	Description
1	processor-usage	Integer	%	The processor percent usage.
2	network-usage	Integer	%	The network percent usage. Set to -1 if there are no adapters of this type.

Table 51. Partitions metric group (continued)				
Pos	Metric field name	Type	Units	Description
3	storage-usage	Integer	%	The storage percent usage. Set to -1 if there are no adapters of this type.
4	accelerator-usage	Integer	%	The accelerator percent usage. Set to -1 if there are no adapters of this type.
5	crypto-usage	Integer	%	The crypto percent usage. Set to -1 if there are no adapters of this type.

zCPC environmentals and power

This metric group reports environmental data and power consumption for the zCPC.

Metric Group Name

"zpc-environmentals-and-power"

Collection Interval

15 seconds

Applicable Managed Object Class

"cpc"

The following metrics are provided in each entry of this metric group:

Table 52. zCPC environmentals and power metric group				
Pos	Metric field name	Type	Units	Description
1	temperature-celsius	Double	Degrees Celsius	The ambient temperature
2	humidity	Integer	%	The relative humidity
3	dew-point-celsius	Double	Degrees Celsius	The dew point
4	power-consumption-watts	Integer	Watts	The power consumption in watts
5	heat-load	Integer	Btu/hour	The total heat load of the system (heat load forced-air + heat load water)
6	heat-load-forced-air	Integer	Btu/hour	The heat load covered by forced-air
7	heat-load-water	Integer	Btu/hour	The heat load covered by water
8	exhaust-temperature-celsius	Double	Degrees Celsius	The exhaust temperature

Power status

This metric group reports line cord power information of connected Power Distribution Units (PDU) or BPAs (Bulk Power Assembly) in the system.

Metric Group Name

"environmental-power-status"

Collection Interval

15 seconds

Applicable Managed Object Class

"cpc"

The following metrics are provided in each entry of this metric group:

Table 53. Power status metric group

Pos	Metric field name	Type	Units	Description
1	linecord-one-name	String	None	Line cord One identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
2	linecord-one-power-phase-A	Integer	Watts	Power in Phase A for line cord One
3	linecord-one-power-phase-B	Integer	Watts	Power in Phase B for line cord One
4	linecord-one-power-phase-C	Integer	Watts	Power in Phase C for line cord One
5	linecord-two-name	String	None	Line cord Two identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
6	linecord-two-power-phase-A	Integer	Watts	Power in Phase A for line cord Two
7	linecord-two-power-phase-B	Integer	Watts	Power in Phase B for line cord Two
8	linecord-two-power-phase-C	Integer	Watts	Power in Phase C for line cord Two
9	linecord-three-name	String	None	Line cord Three identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
10	linecord-three-power-phase-A	Integer	Watts	Power in Phase A for line cord Three
11	linecord-three-power-phase-B	Integer	Watts	Power in Phase B for line cord Three
12	linecord-three-power-phase-C	Integer	Watts	Power in Phase C for line cord Three
13	linecord-four-name	String	None	Line cord Four identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
14	linecord-four-power-phase-A	Integer	Watts	Power in Phase A for line cord Four

Table 53. Power status metric group (continued)

Pos	Metric field name	Type	Units	Description
15	linecord-four-power-phase-B	Integer	Watts	Power in Phase B for line cord Four
16	linecord-four-power-phase-C	Integer	Watts	Power in Phase C for line cord Four
17	linecord-five-name	String	None	Line cord Five identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
18	linecord-five-power-phase-A	Integer	Watts	Power in Phase A for line cord Five
19	linecord-five-power-phase-B	Integer	Watts	Power in Phase B for line cord Five
20	linecord-five-power-phase-C	Integer	Watts	Power in Phase C for line cord Five
21	linecord-six-name	String	None	Line cord Six identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
22	linecord-six-power-phase-A	Integer	Watts	Power in Phase A for line cord Six
23	linecord-six-power-phase-B	Integer	Watts	Power in Phase B for line cord Six
24	linecord-six-power-phase-C	Integer	Watts	Power in Phase C for line cord Six
25	linecord-seven-name	String	None	Line cord Seven identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
26	linecord-seven-power-phase-A	Integer	Watts	Power in Phase A for line cord Seven
27	linecord-seven-power-phase-B	Integer	Watts	Power in Phase B for line cord Seven
28	linecord-seven-power-phase-C	Integer	Watts	Power in Phase C for line cord Seven

Table 53. Power status metric group (continued)

Pos	Metric field name	Type	Units	Description
29	linecord-eight-name	String	None	Line cord Eight identifier. Set to "not-connected" if line cord is not available. If line cord is not available power phases are set to 0.
30	linecord-eight-power-phase-A	Integer	Watts	Power in Phase A for line cord Eight
31	linecord-eight-power-phase-B	Integer	Watts	Power in Phase B for line cord Eight
32	linecord-eight-power-phase-C	Integer	Watts	Power in Phase C for line cord Eight

zCPC processors

This metric group reports the processor usage for each physical zCPC processor on the system. This includes the System Assist Processors (SAPs). An instance of this metric group is created for each processor of a CPC.

Metric Group Name

"zcpc-processor-usage"

Collection Interval

15 seconds

Applicable Managed Object Class

"cpc"

The following metrics are provided in each entry of this metric group:

Table 54. zCPC processors metric group

Pos	Metric field name	Type	Units	Description
1	processor-name	String		The name of the zCPC processor in the form processor-type + processor ID. For example, IFL01.
2	processor-type	String		The type of zCPC processor. The valid types are: "cp" , "ifl" , "icf" , "iip" , "aap" , "sap" and "cbp" . The valid types for DPM enabled systems are: "cp" , "ifl" , and "sap" .
3	processor-usage	Integer	%	The processor percent usage.
4	smt-usage	Integer	%	The percentage of time the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.
5	thread-0-usage	Integer	%	The percent usage of thread 0 when the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.

Table 54. zCPC processors metric group (continued)				
Pos	Metric field name	Type	Units	Description
6	thread-1-usage	Integer	%	The percent usage of thread 1 when the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.

Cryptos

This metric group reports the adapter usage for each crypto on the system. An instance of this metric group is created for each crypto adapter. This metric group is not used for a DPM system. For DPM, crypto adapters are reported in the Adapters metric group. See [“Adapters” on page 138](#).

Metric Group Name
"crypto-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 55. Crypto metric group				
Pos	Metric field name	Type	Units	Description
1	channel-id	String		The physical channel identifier of the crypto
2	crypto-id	String		The crypto identifier of the crypto, decimal 0-15
3	adapter-usage	Integer	%	The adapter percent usage (0-100%)

Adapters

This metric group reports the adapter usage for each adapter on the DPM enabled system. An instance of this metric group is created for each adapter.

Metric Group Name
"adapter-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"adapter"

The following metrics are provided in each entry of this metric group:

Table 56. Adapters metric group				
Pos	Metric field name	Type	Units	Description
1	adapter-usage	Integer	%	The adapter percent usage

Flash memory adapters

This metric group reports the adapter usage for each Flash memory (Flash Express) adapter on the system. An instance of this metric group is created for each Flash memory adapter of the CPC. If a CPC has no flash memory adapters, then no data will appear in this metric group for that CPC.

Note: Flash Express has a planned exploitation of December 2012.

Metric Group Name
"flash-memory-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 57. Flash memory adapters metric group				
Pos	Metric field name	Type	Units	Description
1	channel-id	String		The physical channel identifier of the Flash memory adapter
2	adapter-usage	Integer	%	The adapter percent usage (0-100%)

RoCE adapters

This metric group reports the adapter usage for each RoCE (10GbE RoCE) adapter on the system. An instance of this metric group is created for each RoCE adapter of the CPC.

Metric Group Name
"roce-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 58. RoCE adapters metric group				
Pos	Metric field name	Type	Units	Description
1	channel-id	String		The physical channel identifier of the RoCE adapter
2	adapter-usage	Integer	%	The adapter percent usage (0-100%)

Network management metrics

Following are the network management metric groups.

Network adapter port metric group

OSA and RoCE network adapters have up to two physical ports that connect to the network. Metrics are collected from these ports on a DPM enabled system and provided to the user. This metrics group will contain metrics data representing metrics for one physical port. Metrics are collected and provided on an interval, and each metric provided is the total cumulative value, and not a delta.

Metric Group Name
"network-physical-adapter-port"

Collection Interval
30 seconds

Applicable Managed Object Class
"adapter"

The following metrics are provided in each entry of this metric group:

Table 59. Network adapter port metric group

Pos	Metric field name	Type	Units	Description
1	network-port-id	Integer		Numerical value corresponding to the network adapter's physical port. For OSA adapters, this value can be either 0 or 1, and for RoCE adapters, this value can be 1 or 2. To get more information about the physical port, a URI can be constructed using this value together with the target adapter-id : /api/adapters/{adapter-id}/network-ports/{network-port-id}
2	bytes-sent	Long	Bytes	Number of bytes this physical port sent out to the attached network.
3	bytes-received	Long	Bytes	Number of unicast packets this physical port received from the attached network.
4	packets-sent	Long	Count	Number of unicast packets this physical port sent out to the attached network.
5	packets-received	Long	Count	Number of unicast packets this physical port received from the attached network.
6	packets-sent-dropped	Long	Count	Number of packets that were dropped when this physical port was sending them out to the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
7	packets-received-dropped	Long	Count	Number of packets that were dropped when this physical port was receiving them from the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
8	packets-sent-discarded	Long	Count	Number of packets that were discarded when this physical port was sending them out to the attached network. Packets may be discarded due to errors such as malformed packets.
9	packets-received-discarded	Long	Count	Number of packets that were discarded when this physical port was receiving them from the attached network. Packets may be discarded due to errors such as malformed packets.
10	multicast-packets-sent	Long	Count	Number of multicast packets this physical port sent out to the attached network.

Table 59. Network adapter port metric group (continued)

Pos	Metric field name	Type	Units	Description
11	multicast-packets-received	Long	Count	Number of multicast packets this physical port received from the attached network.
12	broadcast-packets-sent	Long	Count	Number of broadcast packets this physical port sent out to the attached network.
13	broadcast-packets-received	Long	Count	Number of broadcast packets this physical port received from the attached network.
14	interval-bytes-sent	Long	Bytes	Number of bytes sent by this physical port over the collection interval.
15	interval-bytes-received	Long	Bytes	Number of bytes received by this physical port over the collection interval.
16	bytes-per-second-sent	Long	Bytes per second	Number of bytes sent per second by this physical port over the collection interval.
17	bytes-per-second-received	Long	Bytes per Second	Number of bytes per second received by this physical port over the collection interval.
18	utilization	Long	Percentage	Link utilization expressed as usage percentage of overall link bandwidth.
19	mac-address	String		The MAC address of this uplink, if known. If it is not known, then "N/A".
20	flags	Long		Flags indicating the types of metrics that are supported by this interface. The value of this field should be interpreted as a bitmask. The meaning of each bit is as follows: • 0x02 - Byte counts are supported • 0x04 - Packet counts are supported • 0x08 - Drop counts are supported • 0x10 - Discard counts are supported • 0x20 - Multicast counts are supported • 0x40 - Broadcast counts are supported • 0x80 - Interval bytes sent and received are supported

Network interface metric group

This metric group reports metrics for NICs on a DPM enabled system. NICs are network resources associated with DPM partitions. Only NICs that are activated will report metric data. This metrics group will contain metrics data representing metrics for one NIC. Metrics are collected and provided on an interval, and each metric provided is the total cumulative value, and not a delta.

Metric Group Name

"partition-attached-network-interface"

Collection Interval

30 seconds

Applicable Managed Object Class

"nic"

The following metrics are provided in each entry of this metric group:

Table 60. Network interface metric group				
Pos	Metric field name	Type	Units	Description
1	partition-id	String (36)		The unique identifier for the partition that owns the NIC whose metric is contained within this metric group. To get information about the partition, the URI can be constructed using partition-id: <pre>/api/partitions/{partition-id}</pre> To get information about the NIC, the URI can be constructed using the partition-id value together with the target nic-id: <pre>/api/partitions/{partition-id}/nics/{nic-id}</pre>
2	bytes-sent	Long	Bytes	Number of bytes this network adapter sent out to the attached network.
3	bytes-received	Long	Bytes	Number of bytes this network adapter received from the attached network.
4	packets-sent	Long	Count	Number of unicast packets this network adapter sent out to the attached network.
5	packets-received	Long	Count	Number of unicast packets this network adapter received from the attached network.
6	packets-sent-dropped	Long	Count	Number of packets that were dropped when this network adapter was sending them out to the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
7	packets-received-dropped	Long	Count	Number of packets that were dropped when this network adapter was receiving them from the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
8	packets-sent-discarded	Long	Count	Number of packets that were discarded when this network adapter was sending them out to the attached network. Packets may be discarded due to errors such as malformed packets.
9	packets-received-discarded	Long	Count	Number of packets that were discarded when this network adapter was receiving them from the attached network. Packets may be discarded due to errors such as malformed packets.

Table 60. Network interface metric group (continued)

Pos	Metric field name	Type	Units	Description
10	multicast-packets-sent	Long	Count	Number of multicast packets this network adapter sent out to the attached network.
11	multicast-packets-received	Long	Count	Number of multicast packets this network adapter received from the attached network.
12	broadcast-packets-sent	Long	Count	Number of broadcast packets this network adapter sent out to the attached network.
13	broadcast-packets-received	Long	Count	Number of broadcast packets this network adapter received from the attached network.
14	interval-bytes-sent	Long	Bytes	Number of bytes sent by this network adapter over the collection interval.
15	interval-bytes-received	Long	Bytes	Number of bytes received by this network adapter over the collection interval.
16	bytes-per-second-sent	Long	Bytes per second	Number of bytes sent per second by this network adapter over the collection interval.
17	bytes-per-second-received	Long	Bytes per Second	Number of bytes per second received by this network adapter over the collection interval.
18	flags	Long		Flags indicating the types of metrics that are supported by this interface. The value of this field should be interpreted as a bitmask. The meaning of each bit is as follows: • 0x02 - Byte counts are supported • 0x04 - Packet counts are supported • 0x08 - Drop counts are supported • 0x10 - Discard counts are supported • 0x20 - Multicast counts are supported • 0x40 - Broadcast counts are supported • 0x80 - Interval bytes sent and received are supported

Part 3. CPC management

Topics in this part describe CPC management.

Topics covered in this part are:

- [Chapter 10, “Dynamic Partition Manager \(DPM\),” on page 147](#)
- [Chapter 11, “Core IBM Z resources,” on page 561](#)
- [Chapter 12, “Energy management,” on page 945](#)

Chapter 10. Dynamic Partition Manager (DPM)

IBM Dynamic Partition Manager (DPM) expands on the PR/SM (Processor Resource/Systems Manager) concept of logical partitions and provides a simplified, consumable, enhanced user experience reducing the barriers of adoption for new and existing clients.

IBM DPM seamlessly integrates platform I/O resource management, server, network and storage resource provisioning, dynamic resource adjustments and resource monitoring. This enhanced, and encapsulated management through the HMC as the single management end-point eliminates the need for special skills to operate the integrated Z platform and its hardware and virtual infrastructure. The HMC Web Services API enables customers, or IBM and third-party data center management tooling respectively to automate all of the IBM DPM tasks, for example, to discover, manage, and monitor resources, and to provide event notifications.

Read the chapters in Part 4 of the *Dynamic Partition Manager (DPM) Guide, SB10-7170*, for prerequisites for enabling DPM on a mainframe system, and information about supported functions. This part also includes migration instructions and information about I/O adapter configuration.

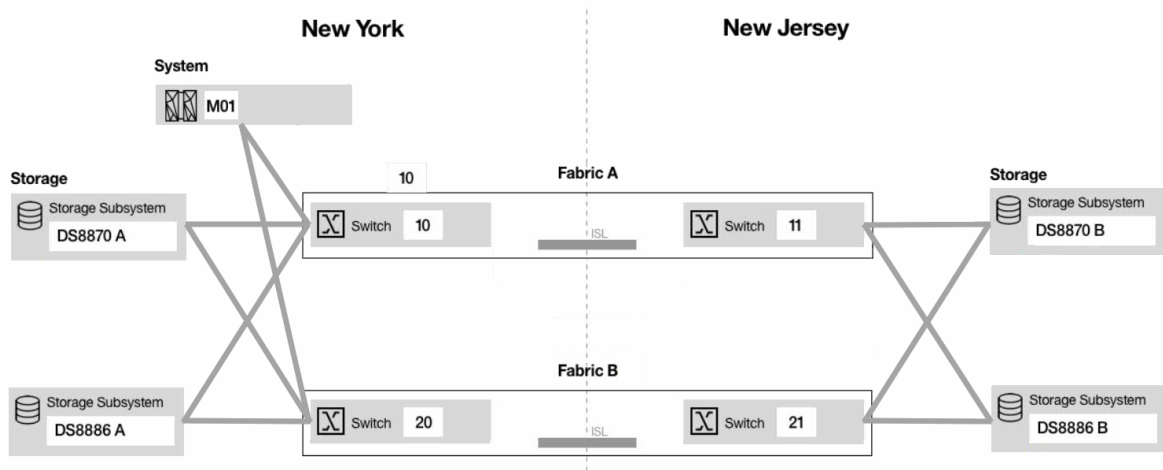
The DPM APIs provide access to and control of the following HMC/SE objects:

- Partition object
- Virtual Function element object
- NIC element object
- HBA element object
- Adapter object
- Network Port element object
- Storage Port element object
- Virtual Switch object
- Capacity Group element object
- FICON Storage Configuration objects:
 - Storage Site object
 - Storage Fabric object
 - Storage Switch object
 - Storage Subsystem object
 - Storage Control Unit object
- Storage Group object
- Storage Template object

FICON storage configuration

A number of the classes defined in the Dynamic Partition Manager chapter are interrelated and in their entirety, define the FICON storage configuration as seen by a single CPC. This configuration is used by the CPC to understand the paths that connect their storage adapters to the storage resources that are needed by the partitions running in that CPC.

The following diagram illustrates the major components of a FICON configuration.



- FICON storage resources are organized into storage sites. The primary site is the one in which the CPC (System "M01" in the diagram) is located. It likely also includes sets of storage switches and storage subsystems that are also local to the CPC. The primary site will always exist. Optionally, a second alternate site can also exist. Alternate sites are typically remote to the CPC, and often used for redundancy. In the diagram, the primary site is labeled "New York" and the alternate site is labeled "New Jersey".
- Storage subsystems represent the physical storage units. They are physically connected (cabled) to a set of storage switches or directly to a set of CPC storage adapters. Storage subsystems are subdivided into logical control units, which provide access to a subset of a subsystem's storage resources. Storage control units are logically connected to CPC storage adapters, optionally through one or two storage switches. In the diagram, storage subsystems named "DS8870 A" and "DS8886 A" are located in the primary "New York" site and storage subsystems named "DS8870 B" and "DS8886 B" are located in the alternate "New Jersey" site.
- Storage switches are optionally used within a CPC storage adapter to storage control unit connection. The switch that is connected to the CPC storage adapter is referred to as the "entrance" switch. The switch that is connected to the storage subsystem is referred to as the "exit" switch. The entrance and exit switches can be the same, which means there is exactly one switch connecting the adapter and control unit. If the entrance and exit switches are different, there are two switches connecting the adapter and control unit, and they are referred to as being in a cascaded switch configuration. In a cascaded configuration, the entrance switch will be in the primary site and the exit switch will be in the alternate site. A storage switch exists in exactly one storage site and in exactly one storage fabric. In the diagram, storage switches 10 and 20 exist in the primary "New York" site. The CPC is connected through those switches to the "DS8870 A" and "DS8886 A" storage subsystems. Switches 11 and 21 exist in the alternate "New Jersey" site. The CPC is connected in cascaded configurations through those switches to the "DS8870 B" and "DS8886 B" storage subsystems.
- The set of all storage switches that are interconnected defines a storage fabric. In a multi-site configuration, switches from both sites are interconnected, therefore in such configurations, a fabric will span multiple sites. In the diagram, storage switches 10 and 11 are interconnected and define storage fabric "Fabric A". Storage switches 20 and 21 define storage fabric "Fabric B".

It is important to understand that a FICON configuration is scoped to a single CPC, and represents that CPC's view of a set of storage resources. A second CPC's FICON configuration may include objects that represent the same physical or logical entities, but they will be returned to an API client as separate objects. The data models for those objects provide no intrinsic way to determine those objects represent the same physical or logical entity. Correlation is possible only if the configurations are created using names or other identifiers that are the same.

Operations summary

Following are the operations summaries for each of the Dynamic Partition Manager (DPM) objects.

Partition operations summary

The following table provides an overview of the operations provided for Partition objects.

Table 61. DPM - Partition: operations summary	
Operation name	HTTP method and URI path
“List Partitions of a CPC” on page 180	GET /api/cpcs/{cpc-id}/partitions
“List Permitted Partitions” on page 182	GET /api/console/operations/list-permitted-partitions
“Create Partition” on page 185	POST /api/cpcs/{cpc-id}/partitions
“Delete Partition” on page 191	DELETE /api/partitions/{partition-id}
“Delete Partition Asynchronously” on page 193	POST /api/partitions/{partition-id}/operations/async-delete
“Get Partition Properties” on page 195	GET /api/partitions/{partition-id}
“Update Partition Properties” on page 198	POST /api/partitions/{partition-id}
“Update Partition Properties Asynchronously” on page 202	POST /api/partitions/{partition-id}/operations/async-update
“Start Partition” on page 206	POST /api/partitions/{partition-id}/operations/start
“Stop Partition” on page 210	POST /api/partitions/{partition-id}/operations/stop
“Dump Partition” on page 213	POST /api/partitions/{partition-id}/operations/scsi-dump
“Start Dump Program” on page 216	POST /api/partitions/{partition-id}/operations/start-dump-program
“Perform PSW Restart” on page 220	POST /api/partitions/{partition-id}/operations/psw-restart
“Create Virtual Function” on page 223	POST /api/partitions/{partition-id}/virtual-functions
“Delete Virtual Function” on page 225	DELETE /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
“Get Virtual Function Properties” on page 227	GET /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
“Update Virtual Function Properties” on page 228	POST /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
“Create NIC” on page 230	POST /api/partitions/{partition-id}/nics
“Delete NIC” on page 235	DELETE /api/partitions/{partition-id}/nics/{nic-id}
“Get NIC Properties” on page 237	GET /api/partitions/{partition-id}/nics/{nic-id}

Table 61. DPM - Partition: operations summary (continued)

Operation name	HTTP method and URI path
“Update NIC Properties” on page 238	POST /api/partitions/{partition-id}/nics/{nic-id}
“Increase Crypto Configuration” on page 242	POST /api/partitions/{partition-id}/operations/increase-crypto-configuration
“Change Crypto Domain Configuration” on page 245	POST /api/partitions/{partition-id}/operations/change-crypto-domain-configuration
“Decrease Crypto Configuration” on page 247	POST /api/partitions/{partition-id}/operations/decrease-crypto-configuration
“Zeroize Crypto Domain” on page 250	POST /api/partitions/{partition-id}/operations/zeroize-crypto-domain
“Mount ISO Image” on page 252	POST /api/partitions/{partition-id}/operations/mount-iso-image
“Unmount ISO Image” on page 254	POST /api/partitions/{partition-id}/operations/unmount-iso-image
“Attach Storage Group to Partition” on page 256	POST /api/partitions/{partition-id}/operations/attach-storage-group
“Detach Storage Group from Partition” on page 258	POST /api/partitions/{partition-id}/operations/detach-storage-group
“Create HBA” on page 260	POST /api/partitions/{partition-id}/hbas
“Delete HBA” on page 263	DELETE /api/partitions/{partition-id}/hbas{hba-id}
“Update HBA Properties” on page 265	POST /api/partitions/{partition-id}/hbas/{hba-id}
“Get HBA Properties” on page 267	GET /api/partitions/{partition-id}/hbas/{hba-id}
“Reassign Storage Adapter Port” on page 268	POST /api/partitions/{partition-id}/hbas/{hba-id}/operations/reassign-storage-adapter-port
“Send OS Command” on page 271	POST /api/partitions/{partition-id}/operations/send-os-cmd
“Open OS Message Channel” on page 272	POST /api/partitions/{partition-id}/operations/open-os-message-channel
“List OS Messages of a Partition” on page 274	GET /api/partitions/{partition-id}/operations/list-os-messages
“Delete Partition OS Message” on page 278	POST /api/partitions/{partition-id}/operations/delete-os-message
“Get ASCII Console WebSocket URI” on page 279	POST /api/partitions/{partition-id}/operations/get-ascii-console-websocket-uri

Table 62. DPM - Partition: URI variables

Variable	Description
{cpc-id}	Object ID of the CPC object.
{partition-id}	Object ID of the Partition object.

Table 62. DPM - Partition: URI variables (continued)	
Variable	Description
<i>{virtual-function-id}</i>	Element ID of the Virtual Function element object.
<i>{nic-id}</i>	Element ID of the NIC element object.
<i>{hba-id}</i>	Element ID of the HBA element object.

Adapter operations summary

The following table provides an overview of the operations provided for Adapter objects.

Table 63. DPM - Adapter: operations summary	
Operation name	HTTP method and URI path
“List Adapters of a CPC” on page 295	GET /api/cpcs/{cpc-id}/adapters
“Get Adapter Properties” on page 298	GET /api/adapters/{adapter-id}
“Update Adapter Properties” on page 299	POST /api/adapters/{adapter-id}
“Change Crypto Type” on page 302	POST /api/adapters/{adapter-id}/operations/change-crypto-type
“Create Hipersocket” on page 304	POST /api/cpcs/{cpc-id}/adapters
“Delete Hipersocket” on page 306	DELETE /api/adapters/{adapter-id}
“Get Partitions Assigned to Adapter” on page 307	GET /api/adapters/{adapter-id}/operations/get-partitions-assigned-to-adapter
“Get Network Port Properties” on page 309	GET /api/adapters/{adapter-id}/network-ports/{network-port-id}
“Update Network Port Properties” on page 311	POST /api/adapters/{adapter-id}/network-ports/{network-port-id}
“Get Storage Port Properties” on page 313	GET /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
“Update Storage Port Properties” on page 314	POST /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
“Change Adapter Type” on page 316	POST /api/adapters/{adapter-id}/operations/change-adapter-type

Table 64. DPM - Adapter: URI variables	
Variable	Description
<i>{cpc-id}</i>	Object ID of a CPC object.
<i>{adapter-id}</i>	Object ID of an Adapter object.
<i>{network-port-id}</i>	Element ID of a Network Port element object.
<i>{storage-port-id}</i>	Element ID of a Storage Port element object.

Virtual Switch operations summary

The following table provides an overview of the operations provided for Virtual Switch objects.

Table 65. DPM - Virtual Switch: operations summary	
Operation name	HTTP method and URI path
“List Virtual Switches of a CPC” on page 321	GET /api/cpcs/{cpc-id}/virtual-switches
“Get Virtual Switch Properties” on page 323	GET /api/virtual-switches/{vswitch-id}
“Get Connected VNICS of a Virtual Switch” on page 324	GET /api/virtual-switches/{vswitch-id}/operations/get-connected-vnics
“Update Virtual Switch Properties” on page 326	POST /api/virtual-switches/{vswitch-id}

Table 66. DPM - Virtual Switch: URI variables	
Variable	Description
{cpc-id}	Object ID of the CPC object
{vswitch-id}	Object ID of the Virtual Switch object

Capacity Group operations summary

The following table provides an overview of the operations provided for Capacity Group objects.

Table 67. DPM - Capacity Group: operations summary	
Operation name	HTTP method and URI path
“List Capacity Groups of a CPC” on page 330	GET /api/cpcs/{cpc-id}/capacity-groups
“Create Capacity Group” on page 332	POST /api/cpcs/{cpc-id}/capacity-groups
“Delete Capacity Group” on page 334	DELETE /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
“Get Capacity Group Properties” on page 336	GET /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
“Add Partition to Capacity Group” on page 338	POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/add-partition
“Remove Partition from Capacity Group” on page 340	POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/remove-partition
“Update Capacity Group Properties” on page 342	POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}

Table 68. DPM - Capacity Group: URI variables	
Variable	Description
{cpc-id}	Object ID of a CPC object
{capacity-group-id}	Element ID of the Capacity Group element object

Storage Site operations summary

The following table provides an overview of the operations provided for Storage Site objects.

Table 69. DPM - Storage Site: operations summary	
Operation name	HTTP method and URI path
“List Storage Sites” on page 345	GET /api/storage-sites
“Create Storage Site” on page 347	POST /api/storage-sites
“Delete Storage Site” on page 350	DELETE /api/storage-sites
“Get Storage Site Properties” on page 352	GET /api/storage-sites/{storage-site-id}
“Update Storage Site Properties” on page 353	POST /api/storage-sites/{storage-site-id}

Table 70. DPM - Storage Site: URI variables	
Variable	Description
{storage-site-id}	Object ID of a Storage Site object

Storage Fabric operations summary

The following table provides an overview of the operations provided for Storage Fabric objects.

Table 71. DPM - Storage Fabric: operations summary	
Operation name	HTTP method and URI path
“List Storage Fabrics” on page 357	GET /api/storage-fabrics
“Create Storage Fabric” on page 359	POST /api/storage-fabrics
“Delete Storage Fabric” on page 362	DELETE /api/storage-fabrics/{storage-fabric-id}
“Get Storage Fabric Properties” on page 363	GET /api/storage-fabrics/{storage-fabric-id}
“Update Storage Fabric Properties” on page 365	POST /api/storage-fabrics/{storage-fabric-id}

Table 72. DPM - Storage Fabric: URI variables	
Variable	Description
{storage-fabric-id}	Object ID of a Storage Fabric object

Storage Switch operations summary

The following table provides an overview of the operations provided for Storage Switch objects.

Table 73. DPM - Storage Switch: operations summary	
Operation name	HTTP method and URI path
“List Storage Switches of a Storage Site” on page 369	GET /api/storage-sites/{storage-site-id}/storage-switches
“List Storage Switches of a Storage Fabric” on page 371	GET /api/storage-fabrics/{storage-fabric-id}/storage-switches

Table 73. DPM - Storage Switch: operations summary (continued)

Operation name	HTTP method and URI path
“Define Storage Switch” on page 373	POST /api/console/operations/define-storage-switch
“Undefine Storage Switch” on page 376	POST /api/storage-switches/{storage-switch-id}/operations/undefine
“Get Storage Switch Properties” on page 377	GET /api/storage-switches/{storage-switch-id}
“Update Storage Switch Properties” on page 379	POST /api/storage-switches/{storage-switch-id}
“Move Storage Switch to Storage Site” on page 381	POST /api/storage-switches/{storage-switch-id}/operations/move-storage-site
“Move Storage Switch to Storage Fabric” on page 383	POST /api/storage-switches/{storage-switch-id}/operations/move-storage-fabric

Table 74. DPM - Storage Switch: URI variables

Variable	Description
{storage-site-id}	Object ID of a Storage Site object
{storage-fabric-id}	Object ID of a Storage Fabric object
{storage-switch-id}	Object ID of a Storage Switch object

Storage Subsystem operations summary

The following table provides an overview of the operations provided for Storage Subsystem objects.

Table 75. DPM - Storage Subsystem: operations summary

Operation name	HTTP method and URI path
“List Storage Subsystems of a Storage Site” on page 387	GET /api/storage-sites/{storage-site-id}/storage-subsystems
“Define Storage Subsystem” on page 389	POST /api/console/operations/define-storage-subsystem
“Undefine Storage Subsystem” on page 392	POST /api/storage-subsystems/{storage-subsystem-id}/operations/undefine
“Get Storage Subsystem Properties” on page 393	GET /api/storage-subsystems/{storage-subsystem-id}
“Update Storage Subsystem Properties” on page 395	POST /api/storage-subsystems/{storage-subsystem-id}
“Move Storage Subsystem to Storage Site” on page 397	POST /api/storage-subsystems/{storage-subsystem-id}/operations/move-storage-site
“Add Connection Endpoint” on page 399	POST /api/storage-subsystems/{storage-subsystem-id}/operations/add-connection-endpoint
“Remove Connection Endpoint” on page 401	POST /api/storage-subsystems/{storage-subsystem-id}/operations/remove-connection-endpoint

Table 76. DPM - Storage Subsystem: URI variables	
Variable	Description
{storage-site-id}	Object ID of a Storage Site object
{storage-subsystem-id}	Object ID of a Storage Subsystem object

Storage Control Unit operations summary

The following table provides an overview of the operations provided for Storage Control Unit objects.

Table 77. DPM - Storage Control Unit: operations summary	
Operation name	HTTP method and URI path
“List Storage Control Units of a Storage Subsystem” on page 407	GET /api/storage-subsystems/{storage-subsystem-id}/storage-control-units
“Define Storage Control Unit” on page 409	POST /api/storage-subsystems/{storage-subsystem-id}/operations/define-storage-control-unit
“Undefine Storage Control Unit” on page 411	POST /api/storage-control-units/{storage-control-unit-id}/operations/undefine
“Get Storage Control Unit Properties” on page 413	GET /api/storage-control-units/{storage-control-unit-id}
“Update Storage Control Unit Properties” on page 414	POST /api/storage-control-units/{storage-control-unit-id}
“Add Volume Range” on page 416	POST /api/storage-control-units/{storage-control-unit-id}/operations/add-volume-range
“Remove Volume Range” on page 418	POST /api/storage-control-units/{storage-control-unit-id}/operations/remove-volume-range
“Create Storage Path” on page 420	POST /api/storage-control-units/{storage-control-unit-id}/storage-paths
“Delete Storage Path” on page 423	DELETE /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
“Get Storage Path Properties” on page 424	GET /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
“Update Storage Path Properties” on page 426	POST /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}

Table 78. DPM - Storage Control Unit: URI variables	
Variable	Description
{storage-subsystem-id}	Object ID of a Storage Subsystem object
{storage-path-id}	Element ID of the Storage Path object
{storage-control-unit-id}	Object ID of a Storage Control Unit object

Storage Group operations summary

The following table provides an overview of the operations provided for Storage Group objects.

Table 79. DPM - Storage Group: operations summary

Operation name	HTTP method and URI path
“List Storage Groups” on page 447	GET /api/storage-groups
“Create Storage Group” on page 450	POST /api/storage-groups
“Delete Storage Group” on page 456	POST /api/storage-groups/{storage-group-id}
“Get Storage Group Properties” on page 459	GET /api/storage-groups/{storage-group-id}
“Modify Storage Group Properties” on page 461	POST /api/storage-groups/{storage-group-id}/operations/modify
“Resend Request” on page 469	POST /api/storage-groups/{storage-group-id}/operations/resend-request
“Add Candidate Adapter Ports to an FCP Storage Group” on page 472	POST /api/storage-groups/{storage-group-id}/operations/add-candidate-adapter-ports
“Remove Candidate Adapter Ports from an FCP Storage Group” on page 474	POST /api/storage-groups/{storage-group-id}/operations/remove-candidate-adapter-ports
“List Storage Volumes of a Storage Group” on page 476	GET /api/storage-groups/{storage-group-id}/storage-volumes
“Get Storage Volume Properties” on page 479	GET /api/storage-groups/{storage-group-id}/storage-volumes{storage-volume-id}
“Fulfill FICON Storage Volume” on page 481	POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-ficon-storage-volume
“Fulfill FCP Storage Volume” on page 488	POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-fcp-storage-volume
“Accept Mismatched FCP Storage Volumes” on page 490	POST /api/storage-groups/{storage-group-id}/operations/accept-mismatched-storage-volumes
“Reject Mismatched FCP Storage Volumes” on page 492	POST /api/storage-groups/{storage-group-id}/operations/reject-mismatched-storage-volumes
“List Virtual Storage Resources of a Storage Group” on page 495	GET /api/storage-groups/{storage-group-id}/virtual-storage-resources
“Get Virtual Storage Resource Properties” on page 498	GET /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}
“Update Virtual Storage Resource Properties” on page 499	POST /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}
“Get Partitions for a Storage Group” on page 502	GET /api/storage-groups/{storage-group-id}/operations/get-partitions
“Validate LUN Path” on page 504	POST /api/cpcs/{cpc-id}/operations/validate-lun-path
“Start FCP Storage Discovery” on page 506	POST /api/storage-groups/{storage-group-id}/operations/start-fcp-storage-discovery

Table 79. DPM - Storage Group: operations summary (continued)	
Operation name	HTTP method and URI path
“Get Connection Report” on page 509	GET /api/storage-groups/{storage-group-id}/operations/get-connection-report
“Get Storage Group Histories” on page 517	GET /api/console/operations/get-storage-group-histories

Table 80. DPM - Storage Group: URI variables	
Variable	Description
{storage-group-id}	Object ID of a Storage Group object
{storage-volume-id}	Element ID of a Storage Volume element object
{virtual-storage-resource-id}	Element ID of a Virtual Storage Resource element object
{cpc-id}	Object ID of the CPC

Storage Template operations summary

The following table provides an overview of the operations provided for Storage Template objects.

Table 81. DPM - Storage Template: operations summary	
Operation name	HTTP method and URI path
“List Storage Templates” on page 541	GET /api/storage-templates
“Create Storage Template” on page 543	POST /api/storage-templates
“Delete Storage Template” on page 546	DELETE /api/storage-templates/{storage-template-id}
“Get Storage Template Properties” on page 548	GET /api/storage-templates/{storage-template-id}
“Modify Storage Template Properties” on page 549	POST /api/storage-templates/{storage-template-id}/operations/modify
“List Storage Template Volumes of a Storage Template” on page 555	GET /api/storage-templates/{storage-template-id}/storage-template-volumes
“Get Storage Template Volume Properties” on page 558	GET /api/storage-templates/{storage-template-id}/storage-template-volumes/{storage-template-volume-id}

Table 82. DPM - Storage Template: URI variables	
Variable	Description
{storage-template-id}	Object ID of a Storage Template object
{storage-template-volume-id}	Element ID of a Storage Template Volume element object

Partition object

The Partition object is central to partition management for IBM Dynamic Partition Manager (DPM). It stores configuration data and is the focal point for various DPM operations.

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 70, including the operational-status-related properties, with the following class-specific specializations:

Table 83. Partition object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path for a Partition object is of the form <code>/api/partitions/{partition-id}</code> , where <code>{partition-id}</code> is the value of the object-id property of the Partition object.
parent	—	String/ URI	The canonical URI path of the hosting CPC object.
class	—	String	The class of the Partition object is "partition" .
name	(w)(pc)	String (1-64)	The name of the partition. The name must be unique on a hosting CPC. The length and character requirements on this property are the same as those described in the “Base managed object properties schema” on page 70.
description	(w)(pc)	String (0-1024)	The description associated with this partition. Default: an empty string.

Table 83. Partition object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
status	(sc)	String Enum	The current operational status of the managed resource. Values: • "communications-not-active" - This status indicates that the HMC is not communicating with the SE. • "status-check" - This status indicates that the current status of the partition is unknown. This state normally indicates a communication issue between the SE and the CPC. • "stopped" - The partition is not running on the CPC. If the partition's reserve-resources property is false, the CPC memory, processor, and adapter resources defined for the partition are not currently allocated to it and are thus available for use by other partitions. • "terminated" - The partition was previously active on the CPC, but is no longer executing the operating system because all processors have been stopped by the operating system via some in-band action. This status may indicate that the operating system has been shut down or has "crashed". This status represents a condition between "active" and "stopped". Although the partition is not executing the operating system, all CPC memory, processor, and adapter resources defined for it remain allocated to it. The Stop Partition action can be used to release the partition's non-reserved resources and place it in "stopped" status. • "starting" - A transitional status between "stopped" and "active" indicating that a Start Partition action has been initiated to put the partition into execution, but processing of that action has not yet completed. • "active" - The partition is up and running. • "stopping" - A transitional status between "active" and "stopped" indicating that a Stop Partition action has been initiated to stop execution of the partition and release its non-reserved resources, but the processing of that action has not yet completed. • "degraded" - The partition is active and one or more resources of the partition are in a degraded state and are not available for use. • "reservation-error" - The partition's reserve-resources property is true but one or more resources of the partition are in a degraded state and are not available for use. • "paused" - The partition was previously active on the CPC, but it is not currently executing the operating system because all processors have been stopped by the user via HMC or SE actions. Typically, partitions are temporarily placed in this status as part of performing diagnosis activities on the operating system in the partition. Although the partition is not currently executing the operating system, all CPC memory, processor, and adapter resources defined for it remain allocated to it. The HMC or SE UI actions can be used to resume execution of the operating system and place it in "active" status again. Alternatively, The Stop Partition action can be used to release the partition's non-reserved resources and place it in "stopped" status.
additional-status	—		This property is not provided.

Class specific additional properties

In addition to the properties defined in the base managed object, this object includes the following additional class-specific properties:

Table 84. Partition object: class specific properties

Name	Qualifier	Type	Description
type	—	String Enum	Defines the type of the partition. One of the following Values: • "linux" - the partition is intended for running a Linux operating system. • "ssc" - the partition will be running an IBM Secure Service Container appliance. • "zvm" - the partition is intended for running a z/VM operating system. The "ssc" type is only available with SE Version 2.13.1 or later that has MCL P00339.304 installed. Default: "linux"
short-name	(w)(pc)	String (1-8)	The short name must be 1-8 characters long, made up of uppercase alphanumeric characters, and have an alphabetic first character. The words PHYSICAL, REC, SYSTEM, and PRIMnnnn (where nnnn is a 4-digit number) are reserved and cannot be used. The short name is provided to the operating system running in the partition, for example by the STORE SYSTEM INFORMATION instruction. Default: Auto-generated Constraint: This property can only be updated when the partition's status is "stopped".

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
partition-id	(w)(pc)	String (2)	The partition ID must be a two character hex number from 00 - 7F. When autogenerate-partition-id is true: - The user will not be able to set the value of partition-id, either on Create Partition or Update Partition Properties. - The value can be either null when the partition is not active and reserve-resources is false, or a value in the defined range. - A non-conflicting value is automatically generated and assigned when the partition is started or when reserve-resources is set to true. It is cleared when the partition is stopped and reserve-resources is set to false. When autogenerate-partition-id is false: - The user must specify a non-conflicting value of partition-id on Create Partition, and can then later change it via Update Partition Properties. - The value specified by the user must be in the defined range. - The value remains assigned to the partition unless changed by the user, or cleared as a consequence of changing the value of autogenerate-partition-id to true. Default: null Constraint: This property can only be updated when the partition's status is "stopped".
autogenerate-partition-id	(w)(pc)	Boolean	Indicates if the partition-id is to be auto-generated. Default: true Constraint: This property can only be updated when the partition's status is "stopped".
os-name	(pc)	String (0-8)	An operating system provided value, used to identify the operating system instance. The format of the value is operating system dependent. If not provided by the operating system or if the partition's status is "stopped", an empty string is returned.
os-type	(pc)	String (0-8)	A human readable form of the operating system provided value for the type of the operating system active in this partition. If not provided by the operating system or if the partition status is "stopped", an empty string is returned.

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
os-version	(pc)	String (0-32)	A human readable form of the operating system provided value for the version of the operating system active in this partition. If not provided by the operating system or if the partition status is "stopped" , an empty string is returned.
reserve-resources	(w)(pc)	Boolean	If true, resource reservation is enabled for this partition, and all physical resources backing the virtual resources configured for this partition are allocated and reserved, even when the partition is in "stopped" state. This guarantees that the partition start will not fail due to non-availability of resources. Default: false
degraded-adapters	(pc)	Array of String/ URI	Array of URIs referring to I/O adapters (NIC, HBA, virtual function, crypto adapter, and virtual storage resource) attached to the partition that are degraded. Only used if the status property of the partition is "degraded" . If the partition has no degraded adapters, the array is empty.
processor-mode	(w)(pc)	String Enum	Defines how processors are allocated to the partition. One of the following values: • "dedicated" - All processors in the partition are to be exclusively available to this specific partition. • "shared" - All processors in the partition are to be shareable across partitions. Default: shared Constraint: This property can only be updated when the partition's status is "stopped" .
cp-processors	(w)(pc)	Integer	Defines the number of general purpose processors (CP) to be allocated for the partition. Default: 0 Note: Exactly one of the cp-processors and ifl-processors must have a value other than 0.
ifl-processors	(w)(pc)	Integer	Defines the number of IFL processors to be allocated for the partition. Default: 0 Note: Exactly one of the cp-processors and ifl-processors must have a value other than 0.
ifl-absolute-processor-capping	(w)(pc)	Boolean	Indicates if absolute processor capping for ifl-processors is enforced. Absolute processor capping prevents this partition from using any more than the specified number of physical processors. Default: false

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
cp-absolute-processor-capping	(w)(pc)	Boolean	Indicates if absolute processor capping for cp-processors is enforced. Absolute processor capping prevents this partition from using any more than the specified number of physical processors. Default: false
ifl-absolute-processor-capping-value	(w)(pc)	Float	The amount of absolute capping applied to ifl-processors . Valid range: 0.01-255.00 in increments of 0.01. Default: 1.0
cp-absolute-processor-capping-value	(w)(pc)	Float	The amount of absolute capping applied to cp-processors . Valid range: 0.01-255.00 in increments of 0.01. Default: 1.0
ifl-processing-weight-capped	(w)(pc)	Boolean	Whether the processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. true : Indicates the IFL processor processing weight for the partition is capped. It represents the partition's maximum share of ifl-processors resources, regardless of the availability of excess IFL processor resources. false : Indicates the IFL processor processing weight for the partition is not capped. It represents the share of ifl-processors resources guaranteed to a partition when all IFL processor resources are in use. Otherwise, when excess IFL processor resources are available, the partition can use them if necessary. Default: false
cp-processing-weight-capped	(w)(pc)	Boolean	Indicates whether the processing weight for general purpose processors is a limit or a target. true : Indicates that the general purpose processor processing weight for the partition is capped. It represents the partition's maximum share of cp-processors . false : Indicates that the general purpose processor processing weight for the partition is not capped. It represents the share of processor resources guaranteed to a partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the partition can use them if necessary. Default: false

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
minimum-ifl-processing-weight	(w)(pc)	Integer	Represents the minimum amount of IFL processor resources allocated to the partition. Valid range: 1-999 Default: 1
minimum-cp-processing-weight	(w)(pc)	Integer	Represents the minimum amount of general purpose processor resources allocated to the partition. Valid range: 1-999 Default: 1
initial-ifl-processing-weight	(w)(pc)	Integer	Defines the initial processing weight of IFL processors. Valid range: 1-999 Default: 100
initial-cp-processing-weight	(w)(pc)	Integer	Defines the initial processing weight of CP processors. Valid range: 1-999 Default: 100
current-ifl-processing-weight	(pc)	Integer	Defines the current IFL processing weight.
current-cp-processing-weight	(pc)	Integer	Defines the current CP processing weight.
maximum-ifl-processing-weight	(w)(pc)	Integer	Represents the maximum amount of IFL processor resources allocated to the partition. Valid range: 1-999 Default: 999
maximum-cp-processing-weight	(w)(pc)	Integer	Represents the maximum amount of shared general processor resources allocated to the partition. Valid range: 1-999 Default: 999
processor-management-enabled	(w)(pc)	Boolean	Indicates whether the processor management is enabled. Default: false

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
initial-memory	(w)(pc)	Integer	The initial amount of memory to assign to the partition when it is started, specified in MB. This value must be at least 4096 when the partition's type is "ssc", and it must be less than or equal to the value of maximum-memory. Default: 1024 Note: If the value input by the user does not fall on an increment boundary, it is rounded off to the closest increment boundary.
reserved-memory	(pc)	Integer	The amount of reserved memory in MB, which equals maximum-memory minus initial-memory.
maximum-memory	(w)(pc)	Integer	The maximum size, specified in MB, to which the partition's memory allocation can be increased while the partition is running. This value must be greater than or equal to the value of initial-memory, and it must be no larger than the amount of entitled memory on the system. Default: 1024 Constraint: This property can only be updated when the partition's status is "stopped". Note: If the value input by the user does not fall on an increment boundary, it is rounded off to the closest increment boundary.
auto-start	(pc)	Boolean	Indicates whether the partition is enabled for auto activation. If true, this partition is automatically started when the CPC is activated. Default: false

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
boot-device	(w)(pc)	String Enum	The type of device from which the partition is booted when it is started. Types: • "storage-adapter" - Boot from the HBA specified in "boot-storage-device". • "storage-volume" - Boot from the storage volume specified in "boot-storage-volume". • "network-adapter" - Boot from the NIC specified in "boot-network-device". • "ftp" - Boot from the host specified in "boot-ftp-host". • "removable-media" - Boot from HMC removable media specified in "boot-removable-media". • "iso-image" - Boot from the ISO image previously mounted and associated with this partition. • "none" - Implies that the partition is not currently bootable. Default: "none" Note: Because a newly created partition does not yet have any HBAs, NICs, mounted ISO images, nor does it have a storage-group attached, , boot device options "network-adapter", "storage-adapter", "iso-image" and "storage-volume" are not valid values for the boot-device field as specified in the request body for the Create Partition operation. If the partition type is "ssc", only "none" can be set as the boot-device when creating or updating a partition.
boot-network-device	(w)(pc)	String/ URI	Specifies the network device that shall be used for the network (PXE) boot. The value must point to a valid NIC URI on the same partition when boot-device mode is "network-adapter". The type of the NIC must be either "iqd" or "osd". The value can be set to either null or a valid URI when boot-device contains a value not relevant to this field. Default: null

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
boot-ftp-host	(w)(pc)	String, String/ IPv4 Address, String/ IPv6 Address	Host name or the IP address of the FTP server that shall be used for the FTP boot. The value must point to a valid host name or the IP address when boot-device mode is "ftp". The value can be set to either null or a valid host name or IP address when boot-device contains a value not relevant to this field. Required if boot-device is "ftp". Default: null
boot-ftp-username	(w)(pc)	String	The user name for the account on the FTP server from which the boot image shall be retrieved. The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "ftp". Default: null
boot-ftp-password	(wo)(pc)	String	The password for the account on the FTP server from which the boot image shall be retrieved. The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "ftp".
boot-ftp-insfile	(w)(pc)	String	The path to the INS-file on the FTP server. The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "ftp". Default: null
boot-removable-media	(w)(pc)	String	Specifies the boot image or the CD/DVD or the USB media containing a bootable image. This must point to a fully-qualified path on the HMC. The value must point to a valid path name when boot-device mode is "removable-media". The value can be set to either null or a valid path name when boot-device contains a value not relevant to this field. Required if boot-device is "removable-media". Default: null

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
boot-removable-media-type	(w)(pc)	String Enum	Specifies the type of the removable media. Valid values are "cdrom" and "usb". The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "removable-media". Default: null
boot-timeout	(w)(pc)	Integer (60-600)	The time, in seconds, that is waited before an ongoing boot is aborted. This is applicable for all modes of boot-device. Default: 60
boot-storage-device	(w)(pc)	String/ URI	Specifies the HBA that shall be used for the partition to boot. The value has to point to a valid HBA URI on the partition when boot-device mode is "storage-adapter". The value can be set to either null or a valid HBA URI when boot-device contains a value not relevant to this field. Default: null
boot-storage-volume	(w)(pc)	String/ URI	Specifies the volume that shall be used for the partition to boot. The value has to point to a valid URI of a storage volume of type "boot" contained in a storage group attached to the partition when boot-device is "storage-volume". The value can be set to either null or a valid boot volume URI when boot-device contains a value not relevant to this field. Default: null
boot-logical-unit-number	(w)(pc)	String (1-16)	The hexadecimal logical unit number (LUN) representing the boot device. The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "storage-adapter". Default: an empty string

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
boot-world-wide-port-name	(w)(pc)	String (16)	The world wide port name (WWPN) of the storage controller containing the target SCSI device to be used for boot, in hexadecimal. The value can be set to either null or a valid value when boot-device contains a value not relevant to this field. Required if boot-device is "storage-adapter". Default: an empty string
boot-configuration-selector	(w)(pc)	Integer	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system to be loaded. Whether and how this parameter is used to determine boot parameters depends on the operating system and its boot process. For Linux on z Systems®, for example, this parameter selects which of the operating system's pre-configured boot configurations is to be used, with the selected boot configuration in turn specifying parameters such as the kernel to be loaded, the kernel parameters to be used, or which disk is used as part of the boot process. Valid range: 0-30 Default: 0, which indicates that the operating system's default boot configuration should be used. Note: Not applicable if "boot-storage-volume" points to a FICON storage volume.
boot-record-lba	(w)(pc)	String (1-16)	Specifies the logical block number, in hexadecimal, of the anchor point for locating the operating system on the SCSI disk from which the operating system is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. For Linux on z Systems, for example, this parameter specifies the block number of the master boot record, which is usually the first block (block number 0) on the boot device. Default: 0, identifying the first block on the device. Note: Not applicable if "boot-storage-volume" points to a FICON storage volume.

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
boot-load-parameters	(w)(pc)	String (0-8)	Specifies parameters that are passed unmodified to the operating system boot process. The way in which these parameters are used depends on the operating system, but in general, these parameters are intended to be used to select an entry in the boot menu or the boot loader. Valid characters are 0-9, A-Z, @, \$, #, blank (), and period (). Default: an empty string
boot-os-specific-parameters	(w)(pc)	String (0-256)	Specifies parameters that are passed unmodified to the loaded operating system as part of the boot process. The way in which these parameters are used depends on the operating system, but in general, these parameters are intended to specify boot-time configuration settings. For Linux on z Systems, for example, this property can be used to specify kernel parameters. Default: an empty string Note: Not applicable if "boot-storage-volume" points to a FICON storage volume.
boot-iso-image-name	(pc)	String	Name of the ISO image. This property is changed by the Mount ISO Image and Unmount ISO Image operations. This name must not be an empty string, and it must not contain any of the following characters: \, ", <, >, , :, &, \$, *, / Default: null
boot-iso-ins-file	(w)(pc)	String	INS file location within the ISO image. This property is changed by the Mount ISO Image and Unmount ISO Image operations. Default: null
secure-execution ²	(pc)	Boolean	If true, Secure Execution for Linux is enabled. If false, Secure Execution for Linux is not enabled. If the partition's status is "stopped", a null object is returned.

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
secure-boot²	(w)(pc)	Boolean	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. This property is only valid for partitions with type "linux". Beyond that, it is only applicable for partitions with boot-device "storage-volume" with volume type fcp. On an Update request, this property is only allowed in the request body if the preconditions listed above are met. Default: false
access-global-performance-data	(w)(pc)	Boolean	Indicates if global performance data authorization control is requested. Default: false
permit-cross-partition-commands	(w)(pc)	Boolean	Indicates if cross partition commands authorization control is requested. Default: false
access-basic-counter-set	(w)(pc)	Boolean	Indicates if basic counter set authorization control is requested. Default: false
access-problem-state-counter-set	(w)(pc)	Boolean	Indicates if problem state counter set authorization control is requested. Default: false
access-crypto-activity-counter-set	(w)(pc)	Boolean	Indicates is crypto activity counter set authorization control is requested. Default: false
access-extended-counter-set	(w)(pc)	Boolean	Indicates if extended counter set authorization control is requested. Default: false
access-coprocessor-group-set	(w)(pc)	Boolean	Indicates if coprocessor group set authorization control is requested. Default: false
access-basic-sampling	(w)(pc)	Boolean	Indicates if basic CPU sampling authorization control is requested. Default: false
access-diagnostic-sampling	(w)(pc)	Boolean	Indicates if diagnostic sampling authorization control is requested. May only be true if access-basic-sampling is true. Default: false

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
permit-des-key-import-functions	(w)(pc)	Boolean	Enables/disables the importing of DES keys for the associated partition. Default: true
permit-aes-key-import-functions	(w)(pc)	Boolean	Enables/disables the importing of AES keys for the associated partition. Default: true
permit-ecc-key-import-functions²	(w)(pc)	Boolean	Enables/disables the importing of Elliptic Curve Cryptography (ECC) keys for the associated partition. Default: true
threads-per-processor	(pc)	Integer	The number of threads the operating system running in this partition can use, for each processor allocated. If the partition has never been activated, a value of 0 is returned. After the initial activation of the partition, this value is controlled by the SMT (Simultaneous Multi-Threading) setting in the OS.
virtual-function-uris	(pc)	Array of String/ URI	Array of URIs referring to virtual functions associated with the partition. If the partition has no virtual functions, the array is empty. Default: an empty array
nic-uris	(pc)	Array of String/ URI	Array of URIs referring to defined NICs (network adapters) attached to the partition. If the partition has no NICs, the array is empty. Default: an empty array
hba-uris	(pc)	Array of String/ URI	Array of URIs referring to defined HBAs attached to the partition. If the partition has no HBAs, the array is empty. Default: an empty array
storage-group-uris	(pc)	Array of String/ URI	Array of URIs referring to storage groups attached to the partition. If the partition has no attached storage groups, or if the DPM Storage feature is disabled, the array is empty. Default: an empty array
crypto-configuration	(pc)	crypto-configuration object	Single instance of a crypto-configuration nested object. See “crypto-configuration object properties” on page 174 . The Increase Crypto Adapter Configuration operation can be used to set the crypto configuration for the partition. Default: null

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
ssc-host-name¹	(w)(pc)	String/ Hostname	The Secure Service Container host name. This string meets the requirements of the String/Hostname data type with the following exceptions: - length is 1-64 characters - valid characters are a-z, A-Z, 0-9, period(.), hyphen(-), and underscore(_)
ssc-boot-selection¹	(w)(pc)	String Enum	Indicates whether to run the Secure Service Container appliance installer or the Secure Service Container appliance itself. One of: "installer" - Boot the Secure Service Container appliance installer to install the Secure Service Container appliance and then start it. "appliance" - Start the most recently installed Secure Service Container appliance and resume its execution from where it was when the partition was stopped. On an Update request, this property can be set from "appliance" to "intaller" only. Default: "installer"
ssc-ipv4-gateway¹	(w)(pc)	String/ IPv4 Address	The default IPv4 Gateway to be used when there is at least one NIC configured in static IPv4 mode. Default: null
ssc-ipv6-gateway	(w)(pc)	String/ IPv6 Address	The default IPv6 Gateway to be used when there is at least one NIC configured in static IPv6 mode. Default: null
ssc-dns-servers¹	(w)(pc)	Array of String/ IPv4 or IPv6 Address	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set. Default: An empty array
ssc-master-userid¹	(w)(pc)	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and underscore(_). Default: null
ssc-master-pw¹	(wo)(pc)	String (8-256)	The Secure Service Container master user password. Valid characters are: a-z, A-Z, 0-9, and !@#\$%^&*()_+{} <>?-=. Default: null
available-features-list	—	Array of partition- feature-info objects	The list of optional features or behavior supported by this Partition. If the Partition has no optional features, then an empty array is provided.

Table 84. Partition object: class specific properties (continued)

Name	Qualifier	Type	Description
Notes: ¹ On a Get request, this property is returned only when type is "ssc" . On an Update request, this property can be updated only when type is "ssc" . ² This property is returned only when the SE version is 2.15.0 or later.			

Table 85. partition-feature-info object properties

Name	Type	Description
name	String Enum	The name of the feature. One of: "dpm-storage-management" - Indicates that the Partition supports Storage Groups and FICON storage resources. FCP and FICON storage resources are defined in Storage Groups, which are then attached to this Partition. If the Partition does not have this feature, FCP storage resources are represented by HBAs, which must be directly attached to this Partition. This feature is inherited from the "dpm-storage-management" feature on the hosting CPC object, and thus, is enabled only when the hosting CPC object has this feature enabled and is disabled otherwise. See Chapter 6, "Firmware features," on page 73 for a list of operations that are affected for each of these features.
description	String	A brief description of the feature.
state	Boolean	Indicates if the feature is currently enabled (true) or disabled (false) for this Partition.

crypto-configuration object properties

The crypto configuration of a partition represents the elements that are required to enable the partition to make use of crypto adapters. The configuration is a nested structure, containing two pieces of information:

- A set of crypto adapters that will be used by this partition, and
- A set of Crypto Domain Configuration objects. (See Table 87 on page 175.)

A crypto configuration that contains no crypto adapters and no crypto domain configurations is valid and is known as an *empty crypto configuration*. A non-empty configuration must contain at least 1 crypto adapter and at least 1 crypto domain configuration with an **access-mode** of **"control-usage"**.

Table 86. crypto-configuration nested object properties

Name	Type	Description
crypto-adapter-uris	Array of String/ URI	Array of URIs listing all crypto adapters that this partition can use.
crypto-domain-configurations	Array of crypto-domain-configuration objects	Array listing all crypto-domain-configuration objects for this partition. See Table 87 on page 175.

Table 87. crypto-domain-configuration nested object properties		
Name	Type	Description
domain-index	Integer	Index value that identifies the domain to which this configuration applies. Minimum index is 0, maximum index depends on the CPC model.
access-mode	String Enum	Specifies the way in which the partition can use this domain. Valid values are: • "control" - The partition can load cryptographic keys into the domain, but it may not use the domain to perform cryptographic operations. • "control-usage" - The partition can load cryptographic keys into the domain, and it can use the domain to perform cryptographic operations.

Crypto configuration conflicts

A crypto configuration conflict occurs when the crypto configuration of two (or more) partitions:

1. Have one (or more) adapter(s) in common, and
2. Specified **"control-usage"** for one (or more) identical domain index(es).

No more than one of the partitions involved in a given crypto configuration conflict may be active or have reserved resources at any one point in time.

According to this definition, the crypto configuration of two partitions can have multiple conflicts (regarding different adapters and/or different domains).

It is also possible for a partition to be involved in conflicts with multiple other partitions. For example, Partition A has 3 crypto adapters in its configuration. Partition B has 2 of those and Partition C has the other one. Assuming they all have a control-usage domain in common, Partition A is now involved in a conflict with Partition B and a separate conflict with Partition C.

Such conflicts are only allowed for partitions that are in **"stopped"** state, and without reserved resources. That means the system will prevent the creation of conflicting crypto configuration for the set of active partitions, and the set of **"stopped"** partitions that have **reserve-resources** enabled.

Data model - Virtual Function element object

The following table contains the Virtual Function element object properties.

Table 88. Partition object - Virtual Function element properties			
Name	Qualifier	Type	Description of specialization
element-id	—	String (36)	The unique identifier for the virtual function instance.
element-uri	—	String/URI	The canonical URI path for the virtual function is of the form <code>/api/partitions/{partition-id}/virtual-functions/{virtual-function-id}</code> , where <code>{partition-id}</code> is the object-id of the partition, and the <code>{virtual-function-id}</code> is the element-id of the virtual function.
parent	—	String/URI	The URI path of the partition that hosts this virtual function.
class	—	String (16)	Always "virtual-function" .

Table 88. Partition object - Virtual Function element properties (continued)			
Name	Qualifier	Type	Description of specialization
name	(w)(pc)	String (1-64)	Name of the virtual function. The name must be unique among all virtual functions of the partition. The length and character requirements on this property are the same as those described in the “Base managed object properties schema” on page 70.
description	(w)(pc)	String (0-1024)	Description of the virtual function. Default: an empty string.
device-number	(w)(pc)	String (4)	Device number of the virtual function. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. Default: auto-generated. Constraint: This number must be unique across the device numbers of all other Virtual Function elements and all NIC elements of type "roce" of the partition.
adapter-uri	(w)(pc)	String/URI	The canonical URI path for the associated Accelerator adapter.
fid	—	Integer	Functional ID of the associated accelerator adapter identified in adapter-uri , or null if the partition is not active.

Data model - NIC element object

The following table contains the NIC element object properties.

Table 89. Partition object - NIC element object properties			
Name	Qualifier	Type	Description of specialization
element-id	—	String (36)	The unique identifier for the NIC within the scope of the partition.
element-uri	—	String/URI	The canonical URI path for the NIC is of the form <code>/api/partitions/{partition-id}/nics/{nic-id}</code> , where <code>{partition-id}</code> is the object-id of the partition, and the <code>{nic-id}</code> is the element-id of the NIC.
parent	—	String/URI	The URI path of the partition that hosts this NIC.
class	—	String (3)	Always "nic" .
name	(w)(pc)	String (1-64)	Name of the NIC. The name must be unique among all NICs of the partition. The length and character requirements on this property are the same as those described in the “Base managed object properties schema” on page 70.
description	(w)(pc)	String (0-1024)	Description of the NIC. Default: an empty string.

Table 89. Partition object - NIC element object properties (continued)

Name	Qualifier	Type	Description of specialization
device-number	(w)(pc)	String (4)	Device number of the NIC. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. If type is "osd" a range of 3 device numbers will be allocated. If type is "cna", the range of device numbers allocated is indicated by the function-range property. Default: auto-generated. Constraint: If type is "roce" or "cna", this number must be unique across the device numbers of all other NIC elements of type "roce" and among all Virtual Function elements of the partition. If type is "iqd" or "osd", this number must be unique across the device numbers of all other NIC elements of type "iqd" or "osd" and all Storage Group or HBA elements of the partition.
network-adapter-port-uri	(w)(pc)	String/ URI	The canonical URI path for the associated Network Port element object. Only present when type is "roce" or "cna".
virtual-switch-uri	(w)(pc)	String/ URI	The canonical URI path for the associated Virtual Switch object. Only present when type is "osd" or "iqd".
type	—	String Enum	The type of the NIC. The value of this property is derived implicitly from the backing adapter associated with this NIC on the Create NIC operation. Valid values are: • "roce" - RDMA over Converged Ethernet. • "iqd" - Internal Queued Direct. • "osd" - OSA Direct Express • "cna" - Cloud Network Adapter
ssc-management-nic	(w)(pc)	Boolean	Indicates that this NIC should be used as a management NIC for Secure Service Container to access the web interface. Can only be set to true if the partition's type is "ssc". If the partition's type is "ssc", there must be at least one management NIC defined before the partition can be started. If true, other parameters are required (at least ssc-ip-address-type) for this NIC. If true, only OSA or HiperSockets adapters can be selected as backing adapters. If the associated SE is version 2.13.1, the only valid port for an OSA backing adapter is port 0.

Table 89. Partition object - NIC element object properties (continued)

Name	Qualifier	Type	Description of specialization
ssc-ip-address-type¹	(w)(pc)	String Enum	Secure Service Container IP address type. Valid types are: • "ipv4" - Network is configured in static IPv4 mode • "ipv6" - Network is configured in static IPv6 mode • "linklocal" - Network is configured in Link Local mode • "dhcp" - Network is configured in DHCP mode.
ssc-ip-address¹	(w)(pc)	String/ IPv4 Address or String/ IPv6 Address	The IP address of the Secure Service Container management web interface.
ssc-mask-prefix¹	(w)(pc)	String	Network mask of the Secure Service Container management NIC. Either the mask is provided in bit notation, e.g. "/24" (both for IPv4 and IPv6), or in mask notation, e.g. "255.255.255.0" (IPv4 only).
vlan-id	(w)(pc)	Integer (1-4094)	The VLAN ID associated with this NIC. When the partition's type is "ssc" , this property is allowed only if the value of ssc-management-nic is "true" . It can be null. When the partition's type is not "ssc" , this property is not allowed when the type of the NIC is "roce" or "cna" .
mac-address	(w)(pc)	String (17)	The MAC address associated with this NIC. It must be unique among all the NICs created in the CPC. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addresses are valid, e.g. "02:ff:12:34:56:78". This value can not be set when the type of the NIC is "roce" or "cna" . Default: Auto-generated
vlan-type	(w)(pc)	String Enum	The type of VLAN tagging to use for the VLAN associated with this NIC, or null, if the NIC is not associated with a VLAN. Valid value: • "enforced" - the network adapter only allows untagged packets or packets tagged for the VLAN identified by vlan-id through to the operating system running in the partition. The network device in the operating system should also be configured with the same vlan-id. This value can not be set when the partition's type is "ssc" or when the type of the NIC is "roce" or "cna" .

Table 89. Partition object - NIC element object properties (continued)

Name	Qualifier	Type	Description of specialization
function-number	(w)(pc)	Integer	The function number of the PCI function on the adapter. This property can only be set when the type of the NIC is "cna" . A value of 0 indicates a physical PCI function (PF); all other values indicate a virtual PCI function (VF). Default: 0
function-range	(w)(pc)	Integer (1-128)	The number of PCI functions on the NIC. This property can only be set when the type of the NIC is "cna" . It controls the number of virtual PCI functions that are created. For example, if function-number is 0 and function-range is 128, that results in 1 physical PCI function and 127 virtual PCI functions. Default: 128
¹ Only applicable if ssc-management-nic is true .			

Data model - HBA element object

An HBA represents a single Host Bus Adapter (HBA) available to a partition in a CPC. An HBA is an access point between a CPC and a Storage Area Network (SAN).

The following table contains the HBA element object properties.

Table 90. Partition object - HBA element object properties

Name	Qualifier	Type	Description of specialization
element-uri	—	String/ URI	The canonical URI path of an HBA element is of the form <code>/api/partitions/{partition-id}/hbas/{hba-id}</code> where <code>{partition-id}</code> represents the object-id of the Partition object and <code>{hba-id}</code> represents the element-id of the HBA.
element-id	—	String (36)	The unique identifier for an HBA element. The string form of a UUID.
parent	—	String/ URI	The canonical URI path of the Partition object.
class	—	String	The class of an HBA element is "hba" .
name	(w)(pc)	String (1-64)	The display name of an HBA element. This name must be unique among all of the partition's HBA elements, and it must conform to the length and character requirements of the name property described in the "Base managed object properties schema" on page 70.
description	(w)(pc)	String (0-1024)	The description of the HBA element. Default: an empty string.
wwpn	—	String (16)	The world wide port name of the HBA element. The string form of wwpn is of 16 hexadecimal characters.

Table 90. Partition object - HBA element object properties (continued)

Name	Qualifier	Type	Description of specialization
device-number	(w)(pc)	String (4)	Device number of the HBA. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. Default: auto-generated. Constraint: This number must be unique across the device numbers of all other HBA elements and all NIC elements of type "iqd" or "osd" of the partition.
adapter-port-uri	(pc)	String/ URI	The canonical URI path of the Storage Port element object to which this HBA is connected.

List Partitions of a CPC

The List Partitions of a CPC operation lists the partitions of a CPC.

HTTP method and URI

GET /api/cpcs/{cpc-id}/partitions

In this request, the URI variable {cpc-id} is the object ID of the target CPC.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property.
status	String Enum	Optional	Filter string to limit returned objects to those that have a matching status property. Value must be a valid partition status property value.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid partition type property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
partitions	Array of partition-info objects	Array of nested partition-info objects, described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the Partition object.
name	String	The name property of the Partition object.
status	String Enum	The status property of the Partition object.
type	String Enum	The type property of the Partition object.

Description

This operation lists the partition objects that belong to a CPC. The object URI, display name, status, and type are provided for each.

If the **name** query parameter is specified, the returned list is limited to those partition objects that have a **name** property matching the specified filter pattern. If the **status** query parameter is specified, the returned list is limited to those partition objects that have a **status** property matching the specified filter value. If the **type** query parameter is specified, the returned list is limited to those partition objects that have a **type** property matching the specified filter value. If no query parameters are provided, no filtering is done.

An object is only included in the list if the API user has object-access permission for that object

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 180](#).

If the CPC is not in DPM mode, or there are no partitions defined to the CPC, or no partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by `{cpc-id}`.
- Object-access permission to any Partition object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 180](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/93634ff4-0599-3f7d-b937-7673de7dfd0c/partitions?name=t.* HTTP/1.1
x-api-session: 2izurpik57ciomzst8z0q1vsqg2kuvfe9qxdja6irmbovo8z1c
```

Figure 34. List Partitions of a CPC: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Feb 2017 09:08:33 GMT
content-type: application/json; charset=UTF-8
content-length: 504
{
  "partitions": [
    {
      "name": "testpweights",
      "object-uri": "/api/partitions/2fa1f646-e9d9-11e6-a392-42f2e9cfe851",
      "status": "stopped",
      "type": "ssc"
    },
    {
      "name": "testVnic",
      "object-uri": "/api/partitions/bc09b56e-e88b-11e6-8715-42f2e9cfe851",
      "status": "active",
      "type": "linux"
    },
    {
      "name": "testCrypto",
      "object-uri": "/api/partitions/798167ba-ec4a-11e6-a040-42f2e9cfe851",
      "status": "stopped",
      "type": "linux"
    },
    {
      "name": "test_ah",
      "object-uri": "/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851",
      "status": "stopped",
      "type": "ssc"
    }
  ]
}

```

Figure 35. List Partitions of a CPC: Response

List Permitted Partitions

The List Permitted Partitions operation lists partitions to which the API user has object-access permission.

HTTP method and URI

GET /api/console/operations/list-permitted-partitions

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid partition type property value.
status	String Enum	Optional	Filter string to limit returned objects to those that have a matching status property. Value must be a valid partition status property value.
has-unacceptable-status	Boolean	Optional	Filter string to limit returned objects to those that have a matching has-unacceptable-status property. Valid values are true and false .

Name	Type	Rqd/Opt	Description
cpc-name	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching name property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
partitions	Array of partition-info objects	Array of nested partition-info objects as described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
name	String	The name property of the Partition object.
object-uri	String/ URI	The object-uri property of the Partition object.
type	String Enum	The type property of the Partition object.
status	String Enum	The status property of the Partition object.
has-unacceptable-status	Boolean	The has-unacceptable-status property of the Partition object.
cpc-name	String	The name property of the partition's parent CPC object.
cpc-object-uri	String/ URI	The object-uri property of the partition's parent CPC object.
se-version	String	The se-version property of the partition's parent CPC object.

Description

This operation lists the Partition objects to which the API user has object-access permission. Some basic properties are provided for each partition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid partition **type** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have a **type** property matching the specified value. If the **type** parameter is omitted, no such filtering is performed.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid partition **status** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have a **status** property matching the specified value. If the **status** parameter is omitted, no such filtering is performed.

If the **has-unacceptable-status** query parameter is specified, the returned list is limited to those partitions that have a **has-unacceptable-status** property matching the specified value. If the **has-unacceptable-status** parameter is omitted, no such filtering is performed.

If the **cpc-name** query parameter is specified, the returned list is limited to those partitions whose parent CPC's **name** property matches the specified filter pattern. If the **cpc-name** parameter is omitted, no such filtering is performed.

A partition is included in the list only if the API user has object-access permission to that object. If there is a partition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no partitions known to the HMC or if no partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 183](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/list-permitted-partitions HTTP/1.1
x-api-session: 1hxko1kyziy64pcd9a9ot59ceb9jnh7vg55ylro930kubzgva5
```

Figure 36. List Permitted Partitions: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 27 Aug 2018 18:37:52 GMT
content-type: application/json; charset=UTF-8
content-length: 294
{
  "partitions": [
    {
      "cpc-name": "SEDPM005",
      "cpc-object-uri": "/api/cpcs/e4e18781-8063-3f2c-8222-044eb58988d9",
      "se-version": "2.14.0",
      "has-unacceptable-status": true,
      "name": "part1",
      "object-uri": "/api/partitions/592125be-76dd-11e7-94f9-02c2000226b7",
      "status": "communications-not-active",
      "type": "linux"
    }
  ]
}

```

Figure 37. List Permitted Partitions: Response

Usage note

The response body of this operation is similar to that of the `Get Inventory` operation, but it returns only a subset of partition properties. The response also includes some properties of the parent CPC, regardless of whether the API user has object-access permission to that CPC.

Create Partition

The `Create Partition` operation creates a partition with the given properties on the identified CPC.

HTTP method and URI

POST `/api/cpcs/{cpc-id}/partitions`

In this request, the URI variable `{cpc-id}` is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
type	String Enum	Optional	The value to be set as the partition's type property.
name	String (1-64)	Required	The value to be set as the partition's name property.
description	String (0-1024)	Optional	The value to be set as the partition's description property.
short-name	String (8)	Optional	The value to be set as the partition's short-name property.
partition-id	String (2)	Required if autogenerate-partition-id is false	The value to be set as the partition's partition-id property.
autogenerate-partition-id	Boolean	Optional	The value to be set as the partition's autogenerate-partition-id property.

Field name	Type	Rqd/Opt	Description
ifl-processors	Integer	Required if cp-processors is not provided	The value to be set as the partition's ifl-processors property.
cp-processors	Integer	Required if ifl-processors is not provided	The value to be set as the partition's cp-processors property.
processor-mode	String Enum	Optional	The value to be set as the partition's processor-mode property.
initial-memory	Integer	Required	The value to be set as the partition's initial-memory property.
maximum-memory	Integer	Required	The value to be set as the partition's maximum-memory property.
reserve-resources	Boolean	Optional	The value to be set as the partition's reserve-resources property.
boot-device	String Enum	Optional	The value to be set as the partition's boot-device property.
boot-timeout	Integer (60-600)	Optional	The value to be set as the partition's boot-timeout property.
boot-ftp-host	String	Required if boot-device is "ftp"	The value to be set as the partition's boot-ftp-host property.
boot-ftp-username	String	Required if boot-device is "ftp"	The value to be set as the partition's boot-ftp-username property.
boot-ftp-password	String	Required if boot-device is "ftp"	The value to be set as the partition's boot-ftp-password property.
boot-ftp-insfile	String	Required if boot-device is "ftp"	The value to be set as the partition's boot-ftp-insfile property.
boot-removable-media	String	Required if boot-device is "removable-media"	The value to be set as the partition's boot-removable-media property.
boot-removable-media-type	String Enum	Required if boot-device is "removable-media"	The value to be set as the partition's boot-removable-media-type property.
access-global-performance-data	Boolean	Optional	The value to be set as the partition's access-global-performance-data property.
permit-cross-partition-commands	Boolean	Optional	The value to be set as the partition's permit-cross-partition-commands property.
access-basic-counter-set	Boolean	Optional	The value to be set as the partition's access-basic-counter-set property.
access-problem-state-counter-set	Boolean	Optional	The value to be set as the partition's access-problem-state-counter-set property.

Field name	Type	Rqd/Opt	Description
access-crypto-activity-counter-set	Boolean	Optional	The value to be set as the partition's access-crypto-activity-counter-set property.
access-extended-counter-set	Boolean	Optional	The value to be set as the partition's access-extended-counter-set property.
access-coprocessor-group-set	Boolean	Optional	The value to be set as the partition's access-coprocessor-group-set property.
access-basic-sampling	Boolean	Optional	The value to be set as the partition's access-basic-sampling property.
access-diagnostic-sampling	Boolean	Optional	The value to be set as the partition's access-diagnostic-sampling property.
permit-des-key-import-functions	Boolean	Optional	The value to be set as the partition's permit-des-key-import-functions property.
permit-aes-key-import-functions	Boolean	Optional	The value to be set as the partition's permit-aes-key-import-functions property.
permit-ecc-key-import-functions	Boolean	Optional	The value to be set as the partition's permit-ecc-key-import-functions property.
ssc-host-name	String/ Hostname	Required, if type is " ssc "	The value to be set as the partition's ssc-host-name property.
ssc-ipv4-gateway	String/ IPv4 Address	Optional	The value to be set as the partition's ssc-ipv4-gateway property.
ssc-ipv6-gateway	String/ IPv6 Address	Optional	The value to be set as the partition's ssc-ipv6-gateway property.
ssc-dns-servers	Array of String/ IPv4 or IPv6 Address	Optional	The value to be set as the partition's ssc-dns-servers property.
ssc-master-userid	String	Required, if type is " ssc "	The value to be set as the partition's ssc-master-userid property.
ssc-master-pw	String	Required, if type is " ssc "	The value to be set as the partition's ssc-master-pw property.
initial-ifl-processing-weight	Integer (1-999)	Optional	The value to be set as the partition's initial-ifl-processing-weight property.
initial-cp-processing-weight	Integer (1-999)	Optional	The value to be set as the partition's initial-cp-processing-weight property.
acceptable-status	Array of String Enum	Optional	The value to be set as the partition's acceptable-status property.
cp-absolute-processor-capping	Boolean	Optional	The value to be set as the partition's cp-absolute-processor-capping property.

Field name	Type	Rqd/Opt	Description
cp-absolute-processor-capping-value	Float	Optional	The value to be set as the partition's cp-absolute-processor-capping-value property.
cp-processing-weight-capped	Boolean	Optional	The value to be set as the partition's cp-processing-weight-capped property.
ifl-absolute-processor-capping	Boolean	Optional	The value to be set as the partition's ifl-absolute-processor-capping property.
ifl-absolute-processor-capping-value	Float	Optional	The value to be set as the partition's ifl-absolute-processor-capping-value property.
ifl-processing-weight-capped	Boolean	Optional	The value to be set as the partition's ifl-processing-weight-capped property.
maximum-cp-processing-weight	Integer	Optional	The value to be set as the partition's maximum-cp-processing-weight property.
maximum-ifl-processing-weight	Integer	Optional	The value to be set as the partition's maximum-ifl-processing-weight property.
minimum-cp-processing-weight	Integer	Optional	The value to be set as the partition's minimum-cp-processing-weight property.
minimum-ifl-processing-weight	Integer	Optional	The value to be set as the partition's minimum-ifl-processing-weight property.
processor-management-enabled	Boolean	Optional	The value to be set as the partition's processor-management-enabled property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the created partition.

Description

This operation creates a partition with the values specified on the identified CPC and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

Any properties identified as required must be included in the request body. Any properties identified as optional may be excluded from the request body; if an optional property is not found in the request body, its value will be set to its default value.

If the request body contents are valid, the partition is created on the target CPC and its properties are defined to their corresponding request body content's properties' values. If a property is omitted from the request body, its default value is used when creating the partition.

If the API user does not have action/task permission to the **New Partition** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** {cpc-id} does not identify a CPC object for which the API user has object-access permission.

If the request body contents fail to validate, a 400 (Bad Request) status code is returned. This may occur because the document fails to define a required property. This may also occur if the document fails to define a single valid partition, for instance defining a property with an invalid value (e.g. an **initial-memory** value less than zero, or a **name** that is already in use). If the status of the CPC is not valid (The

valid states are **"active"**, **"service-required"**, **"degraded"**, **"exceptions"**), 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC identified by *{cpc-id}*.
- Action/task permission to the **New Partition** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in ["Response body contents"](#) on page 188.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 91. Create Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
	8	A partition with the name specified in the request body already exists.
	15	The specified access-diagnostic-sampling value is not valid when access-basic-sampling is false .
	18	A property that is only valid for type "ssc" was provided, but the type is not "ssc" . The partition type "ssc" only allows "none" for boot-device .
	20	The type "ssc" is not supported by the targeted CPC. The property permit-ecc-key-import-functions is not supported by the targeted CPC.
	117	boot-device cannot be set to "network-adapter" , "storage-adapter" , "storage-volume" , or "iso-image" at the time of partition creation.
	118	There is an error in the fields related to the partition ID. One of: • autogenerate-partition-id is false and the partition-id specified in the request body is already in use. • autogenerate-partition-id is false and partition-id is not included in the request body. • autogenerate-partition-id is true and partition-id is included in the request body.
403 (Forbidden)	1	API user does not have action/task permission to the New Partition task.
404 (Not Found)	1	The CPC with object ID <i>{cpc-id}</i> does not exist, or the API user does not have object-access permission to it.

Table 91. Create Partition: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the CPC designated by the URI does not have a valid status. The valid states are "active" , "service-required" , "degraded" , and "exceptions" .
	2	The operation cannot be performed because the CPC designated by the request URI is currently busy performing some other operation.
	5	The operation cannot be performed because the CPC designated by the request URI is currently not enabled for DPM.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The reserve-resources value is true but resources are not available to be reserved for this partition's use.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
Create Partition: Request
POST /api/cpcs/d49a116c-d938-3b87-ad7c-444752db1216/partitions HTTP/1.1
x-api-session: v6n1aljy1tmlsjq7ki955u0s4t7qr8xabmiu0pbpbadgq7fe
content-type: application/json
content-length: 117
{
  "name": "Partition",
  "cp-processors": 3,
  "initial-memory": 1024,
  "maximum-memory": 2048,
  "processor-mode": "shared"
}
```

Figure 38. Create Partition: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/9cfd912-89cf-11e5-8092-020000000056
cache-control: no-cache
date: Fri, 13 Nov 2015 06:26:42 GMT
content-type: application/json; charset=UTF-8
content-length: 69
{
  "object-uri": "/api/partitions/9cfd912-89cf-11e5-8092-020000000056"
}
```

Figure 39. Create Partition: Response

Delete Partition

The Delete Partition operation deletes the identified partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}
```

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Description

This operation deletes the designated partition, which includes the following actions:

- The partition's HBAs are disassociated from their backing physical adapters and deleted.
- The partition's NICs are disassociated from their backing virtual switches and deleted.
- The partition's virtual functions are disassociated from their backing physical adapters and deleted.
- The partition is disassociated from the crypto adapters and crypto domains for which it was configured.
- The ISO image is deleted.
- An Inventory Change notification is emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the API user does not have action/task permission to the Delete Partition operation, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if status of either the partition or the CPC is not valid to perform the operation. A 409 (Conflict) status code is also returned if another operation targeting the partition is already underway.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Delete Partition** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 92. Delete Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action permission to the Delete Partition task.
404 (Not Found)	1	The partition with the object ID <i>{partition-id}</i> does not exist, or the API user does not have object-access permission to it.

Table 92. Delete Partition: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "stopped" or "reservation-error").
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation. The valid states are "active" , "service-required" , "degraded" , and "exceptions" .
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	110	The operation cannot be performed as the partition is a member of a Capacity Group.
	112	The operation cannot be performed as the partition is targeted in a scheduled operation.
	113	The operation cannot be performed as the partition is configured to be automatically started when its hosting CPC starts.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/partitions/9cfd912-89cf-11e5-8092-020000000056 HTTP/1.1
x-api-session: 515ad9soju9cvqsyxkcq3d65c1v2bd6e8rhz4pu2psps2jjae1f
```

Figure 40. Delete Partition: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 13 Nov 2015 06:30:12 GMT

<No response body>
```

Figure 41. Delete Partition: Response

Usage note

This is a synchronous operation and as such does not complete until the partition has been deleted. Depending on the I/O configuration associated with the partition, this operation may take a considerable amount of time to complete. API clients that are concerned about that should use the Delete Partition Asynchronously operation instead.

Delete Partition Asynchronously

The Delete Partition Asynchronously operation deletes the identified partition asynchronously.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/async-delete
```

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Response body contents

Once the delete request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body. The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 194. The **job-results** field contains null when the operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation asynchronously deletes the designated partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body contains a URI that may be queried to retrieve the status of the operation. See [“Query Job Status”](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in [“Job status and reason codes”](#) on page 194.

The deletion includes the following actions:

- The partition's HBAs are disassociated from their backing physical adapters and deleted.
- The partition's NICs are disassociated from their backing virtual switches and deleted.
- The partition's virtual functions are disassociated from their backing physical adapters and deleted.
- The partition is disassociated from the crypto adapters and crypto domains for which it was configured.
- The ISO image is deleted.
- An Inventory Change notification is emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object for which the API user has object-access permission. If the API user does not have authority to perform the Delete Partition operation, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the status of either the partition or the CPC is not valid to perform the operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Delete Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 193.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 93. Delete Partition Asynchronously: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have the required permission for this operation.
404 (Not Found)	1	The object-id in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "stopped" or "reservation-error").
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of the following states: "active" , "service-required" , "degraded" , and "exceptions" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

<i>Table 94. Delete Partition Asynchronously: Job status and reason codes</i>		
HTTP error status code	Reason code	Description
204 (No Content)	N/A	Delete completed successfully.

Table 94. Delete Partition Asynchronously: Job status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "stopped" or "reservation-error").
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of the following states: "active" , "service-required" , "degraded" , and "exceptions" .
	110	The operation cannot be performed as the partition is a member of a Capacity Group.
	112	The operation cannot be performed as the partition is targeted in a scheduled operation.
	113	The operation cannot be performed as the partition is configured to be automatically started when its hosting CPC starts.
500 (Server Error)	100	Partition delete failed.
	101	Partition delete job timed out.

Example HTTP interaction

```
POST /api/partitions/279cee22-50d3-11e7-b285-fa163ef98b8b/operations/async-delete HTTP/1.1
x-api-session: 37o1ukpoqdqwa64iggv0gjtfx6q9xsdihiquuidnvdh5s8twc9
content-type: application/json
```

Figure 42. Delete Partition Asynchronously: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/3ae9a3c8-55b3-11e7-b883-fa163e6e13bb
cache-control: no-cache
date: Tue, 20 Jun 2017 12:23:13 GMT
content-type: application/json; charset=UTF-8
content-length: 60
{
  "job-uri": "/api/jobs/3ae9a3c8-55b3-11e7-b883-fa163e6e13bb"
}
```

Figure 43. Delete Partition Asynchronously: Response

Get Partition Properties

The Get Partition Properties operation retrieves the properties of a single Partition object.

HTTP method and URI

```
GET /api/partitions/{partition-id}
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Partition object as defined in the [“Data model” on page 158](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

The `Get Partition Properties` operation returns the current values of the properties for the Partition object as defined in the [“Data model” on page 158](#).

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined in the [“Data model” on page 158](#). A 404 (Not Found) status code is returned if the **object-id** `{partition-id}` does not identify a partition object for which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object designated by `{partition-id}`

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 196](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI <code>{partition-id}</code> does not designate an existing Partition object, or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/partitions/aa245764-b64f-11e6-9ee6-42f2e9cfe851 HTTP/1.1
x-api-session: 3ikod8v41qpbypte20906jzcavwrnkg51fs8jjjyeipexgvp50
```

Figure 44. Get Partition Properties: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 03 Feb 2017 09:26:58 GMT
content-type: application/json;charset=UTF-8
content-length: 3219
{
  "acceptable-status":[
    "active"
  ],
  "access-basic-counter-set":false,
  "access-basic-sampling":false,
  "access-coprocessor-group-set":false,
  "access-crypto-activity-counter-set":false,
  "access-diagnostic-sampling":false,
  "access-extended-counter-set":false,
  "access-global-performance-data":false,
  "access-problem-state-counter-set":false,
  "auto-start":false,
  "autogenerate-partition-id":true,
  "available-features-list":[
    {
      "description":"The DPM storage management approach in which FCP and FICON storage resources
        are defined in Storage Groups, which are attached to Partitions.",
      "name":"dpm-storage-management",
      "state":true
    }
  ],
  "boot-configuration-selector":0,
  "boot-device":"none",
  "boot-ftp-host":null,
  "boot-ftp-insfile":null,
  "boot-ftp-username":null,
  "boot-iso-image-name":null,
  "boot-iso-ins-file":null,
  "boot-logical-unit-number":"",
  "boot-network-device":null,
  "boot-os-specific-parameters":"",
  "boot-record-lba":"0",
  "boot-removable-media":null,
  "boot-removable-media-type":null,
  "boot-storage-device":null,
  "boot-storage-volume":"/api/storage-groups/1bcc3530-385c-11ea-91f4-00106f23f532/storage-
    volumes/e61ad608-c7cc-11e9-9f1f-00106f0d81c9",
  "boot-timeout":60,
  "boot-world-wide-port-name":"",
  "class":"partition",
  "cp-absolute-processor-capping":false,
  "cp-absolute-processor-capping-value":1.0,
  "cp-processing-weight-capped":false,
  "cp-processors":1,
  "crypto-configuration":null,
  "current-cp-processing-weight":1,
  "current-ifl-processing-weight":1,

```

Figure 45. Get Partition Properties: Response (Part 1)

```

"degraded-adapters": [
  "/api/storage-groups/1bcc3530-385c-11ea-91f4-00106f23f532/virtual-storage-resources/
  4ce7feea-9fe9-11e8-b6db-fa163e3c2af4"
],
"description": "",
"has-unacceptable-status": true,
"hba-uris": [],
"ifl-absolute-processor-capping": false,
"ifl-absolute-processor-capping-value": 1.0,
"ifl-processing-weight-capped": false,
"ifl-processors": 0,
"initial-cp-processing-weight": 100,
"initial-ifl-processing-weight": 100,
"initial-memory": 4096,
"ipl-load-parameter": "",
"is-locked": false,
"maximum-cp-processing-weight": 999,
"maximum-ifl-processing-weight": 999,
"maximum-memory": 4096,
"minimum-cp-processing-weight": 1,
"minimum-ifl-processing-weight": 1,
"name": "AsiPartition",
"nic-uris": [
  "/api/partitions/9e8ca844-c406-11e9-9c04-00106f23f534/nics/43dc38f8-
  daa7-11e9-9546-00106f23f534"
],
"object-id": "9e8ca844-c406-11e9-9c04-00106f23f534",
"object-uri": "/api/partitions/9e8ca844-c406-11e9-9c04-00106f23f534",
"os-current-cp-processors": 1,
"os-current-ifl-processors": 0,
"os-current-memory": 4096,
"os-name": "",
"os-type": "",
"os-version": "",
"parent": "/api/cpcs/25a0b74b-e836-3597-ae18-51e6e3133120",
"partition-id": null,
"permit-aes-key-import-functions": true,
"permit-cross-partition-commands": false,
"permit-des-key-import-functions": true,
"permit-ecc-key-import-functions": true,
"processor-management-enabled": false,
"processor-mode": "shared",
"reserve-resources": false,
"reserved-memory": 0,
"secure-boot": false,
"short-name": "ASIPARTI",
"status": "degraded",
"storage-group-uris": [
  "/api/storage-groups/1bcc3530-385c-11ea-91f4-00106f23f532",
  "/api/storage-groups/8cb87f12-4ef3-11ea-824c-00106f23f532",
  "/api/storage-groups/fdf2dc56-3eac-11ea-b81b-00106f23f532"
],
"threads-per-processor": 0,
"type": "linux",
"virtual-function-uris": []
}

```

Figure 46. Get Partition Properties: Response (Part 2)

Update Partition Properties

The Update Partition Properties operation updates one or more of the writable properties of the Partition object designated by *{partition-id}*.

HTTP method and URI

POST /api/partitions/*{partition-id}*

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain one or more field names representing writable partition properties, along with the new values for those fields. The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Partition object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type. If the API user does not have action/task permission to the Partition Details operation, a 403 (Forbidden) status code is returned. If the status of the CPC hosting the partition is not in a valid state, 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition is in a transitional state ("**starting**" or "**stopping**") or if user sets **boot-device** to "**iso-image**", but there is no ISO image mounted on the partition.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 95. Update Partition Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected.
	15	access-diagnostic-sampling cannot be true when access-basic-sampling is false . boot-device setting is invalid. Since secure-boot is set to true , only boot-device "storage-volume" is allowed.
	18	A property that is only valid for type "ssc" was provided, but the type is not "ssc" . type "ssc" only allows "none" for boot-device .
	118	There is an error in the fields related to the partition ID. One of: • autogenerate-partition-id is false and the partition-id specified in the request body is already in use. • autogenerate-partition-id is false and partition-id is not included in the request body. • autogenerate-partition-id is true and partition-id is included in the request body.
403 (Forbidden)	1	API user does not have the required permission for this operation
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
	2	A URI in the request body does not designate a resource of an expected type.

Table 95. Update Partition Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	8	The operation cannot be completed because it would result in inconsistencies between the boot-device property and the related properties that identify the specific boot device for that boot-device value. For example, if boot-device is "storage-adapter", then boot-storage-device must contain a valid HBA URI. access-diagnostic-sampling cannot be set to true when access-basic-sampling is false. access-basic-sampling cannot be set to false when access-diagnostic-sampling is true. secure-boot cannot be set to true unless the type is "linux", the boot-device is "storage-volume", and the volume type is fc.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	110	Partition's processor-mode cannot be updated as the partition is a member of a Capacity Group.
	116	The reserve-resources value is true but resources are not available to be reserved for this partition's use.
	119	The value of the boot-storage-volume property does not designate a storage volume of type "boot" contained in a storage group that is attached to the partition.
	120	- Request contains boot-storage-volume property when the "dpm-storage-management" feature is disabled. "storage-volume" is provided as value for boot-device property when the "dpm-storage-management" feature is disabled. - Request contains ipl-load-parameter when the "dpm-storage-management" feature is disabled.
	121	- Request contains boot-storage-device property when the "dpm-storage-management" feature is enabled. "storage-adapter" is provided as a value for boot-device property when the "dpm-storage-management" feature is enabled. - Request contains boot-logical-unit-number when the "dpm-storage-management" feature is enabled. - Request contains boot-world-wide-port-name when the "dpm-storage-management" feature is enabled.
	122	The storage-volume provided as value for boot-storage-volume property is "incomplete" or is not mapped to any LUN.

Table 95. Update Partition Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/30d8fe00-89c3-11e5-9b53-020000000056 HTTP/1.1
x-api-session: 2b1cv9k5i0v3tfgm6uo5vggzj3e0er6zeiu7jt79tznjeqanl
content-type: application/json
content-length: 90
{
  "cp-processors":5,
  "description":"Sample partition description",
  "name":"Serv-sample"
}
```

Figure 47. Update Partition Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 13 Nov 2015 06:43:45 GMT

<No response body>
```

Figure 48. Update Partition Properties: Response

Usage note

This is a synchronous operation and as such does not complete until the partition has been updated. Depending on the I/O configuration associated with the partition, this operation may take a considerable amount of time to complete. API clients that are concerned about that should use the Update Partition Properties Asynchronously operation instead.

Update Partition Properties Asynchronously

The Update Partition Properties Asynchronously operation updates one or more of the writable properties of the Partition object designated by *{partition-id}* asynchronously.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/async-update
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain one or more field names representing writable partition properties, along with the new values for those fields. The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Response body contents

Once the update request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body. The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 204. The **job-results** field contains null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The request body object is validated against the data model for the `Partition` object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

This operation asynchronously updates the designated partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body contains a URI that may be queried to retrieve the status of the operation. See [“Query Job Status”](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in [“Job status and reason codes”](#) on page 204.

On successful execution of the asynchronous portion of this operation, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned in the asynchronous result document. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a `Partition` object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type. If the API user does not have action/task permission to the `Partition Details` operation, a 403 (Forbidden) status code is returned. If the status of the CPC hosting the partition is not in a valid state, 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition is in a transitional state (**"starting"** or **"stopping"**).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 203.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 96. Update Partition Properties Asynchronously: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

Table 97. Update Partition Properties Asynchronously: Job status and reason codes		
HTTP error status code	Reason code	Description
204 (No Content)	N/A	Update completed successfully.
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected.
	15	access-diagnostic-sampling cannot be true when access-basic-sampling is false .
	18	A property that is only valid for type "ssc" was provided, but the type is not "ssc" . The partition type "ssc" only allows "none" for the boot-device .
	118	There is an error in the fields related to the partition ID. One of: autogenerate-partition-id is false and partition-id specified in the request body is already in use. autogenerate-partition-id is false and partition-id is not included in the request body. autogenerate-partition-id is true and partition-id is included in the request body.

Table 97. Update Partition Properties Asynchronously: Job status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	8	boot-device can only be set if the boot device type ("network-adapter", "storage-adapter", "ftp", "iso-image", etc.) points to a valid URI/Hostname/ISO image. Boot device type cannot be set to null if the boot device points to the corresponding boot device type. access-diagnostic-sampling cannot be set to true when access-basic-sampling is false. access-basic-sampling cannot be set to false when access-diagnostic-sampling is true.
	110	Partition's processor-mode cannot be updated as the partition is a member of a Capacity Group.
	116	The reserve-resources boolean is true but resources are not available to be reserved for this partition's use.
	119	The value of the boot-storage-volume property does not designate a storage volume of type "boot" contained in a storage group that is attached to the partition.
	120	- Request contains boot-storage-volume property when the "dpm-storage-management" feature is disabled. "storage-volume" is provided as value for boot-device property when the "dpm-storage-management" feature is disabled. - Request contains ipl-load-parameter when the "dpm-storage-management" feature is disabled.
	121	- Request contains boot-storage-device property when the "dpm-storage-management" feature is enabled. "storage-adapter" is provided as a value for boot-device property when the "dpm-storage-management" feature is enabled. - Request contains boot-logical-unit-number when the "dpm-storage-management" feature is enabled. - Request contains boot-world-wide-port-name when the "dpm-storage-management" feature is enabled.
	122	The storage-volume provided as value for boot-storage-volume property is "incomplete" or is not mapped to any LUN.
500 (Server Error)	100	Partition update failed.
	101	Partition update job timed out.

Example HTTP interaction

```
POST /api/partitions/279cee22-50d3-11e7-b285-fa163ef98b8b/operations/async-update HTTP/1.1
x-api-session: 1wb0irgy7jn2edj7ptyqy4twit24m0b7krskctnf7h5dr91goz
content-type: application/json
content-length: 20
{
  "name": "Testname"
}
```

Figure 49. Update Partition Properties Asynchronously: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/df6b825a-55b2-11e7-8f25-fa163e6e13bb
cache-control: no-cache
date: Tue, 20 Jun 2017 12:20:39 GMT
content-type: application/json; charset=UTF-8
content-length: 60
{
  "job-uri": "/api/jobs/df6b825a-55b2-11e7-8f25-fa163e6e13bb"
}
```

Figure 50. Update Partition Properties Asynchronously: Response

Start Partition

The **Start Partition** operation allocates the physical resources required by the partition and begins its execution on the CPC by booting the partition as configured by its boot-related properties.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/start
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Depending on the **boot-device** property of the corresponding partition, the boot process might be handled by the zBootLoader component. The zBootLoader is used with partitions whose associated Support Element version is 2.15.0 or later:

- when **boot-device** is of **type "network-adapter"**, or
- when **boot-device** is of **type "storage-volume"** and **boot-storage-volume** refers to a volume of type FCP.

In case the zBootLoader fails to boot the corresponding partition, the error information provided by the zBootLoader is surfaced to users of the **Start Partition** operation via details of the corresponding **job-results** field (within the response body of the asynchronous job result):

- The **message** field contains an error message and message ID provided by the zBootLoader.
- The **bootloader-error-id** contains the error ID provided by the zBootLoader.

Notes:

- The zBootLoader might raise a console hardware message with additional information for further analysis.
- The zBootLoader is also used for partition **type "ssc"**, but in this case, the zBootLoader error information is not provided.

The set of zBootLoader error IDs are documented in the *Small Computer Systems Interface (SCSI) IPL - Machine Loader Messages*, SC28-7006.

Once the start operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body from the `Start Partition` request.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 208. The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.
bootloader-error-id	String	If the zBootLoader failed to boot the partition, this field contains the zBootLoader error ID; otherwise, this field is not present.

Description

This operation asynchronously starts the identified partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See ["Query Job Status"](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in ["Job status and reason codes"](#) on page 208.

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

If the Partition object's **boot-device** property is **"none"**, then no program is started in the new partition. However, other portions of the operation are performed and the partition is placed in the **"paused"** state.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the user does not have action/task permission to the `Start Partition` action, a 403 (Forbidden) status code is returned. If the status of the partition is not valid, 409 (Conflict) status code is returned. The valid states of the partition are **"stopped"** and **"reservation-error"**. A 409 (Conflict) status code is also returned if the CPC hosting the partition is not in a valid state, or if the CPC does not have sufficient processors, memory or adapter resources to allocate to the partition.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission for the **Start Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in “Response body contents” on page 206.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Table 98. Start Partition: Job status and reason codes		
HTTP error status code	Reason code	Description
200 (OK)	N/A	Start operation completed successfully.

Table 98. Start Partition: Job status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be one of the following states: "active" , "service-required" , "degraded" and "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	43	The operation has been canceled.
	110	There is no DHCP info or an error occurred when trying to resolve DHCP information.
	111	DHCP file error occurred.
	112	DNS lookup error has occurred.
	113	An error has occurred during network boot component/server/ configuration download.
	114	An error occurred when parsing network boot configuration, or when executing network boot program.
	114	An error occurred during removable media load.
	115	An error occurred during FTP load.
	116	The CPC does not have sufficient processor, memory, or adapter resources to allocate to the partition to perform start operation.
	117	An error occurred during initialization on the network boot device or there was an error in internal setup during network boot.
	118	The count of the partitions in active state has reached its maximum.
	119	The specified device does not contain a bootable dump program.
	119	DHCP lease failed on device or there was a DHCP lease internal error.
	120	The partition configuration is not valid for a Secure Service Container partition. The specific error reason is returned as additional error text. Possible error conditions are: - Secure Service Container Management NIC is missing (there needs to be at least one Secure Service Container Management NIC). - At least one NIC was configured in static IPv4 mode, but no ssc-ipv4-gateway was provided.
	122	The operation cannot be performed because the boot-storage-volume property does not designate a storage volume that is fulfilled.
	125	A timeout occurred during partition boot. Check the integrity of the boot source, and the boot-timeout property.
	130	An error occurred during partition boot via the zBootLoader, caused by invalid settings or configuration related problems.

Table 98. Start Partition: Job status and reason codes (continued)

HTTP error status code	Reason code	Description
500 (Server Error)	100	Partition start failed.
	101	Partition start job timed out.
	130	An internal error occurred during partition boot via the zBootLoader.
	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/start HTTP/1.1
x-api-session: pd0nrulei0qsa1mwlpaw7cmq26rnsdcdhtp4w4m9gzse7gybg
content-type: application/json
```

Figure 51. Start Partition: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/913b0490-d537-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
  "job-uri": "/api/jobs/913b0490-d537-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 52. Start Partition: Response

Stop Partition

The Stop Partition operation stops the Partition object designated by *{partition-id}*.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/stop
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the stop request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job

completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 212](#). The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

Stop Partition is an orderly process for terminating a partition.

Stopping a partition includes:

- Stopping the execution of (logical) processors associated with the partition.
- Unloading the partition's operating system.
- Freeing non-reserved CPC processor, memory, and adapter resources so that those resources are available for use by other partitions.

After the partition is stopped, the partition is no longer operational.

The operation asynchronously stops the identified partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, as described in [“Job status and reason codes” on page 212](#).

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the user does not have action/task permission to the Stop Partition action, a 403 (Forbidden) status code is returned. If the partition is not in a valid state to perform the stop operation, 409 (Conflict) status code is returned. The valid states to perform Stop Partition are **"active"**, **"degraded"**, **"paused"**, and **"terminated"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **Stop Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 210](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Table 99. Stop Partition: Job status and reason codes		
HTTP error status code	Reason code	Description
200 (OK)	N/A	Stop completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "active" , "paused" , or "terminated") .
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	43	The operation has been canceled.
500 (Server Error)	100	Partition stop failed.
	101	Partition stop job timed out.
	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/stop HTTP/1.1
x-api-session: 44p85d6k4nve3pdag1bdxyohewkhj6hwnqo9jv0czp226h82p
content-type: application/json
```

Figure 53. Stop Partition: Request

```

202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/eebbef94-d537-11e5-9e2e-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 05:33:10 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
  "job-uri":"/api/jobs/eebbef94-d537-11e5-9e2e-5ef3fcb21ee8"
}

```

Figure 54. Stop Partition: Response

Dump Partition

The Dump Partition operation loads a standalone dump program from a designated SCSI device. This operation is not supported when the **"dpm-storage-management"** feature is enabled on the target Partition object.

HTTP method and URI

POST /api/partitions/{*partition-id*}/operations/scsi-dump

In this request, the URI variable {*partition-id*} is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
dump-load-hba-uri	String/ URI	Required	The URI of the HBA associated with the partition that provides access to the storage-area network containing the SCSI device from which the dump program is loaded.
dump-world-wide-port-name	String (16)	Required	The worldwide port name (WWPN) of the target storage controller that contains the SCSI device from which the dump program is loaded, in hexadecimal.
dump-logical-unit-number	String (1-16)	Required	The hexadecimal logical unit number (LUN) that identifies the SCSI device from which the dump program is loaded.
dump-configuration-selector	Integer (0-30)	Optional	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system dump program to be loaded. Whether and how this parameter is used to determine boot parameters depends on the dump program and its boot process. Default: 0, which indicates that the dump program's default boot configuration should be used.
dump-os-specific-parameters	String	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Default: an empty string.

Field name	Type	Rqd/Opt	Description
dump-record-lba	String (1-16)	Optional	Specifies the logical block number of the anchor point for locating the operating system dump program on the SCSI disk from which the dump program is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. Default: 0, identifying the first block on the device.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in “Job status and reason codes” on page 215. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation loads a standalone dump program into the partition and begins its execution. It does so in such a way that the existing contents of the partition's memory are not overwritten so that the dump program can dump those contents.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See “[Query Job Status](#)” on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in “[Job status and reason codes](#)” on page 215.

A 404 (Not Found) status code is returned if the object ID *{partition-id}* does not identify a Partition object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type.

If the user does not have action/task permission to the **Dump Partition** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's status is not valid to perform the operation. The valid partition states are **"active"**, **"degraded"**, **"paused"**, or **"terminated"**. A 409 (Conflict) is also returned when there is no storage controller at the WWPN specified, when the storage controller does not have a device with the specified LUN, or if the specified device does not contain a bootable dump program. If the partition has the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **Dump Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 214](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 100. Dump Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate a resource of an expected type
409 (Conflict)	12	The operation is not supported when the "dpm-storage-management" feature is enabled on the partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

Table 101. Dump Partition: Job status and reason codes		
HTTP error status code	Reason code	Description
200 (OK)	N/A	Operation completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "active" , "paused" , or "terminated").
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	119	The specified device does not contain a bootable dump program.

Table 101. Dump Partition: Job status and reason codes (continued)

HTTP error status code	Reason code	Description
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/scsi-dump HTTP/1.1
x-api-session: 12a74m6cfuaiebwuceikrhaqq8bxx1a6kx16xoihyfel8x5d
content-type: application/json
content-length: 205
{
  "dump-load-hba-uri":"/api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/hbas/bab7e3f8-
    d53a-11e5-a366-42f2e9cfe851",
  "dump-logical-unit-number":"00000",
  "dump-world-wide-port-name":"AFFCB01FF21BCAFA"
}
```

Figure 55. Dump Partition: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 06:01:45 GMT
content-type: application/json; charset=UTF-8
content-length: 60
{
  "job-uri":"/api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 56. Dump Partition: Response

Start Dump Program

The Start Dump Program operation loads a standalone dump program from a designated external location such as a storage disk.

HTTP method and URI

POST /api/partitions/{*partition-id*}/operations/start-dump-program

In this request, the URI variable {*partition-id*} is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
dump-program-info	Object	Required	An object which identifies the location of the dump program and any additional parameters for loading or executing the dump program. The dump-program-type field indicates the type dump program and the type of object contained in this field.

Field name	Type	Rqd/Opt	Description
dump-program-type	String Enum	Required	Indicates the type of dump program and identifies the type of object in dump-program-info. Valid values: • "storage" - The dump program resides on a storage volume. The dump-program-info field contains a storage-volume-dump-program-info object as described in the next table.

The storage-volume-dump-program-info object contains the following fields:

Field name	Type	Rqd/Opt	Description
storage-volume-uri	String/ URI	Required	Specifies the storage volume that shall be used to load the dump program. The value has to point to a valid URI of a fulfilled storage volume contained in a storage group attached to the partition.
dump-configuration-selector	Integer (0-30)	Optional	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system dump program to be loaded. Whether and how this parameter is used to determine boot parameters depends on the dump program and its boot process. Not valid when storage-volume-uri points to a FICON volume. Default: 0 which indicates that the dump program's default boot configuration should be used.
dump-load-parameters	String (0-8)	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Not valid when storage-volume-uri points to a FICON volume. Default : an empty string
dump-os-specific-parameters	String (0-256)	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Default : an empty string

Field name	Type	Rqd/Opt	Description
dump-record-lba	String (1-16)	Optional	Specifies the logical block number of the anchor point for locating the operating system dump program on the SCSI disk from which the dump program is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. Not valid when storage-volume-uri points to a FICON volume. Default : 0 (identifying the first block on the device)
timeout	Integer (60-600)	Optional	The time in seconds that is waited before the load of the dump program is aborted. Valid only when storage-volume-uri points to a FICON volume. Default: 60

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the **status** of the job is "**complete**", the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 220](#). The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation loads a standalone dump program into the partition and begins its execution. It does so in a special way that the existing contents of the partition's memory are not overwritten so that the dump program can dump those contents.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in [“Job status and reason codes” on page 220](#).

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the URI field in the **"dump-program-info"** does not designate a valid object to which the user has object-access permission.

If the user does not have authority to perform the Dump Partition action, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's status is not valid to perform the operation. The valid partition states are **"active"**, **"degraded"**, **"paused"**, or **"terminated"**.

If the partition does not have the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission for the **Dump Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in ["Response body contents"](#) on page 218.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The operation is not supported when the "dpm-storage-management" feature is not enabled on the partition.
	2	A URI in the request body does not designate a storage volume in a storage group attached to the partition.
409 (Conflict)	13	The operation is not supported when the "dpm-storage-management" feature is not enabled on the partition.
	122	The operation cannot be performed because the "dump-program-info" contains a "storage-volume-uri" property that does not designate a storage volume that is fulfilled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, "Invoking API operations,"](#) on page 43.

Job status and reason codes

Table 102. Start Dump Program: Job status and reason codes		
HTTP error status code	Reason code	Description
200 (OK)	N/A	Start operation completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "active" , "paused" or "terminated" .)
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states : "active" , "service-required" , "degraded" and "exceptions" .)
	119	The specified device does not contain a bootable dump program.
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/a5987806-9af5-11e8-86f5-00106f0ddbc9/operations/start-dump-program HTTP/1.1
x-api-session: 5o0m316dydfq37m0uo06mqyzg0bf06yq9t8qm4j80kyx18cnc
content-type: application/json
content-length: 222
{
  "dump-program-info":{
    "dump-load-parameters":"ABCD",
    "storage-volume-uri":"/api/storage-groups/9e5d850c-9fc5-11e8-8c3e-00106f0ddbc9/
    storage-volumes/9e765546-9fc5-11e8-8c3e-00106f0ddbc9"
  },
  "dump-program-type":"storage"
}
```

Figure 57. Start Dump Program: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/8cbf1be8-a057-11e8-a48e-00106f0d84e1
cache-control: no-cache
date: Wed, 15 Aug 2018 06:50:51 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
  "job-uri":"/api/jobs/8cbf1be8-a057-11e8-a48e-00106f0d84e1"
}
```

Figure 58. Start Dump Program: Response

Perform PSW Restart

The Perform PSW Restart operation restarts the first available processor of the Partition object designated by *{partition-id}*.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/psw-restart
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 222](#). The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

Information about PSW Restart can be found on the console help system.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, as described in [“Job status and reason codes” on page 222](#).

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object for which the API user has object-access permission. If the user does not have action/task permission to the **PSW Restart** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's **status** is not valid to perform this operation. The valid partition states are **"active"**, **"paused"** or **"terminated"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **PSW Restart** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 221](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to it.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Table 103. Perform PSW Restart: Job status and reason codes		
HTTP error status code	Reason code	Description
200 (OK)	N/A	Operation completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be "active" , "paused" , or "terminated").
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/operations/psw-restart
x-api-session: 65aw2jahugn1wop51hsq0c6aldkx773dz9ulirrvvg2z853m4u
content-type: application/json
```

Figure 59. Perform PSW Restart: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 06:01:45 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
  "job-uri": "/api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 60. Perform PSW Restart: Response

Create Virtual Function

The Create Virtual Function operation creates a virtual function for the partition with the given identifier.

HTTP method and URI

POST /api/partitions/{*partition-id*}/virtual-functions

In this request, the URI variable {*partition-id*} is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the virtual function's name property.
description	String (0-1024)	Optional	The value to be set as the virtual function's description property.
adapter-uri	String/URI	Required	The URI of the physical Accelerator adapter which will back the new virtual function.
device-number	String	Optional	The value to be set as the virtual function's device-number property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/URI	The element-uri property of the created virtual function object.

Description

This operation creates a virtual function for the identified partition and then returns the URI of the created object. Upon success, the response includes a **Location** header that provides the URI of the created virtual function object. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing partition or designates a partition for which the API user does not have object-access permission. If the API user doesn't have action/task permission to the **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by {*partition-id*}.
- Object-access permission to the Accelerator adapter designated by the request body.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 223.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 104. Create Virtual Function: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	The virtual function name provided by the user is already in use by another virtual function of the partition, or the provided device-number is already in use by another Virtual Function element or by a NIC element of type "roce" of the partition.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	The adapter-uri in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID { <i>partition-id</i> } was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: " active ", " service-required ", " degraded ", or " exceptions ".)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The partition does not have sufficient resources to perform this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions HTTP/1.1
x-api-session: 63ba4ynuscmvljkablyprvm2ajhr67pa9b9jon5fz4k5jlt7gw
content-type: application/json
content-length: 101
{
  "adapter-uri":"/api/adapters/f36bf9ec-974f-11e5-bfaa-020000000192",
  "name":"New Virtual Function"
}
```

Figure 61. Create Virtual Function: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
97e6-11e5-9e1e-020000000192
cache-control: no-cache
date: Tue, 01 Dec 2015 04:45:49 GMT
content-type: application/json;charset=UTF-8
content-length: 124
{
  "element-uri":"/api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/
621c6430-97e6-11e5-9e1e-020000000192"
}
```

Figure 62. Create Virtual Function: Response

Delete Virtual Function

The Delete Virtual Function operation removes an existing virtual function designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{virtual-function-id}	String	Element ID of the virtual function object.

Description

This operation deletes the specified virtual function for the identified partition. Upon success, an Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing partition or designates a partition for which the API user does not have object-access permission. If the API user does not have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional state ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 105. Delete Virtual Function: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission
	5	The request URI does not designate an existing virtual function element of an existing partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
97e6-11e5-9e1e-020000000192 HTTP/1.1
x-api-session: pz3qrhjpvhw10jsfypo8m8qyca10np08wgq8zmdifixd9r629
```

Figure 63. Delete Virtual Function: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 04:52:01 GMT
```

Figure 64. Delete Virtual Function: Response

Get Virtual Function Properties

The Get Virtual Function Properties operation retrieves the properties of a single virtual function that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{virtual-function-id}	String	Element ID of the virtual function for which properties are to be obtained.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the virtual function object.

Description

This operation returns the current properties of the virtual function object that is specified by the request URI. On successful execution, all of the current properties as defined by the [“Data model - Virtual Function element object”](#) on [page 175](#) are provided in the response body and HTTP status code 200 (OK) is returned.

A 404 (Not Found) status code is returned if the request URI does not designate an existing virtual function element of an existing partition, or if the API user does not have object-access permission to that partition.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the partition.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on [page 227](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 106. Get Virtual Function Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission.
	5	The request URI does not designate an existing virtual function element of an existing partition.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb/virtual-functions/
7e89bce4-4479-11e7-9af6-00106f0d81cb HTTP/1.1
x-api-session: 3yg30uqxt7k97dxkei8rxn8mhbvngwka6nq9ftuxk3qhaka69f
```

Figure 65. Get Virtual Function Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 31 May 2017 09:44:55 GMT
content-type: application/json;charset=UTF-8
content-length: 399
{
  "adapter-uri": "/api/adapters/d7077406-f839-11e6-af0f-00106f0d81cb",
  "class": "virtual-function",
  "description": "",
  "device-number": "0001",
  "element-id": "7e89bce4-4479-11e7-9af6-00106f0d81cb",
  "element-uri": "/api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb/virtual-functions/
7e89bce4-4479-11e7-9af6-00106f0d81cb",
  "fid": null,
  "name": "ttt",
  "parent": "/api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb"
}
```

Figure 66. Get Virtual Function Properties: Response

Update Virtual Function Properties

The Update Virtual Function Properties operation modifies an existing virtual function that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
```

URI variables:

Name	Type	Description
<i>{partition-id}</i>	String	Object ID of the Partition object.
<i>{virtual-function-id}</i>	String	Element ID of the virtual function.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation modifies an existing virtual function specified by the request URI.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing virtual function element of an existing partition, or if the API user does not have object-access permission to that partition. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Partition Details** task.
- When updating **adapter-uri**, object-access permission to the adapter identified in that URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 107. Update Virtual Function Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “ Common request validation reason codes ” on page 50 for a list of the possible reason codes.
	8	The virtual function name provided by the user is already in use by another virtual function of the partition, or the provided device-number is already in use by another Virtual Function element or by a NIC element of type "roce" of the partition.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission.
	2	The adapter-uri in the request body does not designated an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	5	The request URI does not designate an existing virtual function element of an existing partition.
409 (Conflict)	1	Partition status is not valid to perform the operation
	2	Partition object with ID { <i>partition-id</i> } was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: " active ", " service-required ", " degraded " or " exceptions ".)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.

Table 107. Update Virtual Function Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
97e6-11e5-9e1e-020000000192 HTTP/1.1
x-api-session: 1chtesqt42ehx99ayericoo6m6gz1gt85wedt8piouotf6doyh
content-type: application/json
content-length: 46
{
  "name": "Virtual Function Name after update"
}
```

Figure 67. Update Virtual Function Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 04:50:36 GMT
<No response body>
```

Figure 68. Update Virtual Function Properties: Response

Create NIC

The Create NIC operation creates a NIC for the partition with the given identifier.

HTTP method and URI

```
POST /api/partitions/{partition-id}/nics
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the NIC's name property.
description	String (0-1024)	Optional	The value to be set as the NIC's description property.

Field name	Type	Rqd/Opt	Description
network-adapter-port-uri	String/ URI	Required, if the adapter type is " roce " or " cna "	The value to be set as the NIC's network-adapter-port-uri property. Required if the type of the adapter containing the port referenced by the network-adapter-port-uri field is " roce " or " cna ".
virtual-switch-uri	String/ URI	Required, if the adapter type is " osd " or " iqd "	The value to be set as the NIC's virtual-switch-uri property. Required if the type of the adapter containing the port referenced by the virtual-switch-uri field is " osd " or " iqd ".
device-number	String (4)	Optional	The value to be set as the NIC's device-number property. Device number for this NIC. If not provided, a device-number is auto-generated.
ssc-management-nic	Boolean	Optional	The value to be set as the NIC's ssc-management-nic property. Cannot be set to true when the partition's type is not " ssc " or the type of the adapter referenced by the network-adapter-port-uri is " roce " or " cna ". Default: false
function-number	Integer	Optional	The value to be set as the NIC's function-number property.
function-range	Integer (1-128)	Optional	The value to set as the NIC's function-range property.
ssc-ip-address-type	String Enum	Required if ssc-management-nic is true	The value to be set as the NIC's ssc-ip-address-type property. Cannot be set when the partition's type is not " ssc ".
ssc-ip-address	String/ IPv4 Address or String/ IPv6 Address	Required if ssc-ip-address-type is " ipv4 " or " ipv6 "	The value to be set as the NIC's ssc-ip-address property. Cannot be set when the partition's type is not " ssc ".
vlan-id	Integer	Optional	The value to be set as the NIC's vlan-id property.
ssc-mask-prefix	String	Required if ssc-ip-address-type is " ipv4 " or " ipv6 "	The value to be set as the NIC's ssc-mask-prefix . Cannot be set when the partition's type is not " ssc ".
mac-address	String	Optional	The value to be set as the NIC's mac-address property.
vlan-type	String Enum	Required if vlan-id is provided	The value to be set as the NIC's vlan-type property.

Response body contents

On successful completion, the response body contains the URI of the created NIC object.

Field name	Type	Description
element-uri	String/ URI	The element-uri property of the created NIC object.

Description

This operation creates a NIC for the identified partition and then returns the URI of the created object. Upon success, the response includes a **Location** header that provides the URI of the created NIC object. An Inventory Change notification is emitted asynchronously to this operation. The request identifies the NIC's backing adapter by specifying either a network port URI or a virtual switch URI.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state , a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the backing Adapter object of the Virtual Switch object or to the RoCE adapter specified in the request body.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 231](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 108. Create NIC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	5	For partitions that are not of type "ssc" : - The user specified a valid value for the vlan-id property, but did not specify a value for the vlan-type property or it was null. - The user did not specify a value for the vlan-id property, but specified the value for the vlan-type property as "enforced".
	7	The locally administered bit in the value specified for the mac-address property is invalid or the ssc-management-nic property was set to true, but the type of the NIC element is not "iqd" or "osd" .
	8	For NIC elements of type "roce" or "cna" , the NIC name provided by the user is already in use by another NIC of the partition, or the provided device-number is already in use by another NIC element of type "roce" or "cna" or by a Virtual Function element of the partition. For NIC elements of type "iqd" or "osd" , the NIC name provided by the user is already in use by another NIC of the partition, or the provided device-number is already in use by another NIC element of type "iqd" or "osd" or by an HBA element, or the mac-address provided by the user is already in use by another NIC in any of the partitions in the CPC.
	15	ssc-management-nic was set to true , but no value for ssc-ip-address-type was provided. ssc-ip-address-type was set to "ipv4" or "ipv6" , but no value for ssc-ip-address or ssc-mask-prefix was provided. For partitions that are not of type "ssc" : - The user specified a valid value for the vlan-id property, but vlan-type was null. - The user specified the value for the vlan-id property as null, but specified the value of the vlan-type property as "enforced".
	18	For partitions of type "ssc" : - A non-null value was specified for the vlan-type property. - A value for the mac-address property was specified for the NIC element of type "roce" or "cna". For partitions of other types: - A value for the properties mac-address, vlan-id, or vlan-type was specified for the NIC element of type "roce" or "cna".
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Table 108. Create NIC: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
	2	The object ID in the Network Port URI in request body field network-adapter-port-uri does not designate an existing Adapter object, or the API user does not have object-access permission to that adapter, or the object ID in the virtual switch URI in the request body field virtual-switch-uri does not designate an existing Virtual Switch object, or the API user does not have object-access permission to that virtual switch.
	6	The element ID in the Network Port URI in request body field network-adapter-port-uri does not designate an existing adapter port of the adapter.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	8	The property ssc-management-nic was set to true , but the partition's type is not "ssc" or a function-number for a virtual function was provided, but no physical function was defined yet (on the adapter designated by network-adapter-port-uri).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The partition does not have sufficient resources to perform this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics HTTP/1.1
x-api-session: 35yero3ati5fholesprwebpbn3ktukx59cucm4tt4c86m6n9id
content-type: application/json
content-length: 100
{
  "name": "Nic1",
  "virtual-switch-uri": "/api/virtual-switches/0c797342-9750-11e5-bfaa-020000000192"
}
```

Figure 69. Create NIC: Request


```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-11e5-9d1f-020000000192
cache-control: no-cache
date: Tue, 01 Dec 2015 05:03:57 GMT
content-type: application/json;charset=UTF-8
content-length: 123
{
  "element-uri":"/api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-11e5-9d1f-020000000192"
}
```

Figure 70. Create NIC: Response

Delete NIC

The Delete NIC operation deletes an existing NIC objects that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Description

This operation deletes the specified NIC. Upon success, an Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition. If the API user doesn't have action/task permission to **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by {partition-id}.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 109. Delete NIC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The request URI does not designate an existing NIC of an existing partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	8	NIC cannot be deleted as it is set as the partition's boot device.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	100	NIC cannot be deleted as it is the last SSC Management NIC (ssc-management-nic is true) on an active partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-
11e5-9d1f-020000000192 HTTP/1.1
x-api-session: 4qhwa0j0lyleh8157e0z8znsnpghelqnatcpe5pu5cjk69qeg
```

Figure 71. Delete NIC: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 05:21:30 GMT
<No response body>
```

Figure 72. Delete NIC: Response

Get NIC Properties

The Get NIC Properties operation retrieves the properties of a single NIC object that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the NIC object as defined in the “Data model - NIC element object” on page 176. Field names and data types in the JSON object are the same as the property names and data types defined in the “Data model” on page 158.

Description

This operation returns the current properties for the NIC object that is specified by the request URI.

On successful execution, all of the current properties as defined by the “Data model - NIC element object” on page 176 are provided in the response body and HTTP status code 200 (OK) is returned.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object designated by {partition-id}.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 237.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 110. Get NIC Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object-id in the URI {partition-id} does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The request URI does not designate an existing NIC of an existing partition.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851/nics/9ff431cc-e92d-11e6-9563-42f2e9cfe851 HTTP/1.1
x-api-session: 14ort688m7f2j5yi4tp1eedxo78x7d0ko7qrm7evk5h28p7bzc
```

Figure 73. Get NIC Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Feb 2017 09:11:27 GMT
content-type: application/json; charset=UTF-8
content-length: 551
{
  "class": "nic",
  "description": "",
  "device-number": "0004",
  "element-id": "9ff431cc-e92d-11e6-9563-42f2e9cfe851",
  "element-uri": "/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851/nics/9ff431cc-e92d-11e6-9563-42f2e9cfe851",
  "mac-address": "02:ff:ec:00:21:07",
  "name": "ttt",
  "parent": "/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851",
  "ssc-ip-address": "fc01:1234:1234:1234::3",
  "ssc-ip-address-type": "ipv6",
  "ssc-management-nic": true,
  "ssc-mask-prefix": "/24",
  "type": "osd",
  "virtual-switch-uri": "/api/virtual-switches/2b0d93e8-b64d-11e6-99ee-42f2e9cfe851",
  "vlan-id": 10,
  "vlan-type": "enforced"
}
```

Figure 74. Get NIC Properties: Response

Update NIC Properties

The Update NIC Properties operation updates the properties of a single NIC object that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Request body contents

The request body is expected to contain one or more field names representing writable NIC properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

This operation modifies the NIC properties for the NIC specified by the request URI.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

If the API user doesn't have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.
- When updating **network-adapter-port-uri**, object-access permission to the adapter identified in that URI.
- When updating **virtual-switch-uri**, object-access permission to the backing Adapter object of the Virtual Switch object identified in that URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 111. Update NIC Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	5	For partitions that are not of type "ssc" : - The user specified a valid value for the property vlan-id, but did not specify a value for the property vlan-type or was null. - The user did not specify a value for the property vlan-id or was null, but specified the value for the property vlan-type as "enforced".
	7	The locally administered bit in the value specified for the mac-address property is invalid. or the ssc-management-nic was set to true, but the type of the NIC element is not "iqd" or "osd" .
	8	For NIC elements of type "roce" or "cna" , the NIC name provided by the user is already in use by another NIC of the partition, or the provided device-number is already in use by another NIC element of type "roce" or "cna" or by a Virtual Function element of the partition. For NIC elements of type "iqd" or "osd" , the NIC name provided by the user is already in use by another NIC of the partition, or the provided device-number is already in use by another NIC element of type "iqd" or "osd" or by an HBA element, or the mac-address provided by the user is already in use by another NIC in any of the partitions in the CPC.
	15	ssc-management-nic was set to true , but no value for ssc-ip-address-type was provided. ssc-ip-address-type was set to "ipv4" or "ipv6" , but no value for ssc-ip-address or ssc-mask-prefix was provided. For partitions that are not of type "ssc" : The user specified a value for both vlan-id and vlan-type properties, but either or both of the properties values are invalid.
403 (Forbidden)	18	For partitions of type "ssc" : - A non-null value was specified for the vlan-type property. - A value for the mac-address property was specified for the NIC element of type "roce". For partitions of other types: - A value for the properties mac-address, vlan-id, or vlan-type was specified for the NIC element of type "roce".
	1	The API user does not have the required permission for this operation.

Table 111. Update NIC Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object-id in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
	2	The object ID in the Network Port URI in request body field network-adapter-port-uri does not designate an existing Adapter object, or the API user does not have object-access permission to that adapter, or the object ID in the virtual switch URI in request body field virtual-switch-uri does not designate an existing Virtual Switch object, or the API user does not have object-access permission to that virtual switch.
	5	The request URI does not designate an existing NIC of an existing partition.
	6	The element ID in the Network Port URI in request body field network-adapter-port-uri does not designate an existing adapter port of the adapter.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: " active ", " service-required ", " degraded ", or " exceptions ".)
	8	The request cannot be processed because the update to virtual-switch-uri/network-adapter-port-uri attempts to change the switch/port type. ssc-management-nic was set to true, but the partition's type is not "ssc". For partitions that are not of type "ssc": • The user specified a valid value for vlan-id but it is not compatible with the current value of vlan-type. • The user specified a valid value for vlan-type but it is not compatible with the current value of vlan-id. • A function-number for a virtual function was provided, but no physical function was defined yet (on the adapter designated by network-adapter-port-uri).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	18	For partitions of type "ssc": • A non-null value was specified for the vlan-type property. • A value for the mac-address property was specified for the NIC element of type "roce" or "cna". For partitions of other types: • A value for the properties mac-address, vlan-id, or vlan-type was specified for the NIC element of type "roce" or "cna".

Table 111. Update NIC Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-11e5-9d1f-020000000192 HTTP/1.1
x-api-session: 3nho0b70boh2ei3wjryvayxgxxnqo7wtqyj3lsifu4kfok2nf
content-type: application/json
content-length: 25
{
  "name": "NicUpdateName"
}
```

Figure 75. Update NIC Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 05:17:57 GMT
<No response body>
```

Figure 76. Update NIC Properties: Response

Increase Crypto Configuration

The Increase Crypto Configuration operation can be used to add more elements to an existing (empty or non-empty) crypto configuration.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/increase-crypto-configuration
```

In this request, the URI variable *{partition-id}* is the object ID of the target partition.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
crypto-adapter-uris	Array of String/ URI	Optional	Array of URIs listing crypto adapters that should be added to the crypto configuration of this partition.
crypto-domain-configurations	Array of crypto-domain-configuration objects	Optional	Array of crypto-domain-configuration objects (see Table 87 on page 175) that should be added to the crypto configuration of this partition.

Description

This operation adds the specified adapters and/or domain configurations to the crypto configuration of the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned. A 400 (Bad Request) status code is returned when no lists are provided or both the lists are empty.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition object designated by *{partition-id}*.
- Object-access permission to all crypto adapter objects specified in **crypto-adapter-uris**.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 112. Increase Crypto Configuration: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	101	No lists provided, or both provided lists are empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id <i>{partition-id}</i> does not exist, or the API user does not have object-access permission to it.
	2	A URI in the request body does not designated an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.

Table 112. Increase Crypto Configuration: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with "control-usage" .
	112	The resulting crypto configuration contains at least one domain index with "control-usage" which is already configured for "control-usage" by another partition.
	113	The list crypto-adapter-uris contains an adapter that is already part of the existing crypto configuration.
	114	The list crypto-domain-configurations contains a domain index which is already part of the existing crypto configuration.
	119	The resulting crypto configuration is invalid because it does not contain at least one adapter URI.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/increase-crypto-configuration HTTP/1.1
x-api-session: 4zzbvb5b1f5i2huscdbya9c3jpp5iwoqv1jpp5p5qal2vvmpwml
content-type: application/json
content-length: 166
{
  "crypto-adapter-uris": [
    "/api/adapters/cd8d52b2-d614-11e5-93bf-42f2e9cfe851"
  ],
  "crypto-domain-configurations": [
    {
      "access-mode": "control-usage",
      "domain-index": 1
    }
  ]
}
```

Figure 77. Increase Crypto Configuration: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 03 Mar 2016 05:28:51 GMT

<No response body>
```

Figure 78. Increase Crypto Configurations: Response

Change Crypto Domain Configuration

The Change Crypto Domain Configuration operation can be used to change the configuration of a single crypto domain that is already configured.

HTTP method and URI

POST /api/partitions/{*partition-id*}/operations/change-crypto-domain-configuration

In this request, the URI variable {*partition-id*} is the object ID of the partition.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
domain-index	Integer	Required	Index of the domain to be changed. See Table 87 on page 175 .
access-mode	String Enum	Required	The new value of the access-mode property of the crypto domain configuration identified by the domain-index . See Table 87 on page 175 .

The structure of crypto domain configuration objects is described in [“crypto-configuration object properties” on page 174](#).

Description

This operation changes the access mode for a crypto domain configuration that is currently included in the crypto configuration of the partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI {*partition-id*} does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by {*partition-id*}.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 113. Change Crypto Domain Configuration: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id { <i>partition-id</i> } does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with "control-usage" .
	112	The resulting crypto configuration contains at least one domain index with "control-usage" which is already configured for "control-usage" by another partition.
503 (Service Unavailable)	115	The index used in the request is not part of the crypto configuration of the targeted partition.
	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/change-crypto-  
domain-configuration HTTP/1.1  
x-api-session: 4eiaktj22cbpeiya9yxax1af21u0lwyod1l9iicrhy6egy3trp  
content-type: application/json  
content-length: 51  
{  
  "access-mode": "control-usage",  
  "domain-index": 1  
}
```

Figure 79. Change Crypto Domain Configuration: Request

```
204 No Content  
server: zSeries management console API web server / 2.0  
cache-control: no-cache  
date: Thu, 03 Mar 2016 06:01:20 GMT  
  
<No response body>
```

Figure 80. Change Crypto Domain Configuration: Response

Decrease Crypto Configuration

The Decrease Crypto Configuration operation can be used to remove some or all elements of an existing (non-empty) crypto configuration.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/decrease-crypto-configuration
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
crypto-adapter-uris	Array of String/ URI	Optional	Array of URIs listing crypto adapters that should be removed from the crypto configuration of this partition.
crypto-domain-indexes	Array of Integer	Optional	Array of integers, listing all crypto domain indexes that should be removed from the crypto configuration of this partition.

Description

This operation removes the specified adapters and/or domain configurations from the crypto configuration of the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), or if the CPC is not in a valid

state, a 409 (Conflict) status code is returned. A 400 (Bad Request) status code is returned if no lists were provided or both the lists were empty.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to all crypto adapter objects specified in **crypto-adapter-uris**.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 114. Decrease Crypto Configuration: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	101	No lists provided, or both provided lists are empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id <i>{partition-id}</i> does not exist, or the API user does not have object-access permission to it.
	2	A URI in request body field crypto-adapter-uris does not designate an existing crypto adapter, or the API user does not have object-access permission to it.

Table 114. Decrease Crypto Configuration: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with "control-usage" .
	112	The resulting crypto configuration contains at least one domain index with "control-usage" which is already configured for "control-usage" by another partition.
	115	The index used in the request is not part of the crypto configuration of the targeted partition.
	117	One or more of the provided adapter URIs are not part of the configuration of the targeted partition.
	119	The resulting crypto configuration is invalid because it contains a domain configuration but does not contain any crypto adapters.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/decrease-crypto-configuration HTTP/1.1
x-api-session: 6a10zfk8sivht7i6bd0xao4byaklto8a10de8fru67wpdn0coz
content-type: application/json
content-length: 109
{
  "crypto-adapter-uris": [
    "/api/adapters/cd8d52b2-d614-11e5-93bf-42f2e9cfe851"
  ],
  "crypto-domain-indexes": [
    1
  ]
}
```

Figure 81. Decrease Crypto Configuration: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 03 Mar 2016 05:32:46 GMT

<No response body>
```

Figure 82. Decrease Crypto Configuration: Response

Zeroize Crypto Domain

The Zeroize Crypto Domain operation clears a domain configured for **"control-usage"** on a specific crypto adapter for the given partition.

HTTP method and URI

POST /api/partitions/{partition-id}/operations/zeroize-crypto-domain

In this request, the URI variable *{partition-id}* is the **object-id** of the Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
crypto-adapter-uri	String/ URI	Required	The canonical URI path of the crypto adapter containing the domain to be zeroized.
domain-index	Integer	Required	The index of the domain to be zeroized.

Description

This operation clears the cryptographic keys and non-compliance mode settings within the given domain (which must have an access-mode of **"control-usage"**) of the given crypto adapter.

If the API user does not have action/task permission to the **Zeroize Crypto Domain (API only)** task, a 403 (Forbidden) status code is returned. A 404 (Not found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the **crypto-adapter-uri** field does not identify a crypto adapter object for which the API user has object-access permission.

A 409 (Conflict) status code is returned if the status of the CPC, partition, or adapter is not valid to perform the operation.

Both the adapter identified by **crypto-adapter-uri** and the domain identified by **domain-index** must be part of the crypto configuration of the partition. The identified domain must also be configured for access-mode **"control-usage"**. If any of these preconditions is not met, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

When this operation completes with HTTP status code 204 (No Content), the domain was successfully zeroized on the adapter.

When this operation completes with HTTP status code 409 (Conflict) with reason code 124, the clearing of keys and settings was initiated, but it might or might not have completed. If the adapter status is not **"active"** after receiving HTTP status code 409 (Conflict) with reason code 124, it is recommended to repeat this operation as soon as the status of the adapter becomes **"active"** again.

Although this operation is accepted for all kinds of crypto adapters, it has no effect for adapters of type **"accelerator"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition designated by *{partition-id}*.
- Object-access permission to all crypto adapters specified in the request body.
- Action/task permission to the **Zeroize Crypto Domain (API only)** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 115. Zeroize Crypto Domain: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id <i>{partition-id}</i> does not exist, or the API user does not have object-access permission for it.
	2	The crypto adapter specified by crypto-adapter-uri in the request body does not exist or the API user does not have object-access permission for it.
	4	The partition does not support this operation.
409 (Conflict)	1	The status of the partition is not valid to perform the operation (must be one of "active" , "degraded" , "paused" , or "terminated").
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	10	The operation cannot be performed because the SE is in the process of being shut down.
	120	The status of the adapter designated in the request body is not valid to perform the operation (must be "active").
	121	The adapter designated in the request body is not part of the crypto configuration of the partition.
	122	The domain index designated in the request body is not part of the crypto configuration of the partition.
	123	The domain index designated in the request body is configured as "control" (must be "control-usage").
	124	The clearing of keys and settings was initiated, but it might or might not have completed.
500 (Server Error)	273	An unexpected error occurred during the operation.

Table 115. Zeroize Crypto Domain: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/5c6ce80a-e402-11e8-8c9d-fa163ebe78b2/operations/zeroize-
crypto-domain HTTP/1.1
x-api-session: 27daquoih13tq89tkzpivbq28hmghxegydzoa8pxwata3jrpgh
content-type: application/json
content-length: 95
{
  "crypto-adapter-uri":"/api/adapters/196a234a-e3ff-11e8-a662-fa163ebe78b2",
  "domain-index":0
}
```

Figure 83. Zeroize Crypto Domain: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 12 Nov 2018 11:47:29 GMT

<No response body>
```

Figure 84. Zeroize Crypto Domain: Response

Mount ISO Image

The Mount ISO Image operation can be used to upload a user provided ISO image. The uploaded ISO image is the image that gets attached to the partition. The contents of the ISO image are specified as binary data in the body of the POST request.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/mount-iso-image
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Query parameters:

Name	Type	Rqd/Opt	Description
image-name	String	Required	This field is used as the displayable name of the ISO image contained in the request body. It becomes the value of the boot-iso-image-name in the partition properties. The image-name must conform to the requirements of the boot-iso-image-name property as documented in the “Data model” on page 158.

Name	Type	Rqd/Opt	Description
ins-file-name	String	Required	This field is used to set the boot-iso-ins-file in the partition properties. The ins-file-name must conform to the requirements of the boot-iso-ins-file property as documented in the “Data model” on page 158.

Request body contents

The request body is the binary contents of an ISO image file. A MIME media type of application/octet-stream should be specified as the **content-type** on the request.

Description

This operation uploads an ISO image and associates it to the partition. If this operation is requested when the partition already has an ISO image associated, the newly uploaded image replaces the current one.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object to which the API user has object-access permission. If the API user does not have action permission for the **Partition Details** task, a 403 (Forbidden) status code is returned. If the partition is in one of the transitional states ("**starting**" or "**stopping**"), a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 116. Mount ISO Image: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	The required query parameters have not been specified.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id <i>{partition-id}</i> does not exist, or the API user does not have object-access permission to it.

Table 116. Mount ISO Image: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/mount-iso-image?image-
name=TestISO&ins-file-name=TestISO.ins HTTP/1.1
x-api-session: pd0nrulei0qsa1mwlpaw7cmq26rnsdchttp4w4m9gzse7gybg
```

Figure 85. Mount ISO Image: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
<No response body>
```

Figure 86. Mount ISO Image: Response

Unmount ISO Image

The Unmount ISO Image operation unmounts the currently mounted ISO from the identified partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/unmount-iso-image
```

In this request, the URI variable *{partition-id}* is the object ID of the targeted partition.

Description

This operation unmounts an ISO image that is associated to a partition. This operation sets the partition's **boot-iso-image-name** and **boot-iso-ins-file** properties to null.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object to which the API user has object-access permission. If the API user does not have action permission for the **Partition Details** task, a 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition does not currently have an ISO image associated with it.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 117. Unmount ISO Image: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with object-id <i>{partition-id}</i> does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	118	The partition does not have any image associated with it to perform the operation.
	119	The partition is currently configured to boot from the ISO image. That is, the partition's boot-device property is currently set to "iso-image" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/unmount-iso-image HTTP/1.1
x-api-session: pd0nrulei0qsa1mwl paw7cmq26rnsdc dhtp4w4m9gzse7gybg
```

Figure 87. Unmount ISO Image: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
<No response body>
```

Figure 88. Unmount ISO Image: Response

Attach Storage Group to Partition

The Attach Storage Group to Partition operation attaches a storage group to a partition.

HTTP method and URI

POST /api/partitions/{partition-id}/operations/attach-storage-group

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
storage-group-uri	String/ URI	Required	The canonical URI of the Storage Group object to be attached to the partition.

Description

The Attach Storage Group to Partition operation attaches the storage group to the partition specified by the *{partition-id}* portion of the request URI.

On successful execution, the storage group gets associated with the partition.

For a storage group of type **"fc"**, the virtual storage resources are created if the partition is in one of the following states: **"active"**, **"degraded"**, **"paused"** or **"terminated"**, if the fulfillment state of the storage group is **"complete"**.

The virtual storage resources are created immediately on the storage ports depending on the status of the unassigned world wide port names of the storage group. If the status of the world wide port name is **"validated"**, a previously verified storage port is used to create the virtual storage resource. If the status of the world wide port name is **"not-validated"**, a virtual storage resource is created without a storage port. In that case, a storage port is assigned to the virtual storage resource later, when DPM successfully verifies connectivity to the defined storage volumes on storage ports.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

This operation enables the operating system in the partition to access the storage volumes defined in the storage group, through the virtual storage resources that are assigned to storage ports. The operating system may not be able to access all the volumes defined in the storage group, until the fulfillment state of the storage group is **"complete"**.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **storage-group-uri** field in the request body must designate an existing Storage Group and the API user must have object-access permission to that storage group.

If either of these conditions are not met, status code 404 (Not Found) is returned. In addition, the API user must have action/task permissions to **Partition Details** task; otherwise, status code 403 (Forbidden) is returned. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the "**dpm-storage-management**" feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the Storage Group object designated by the **storage-group-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 118. Attach Storage Group to Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the storage group URI in request body field storage-group-uri does not designate an existing Storage Group object, or the API user does not have object-access permission to that storage group.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: " active ", " service-required ", " degraded ", or " exceptions ".)
	13	The operation is not supported when the " dpm-storage-management " feature is not enabled on the partition.
	118	The Storage Group specified by storage-group-uri is already attached to the partition.
	119	The Storage Group specified by storage-group-uri is already attached to maximum partitions according to its specification.
503 (Service Unavailable)	124	The storage group object designated by the <i>{storage-group-uri}</i> was busy performing some other operation.
	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/acab6d72-8107-11e8-9d6f-00106f0d81cb/operations/attach-storage-group HTTP/1.1
content-length:81,
content-type:application/json,
x-api-session:1zpqlgsp02sdzr2h04uxhc91ou8fdmxvp0ukxtaouqg6k3e0
{
  "storage-group-uri": "/api/storage-groups/519578c6-9569-11e8-a732-00106f0d81cb"
}
```

Figure 89. Attach Storage Group to Partition: Request

```
204 No Content
cache-control:no-cache,
date:Wed, 01 Aug 2018 09:00:27 GMT,
server:Hardware management console API web server / 2.0

<No response body>
```

Figure 90. Attach Storage Group to Partition: Response

Detach Storage Group from Partition

The Detach Storage Group from Partition operation detaches a storage group from a partition specified by the *{partition-id}* portion of the request URI.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/detach-storage-group
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
storage-group-uri	String/ URI	Required	The canonical URI of the Storage Group object to be detached from the partition.

Description

The Detach Storage Group from Partition operation detaches a storage group from the partition specified by the *{partition-id}* portion of the request URI. The virtual storage resources that were created for this partition will be deleted as part of this operation.

On successful execution, the storage group is detached from the partition and the virtual storage resources created in this storage group specifically for this partition are removed. The operating system in the partition will no longer be able to access the storage volumes defined in the storage group. If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **storage-group-uri** field in the request body must designate an existing Storage Group. If either of these conditions are not met, status code 404 (Not Found) is returned.

In addition, the API user must have action/task permissions to the **Partition Details** task; otherwise, status code 403(Forbidden) is returned.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the "**dpm-storage-management**" feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 119. Detach Storage Group from Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI <i>{partition-id}</i> does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the storage group URI in request body field storage-group-uri does not designate an existing Storage Group object, or the API user does not have object-access permission to that storage group.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: " active ", " service-required ", " degraded ", or " exceptions ".)
	13	The operation is not supported when the " dpm-storage-management " feature is not enabled on the partition.
	117	The storage-group-uri does not designate a storage group that is attached to the partition.
	124	The storage group object designated by the <i>{storage-group-uri}</i> was busy performing some other operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/acab6d72-8107-11e8-9d6f-00106f0d81cb/operations/detach-storage-  
group HTTP/1.1  
content-type:application/json,  
x-api-session:1zpqlersp02sdzr2h04uxhc91ou8fdmxvpn0ukxtaouqg6k3e0  
content-length:81,  
{  
  "storage-group-uri": "/api/storage-groups/519578c6-9569-11e8-a732-00106f0d81cb"  
}
```

Figure 91. Detach Storage Group from Partition: Request

```
204 No Content  
cache-control:no-cache,  
date:Wed, 01 Aug 2018 09:00:28 GMT,  
server:Hardware management console API web server / 2.0  
  
<No response body>
```

Figure 92. Detach Storage Group from Partition: Response

Create HBA

The Create HBA operation creates an HBA and adds it to the specified partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas
```

In this request, the URI variable *{partition-id}* is the object ID of the partition to which the HBA is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the HBA's name property.
description	String (0-1024)	Optional	The value to be set as the HBA's description property.
adapter-port-uri	String/URI	Required	The value to be set as the HBA's adapter-port-uri property.
device-number	String (4)	Optional	The value to be set as the HBA's device-number property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/ URI	The element-uri property of the newly created HBA object.

Description

The Create HBA operation creates and adds a new HBA to the partition specified by the *{partition-id}* portion of the request URI.

On successful execution, the **element-uri** field of the response body and the **Location** response header identify the new HBA. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, the **adapter-port-uri** field in the request body must designate an existing Storage Port of an existing adapter, and the API user must have object-access permission to that adapter.

If either of these conditions is not met, status code 404 (Not Found) is returned. In addition, the API user must have action task permission to **Partition Details** task; otherwise, status code 403 (Forbidden) is returned. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or CPC is not in a valid state or if the partition has the "**dpm-storage-management**" feature enabled, 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}* specified in the request URI.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 260, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 120. Create HBA: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
	7	The value of device-number is not in the range as expected by the "Data model - HBA element object" on page 179.
	8	The HBA name provided by the user is already in use by another HBA of the partition, or the provided device-number is already in use by another HBA element or by a NIC element of type " iqd " or " osd " of the partition.

Table 120. Create HBA: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the request URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	2	The object ID in the Storage Port URI in request body field adapter-port-uri does not designate an existing Adapter object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID <i>{partition-id}</i> was busy and request timed out
	6	The state of the CPC hosting partition is not valid to perform the operation. It must be in one of these valid states: "active" , "service-required" , "degraded" , or "exceptions" .
	12	The operation is not supported when the "dpm-storage-management" feature is enabled on the partition.
	116	The partition does not have sufficient resources to perform this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas HTTP/1.1
x-api-session: p7gzv1s0wo6ops84jpy88eqspkomjrn0u887n2zm8f815pcnd
content-type: application/json
content-length: 134
{
  "adapter-port-uri": "/api/adapters/55a89b60-c027-11e5-80b4-020000000192/storage-ports/0",
  "device-number": "1007",
  "name": "MyHba_7"
}
```

Figure 93. Create HBA: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/b6fe29c2-c8b3-11e5-
        bcda-020000000192
cache-control: no-cache
date: Mon, 01 Feb 2016 07:15:09 GMT
content-type: application/json;charset=UTF-8
content-length: 112
{
  "element-uri":"/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/b6fe29c2-c8b3-11e5-
        bcda-020000000192"
}
```

Figure 94. Create HBA: Response

Delete HBA

The Delete HBA operation deletes the HBA and removes it from the specified partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition from which the HBA is to be removed.
{hba-id}	String	Element ID of the HBA to be removed from the partition.

Description

The Delete HBA operation deletes the HBA and removes it from the partition specified by the {partition-id} portion of the request URI. The HBA to be deleted is identified by the {hba-id} variable in the URI.

Upon successfully removing the HBA, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Partition Details** task; otherwise, status code 403 (Forbidden) is returned.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the "**dpm-storage-management**" feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition containing the HBA element specified by the request URI.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 121. Delete HBA: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The element ID in the request URI <i>{hba-id}</i> does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the "dpm-storage-management" feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting partition is not valid to perform the operation. It must be in one of these valid states: "active" , "service-required" , "degraded" , or "exceptions" .
	151	The HBA cannot be deleted because it is currently identified with the partition's boot device.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

Delete HBA: Request

```
DELETE /api/partitions/5fca3d9a-c8b2-11e5-97e4-020000000192/hbas/4658f530-
c8b3-11e5-bcda-020000000192 HTTP/1.1
x-api-session: 1xsytajjb0a8oer7b12s7e7uarli8qb33vh3kenfgv9ibesj7c
```

Figure 95. Delete HBA: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:55:38 GMT

<No response body>
```

Figure 96. Delete HBA: Response

Update HBA Properties

The Update HBA Properties operation updates one or more writable properties of an HBA designated by the element ID and object ID of the hosting partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
<i>{partition-id}</i>	String	Object ID of the partition that has the HBA.
<i>{hba-id}</i>	String	Element ID of the HBA for which properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates the writable properties of HBA specified by *{hba-id}* as mentioned in the [“Data model - HBA element object”](#) on page 179.

On successful execution, the HBA object has been updated with the supplied property values and status code 204 (No Content) is returned without supplying a response body.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The URI path must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned. In addition, the API user must also have action/task permission to the **Partition Details** task as well, otherwise status code 403 (Forbidden) is returned.

The request body is validated against the data model for this object type to ensure that it contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or the CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the "**dpm-storage-management**" feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Partition Details** task.
- Object-access permission to the partition containing the HBA element specified by the request URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 122. Update HBA Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The value of device-number is not in the range.
	8	The HBA name provided by the user is already in use by another HBA of the partition, or the provided device-number is already in use by another HBA element or by a NIC element of type "iqd" or "osd" of the partition.
403 (Forbidden)	1	API user does not have the required permission to this operation.
404 (Not Found)	1	The <i>{partition-id}</i> in the request URI does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The element ID in the request URI <i>{hba-id}</i> does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the "dpm-storage-management" feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of these valid states: "active" , "service-required" , "degraded" , or "exceptions" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/09aea4b8-c8b3-11e5-a3ad-020000000192 HTTP/1.1
x-api-session: 3pwf2qvaioh3fld3vrlwye8zmdq4cckae3s2mtsn5sxu9dv6c0
content-type: application/json
content-length: 44
{
  "device-number": "1007",
  "name": "MyHba_7"
}
```

Figure 97. Update HBA Properties: Request


```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT
<No response body>
```

Figure 98. Update HBA Properties: Response

Get HBA Properties

The Get HBA Properties operation retrieves the properties of an HBA designated by its element ID and object ID of the hosting partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition that owns the HBA.
{hba-id}	String	Element ID of the HBA for which properties are to be returned.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the HBA element object as defined in the [“Data model - HBA element object”](#) on page 179. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the HBA specified by element ID {hba-id}.

On successful execution, all of the current properties as defined by the [“Data model - HBA element object”](#) on page 179 are provided in the response body and HTTP status code 200 (OK) is returned.

If the URI path does not designate an existing HBA element of an existing partition or if the API user does not have object-access permission to that partition or if the partition has the **"dpm-storage-management"** feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object in the request URI.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 267.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 123. Get HBA Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action permission to this operation.
404 (Not Found)	1	The object ID in the request URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	5	The element ID in the request URI <i>{hba-id}</i> does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the "dpm-storage-management" feature is enabled on the partition.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/ea792b18-c8b2-11e5-a3ad-020000000192 HTTP/1.1
x-api-session: 2xvikaohvoji3npmjrdx2ezrpi7tll6z6zur1chx864oda1f6u
```

Figure 99. Get HBA Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 10:02:29 GMT
content-type: application/json;charset=UTF-8
content-length: 417
{
  "adapter-port-uri": "/api/adapters/55e59092-c027-11e5-80b4-020000000192/storage-ports/0",
  "class": "hba",
  "description": "Test",
  "device-number": "1003",
  "element-id": "ea792b18-c8b2-11e5-a3ad-020000000192",
  "element-uri": "/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/ea792b18-c8b2-11e5-a3ad-020000000192",
  "name": "MyHba_3",
  "parent": "/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192",
  "wwpn": "0000000000000000"
}
```

Figure 100. Get HBA Properties: Response

Reassign Storage Adapter Port

The Reassign Storage Adapter Port operation reassigns the backing storage adapter to which the specified HBA is associated to a new backing FCP adapter.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas/{hba-id}/operations/reassign-storage-adapter-port
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition that owns the HBA.
{hba-id}	String	Element ID of the HBA for which properties are to be reassigned.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
adapter-port-uri	String/ URI	Required	The value to be set as the HBA's adapter-port-uri property.

Description

The Reassign Storage Adapter Port operation changes the adapter port associated with specified partition to different adapter port specified in the request body.

On successful execution, status code 204 (No Content) is returned without supplying a response body.

The request URI must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned.

In addition, the API user must have action/task permission to the **Partition Details** task, otherwise status code 403 (Forbidden) is returned.

If the partition is in any of the transitional states ("**starting**" or "**stopping**") or CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the "**dpm-storage-management**" feature enabled, a 404 (Not Found) status code is returned.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition containing the HBA element specified by the request URI.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 124. Reassign Storage Adapter Port: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have required permission for this operation.
404 (Not Found)	1	The object ID in the request URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	2	The object ID in the Storage Port URI in the request body field adapter-port-uri does not designate an existing Adapter object, or the API user does not have object-access permission to the object.
	5	The element ID in the request URI <i>{hba-id}</i> does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the "dpm-storage-management" feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with <i>{partition-id}</i> was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of these valid states: "active" , "service-required" , "degraded" , or "exceptions" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/9b76b558-c8b2-11e5-97e4-020000000192/operations/reassign-storage-adapter-port HTTP/1.1
x-api-session: 4x5d5t0yyqcr52ngjj3zc8y8d1ztaunjkw8g9bw2uku2iflii
content-type: application/json
content-length: 90
{
  "adapter-port-uri": "/api/adapters/55a89b60-c027-11e5-80b4-020000000192/storage-ports/0"
}
```

Figure 101. Reassign Storage Adapter Port: Request

```
204 No Content
```

Figure 102. Reassign Storage Adapter Port: Response

Send OS Command

The Send OS Command operation sends a command to the operating system running in a partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/send-os-cmd
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
is-priority	Boolean	Optional	An indication of whether this is a priority operating system command. Set to true for priority operating system commands or false for non-priority operating system commands. The default is false .
operating-system-command-text	String (1-200)	Required	The text of the operating system command.

Description

This operation sends a command to the operating system running in the Partition targeted by the request URI.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the targeted partition is not available. Some examples are when the Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

On successful execution, the command is sent to the operating system running in the target Partition object and status code 204 (No Content) is returned without supplying a response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required action/task permission to this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	332	The messages interface is not available.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd/operations/
  send-os-cmd HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 69
{
  "is-priority": false,
  "operating-system-command-text": "help"
}
```

Figure 103. Send OS Command: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT

<No response body>
```

Figure 104. Send OS Command: Response

Open OS Message Channel

The Open OS Message Channel operation opens a message channel to the operating system running in a partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/open-os-message-channel
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

An optional request body can be specified as a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
include-refresh-messages	Boolean	Optional	An indication of whether refresh operating system messages should be sent. Set to true to receive refresh messages, or false to prevent refresh messages. The default is true .

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
topic-name	String (1-128)	The name of the os-message-notification topic.

Description

This operation opens a message channel to the operating system running in the Partition targeted by the request URI. The message channel is implemented as a JMS topic, specifically as an os-message-notification topic. See [Chapter 4, “Asynchronous notification,”](#) on page 57 for information on JMS usage on the HMC. The API user can connect to this topic to start the flow of new (and refreshed) operating system messages.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the targeted partition is not available. Some examples are when the Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If an os-message-notification topic already exists for this partition for the current API session, the operation fails.

On successful execution, the message channel is opened and the os-message-notification topic name is returned in the response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 273.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 125. Open OS Message Channel: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Table 125. Open OS Message Channel: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required action/task permission to this operation.
404 (Not Found)	1	The object ID in the URI (<i>{partition-id}</i>) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	331	An os-message-notification topic already exists for this partition for the current API session. Use the Get Notification Topics operation to determine the topic name.
	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd/operations/open-os-message-channel
HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 38
{ "include-refresh-messages": false }
```

Figure 105. Open OS Message Channel: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json; charset=UTF-8
{ "topic-name": "mikeuser.1osmsg.cpc1.par1" }
```

Figure 106. Open OS Message Channel: Response

List OS Messages of a Partition

The List OS Messages of a Partition operation lists all currently available operating system (OS) messages for a partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/operations/list-os-messages
```

In this request, the URI variable *{partition-id}* is the object ID of the target partition.

Query parameters:

Name	Type	Rqd/Opt	Description
begin-sequence-number	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number less than this are omitted from the results. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
end-sequence-number	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number greater than this are omitted from the results. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
os-messages	Array of os-message-info objects	Array of nested os-message-info objects as described in the next table. The array elements are in order of increasing sequence number, unless that number has wrapped back around to 0 in which case the element with sequence number 0 follows the element with the highest sequence number, thus maintaining the temporal order of the messages.

Each nested os-message-info object contains the following fields:

Field name	Type	Description
sequence-number	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message. It can be used for filtering purposes.
message-text	String	The text of the operating system message.
message-id	String	The message identifier of the operating system message.
timestamp	Timestamp	The time when the operating system message was created or -1 if this information is not available from the corresponding operating system.
sound-alarm	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (true) or not (false).
is-priority	Boolean	Specifies whether the operating system message is a priority message (true) or not (false). A priority message indicates a critical condition that requires immediate attention.
is-held	Boolean	Specifies whether the operating system message is a held message (true) or not (false). A held message is one that requires a response.
prompt-text	String	Specifies the prompt text that is associated with this operating system message or null indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.

Field name	Type	Description
os-name	String	Specifies the name of the operating system that generated this operating system message or null indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the partition in which the operating system is running.

Description

This operation lists the currently available messages from the operating system running in the specified partition. Only a certain amount of OS message data from each partition is preserved by the HMC for retrieval by this operation. If the OS produces more than that amount, the oldest non-held, non-priority OS messages are no longer available. A gap in the sequence numbers indicates a loss of messages. A loss may be due to that space limitation, or it may be due to the deletion of messages by a console user or the OS.

If the request URI does not identify a Partition object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned. In addition to having object-access permission to the partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target partition is not available. Some examples are when the partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If the **begin-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** less than that are omitted from the response. If the **end-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** greater than that are omitted from the response.

If there are no available OS messages for the specified partition or if no OS messages are to be included in the response due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 275](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/partitions/05ca8cc6-8d0c-11e7-804a-00106f0dc539/operations/
list-os-messages HTTP/1.1
x-api-session: 5uj6itqzadjd9rb7o6rjgpf0de5494qop489hv8fdp2k6mfjjf
```

Figure 107. List OS Messages of a Partition: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 31 Aug 2017 02:02:26 GMT
content-type: application/json; charset=UTF-8
content-length: 593
{
  "os-messages": [
    {
      "is-held": false,
      "is-priority": false,
      "message-id": 1011,
      "message-text": "Preparing system.\n",
      "os-name": null,
      "prompt-text": "",
      "sequence-number": 0,
      "sound-alarm": false,
      "timestamp": -1
    },
    {
      "is-held": false,
      "is-priority": false,
      "message-id": 1012,
      "message-text": "Starting system.\n",
      "os-name": null,
      "prompt-text": "",
      "sequence-number": 1,
      "sound-alarm": false,
      "timestamp": -1
    },
    {
      "is-held": false,
      "is-priority": false,
      "message-id": 1023,
      "message-text": "[ 75.476186] systemd-udevd[66]: starting version 208\n",
      "os-name": null,
      "prompt-text": "",
      "sequence-number": 2,
      "sound-alarm": false,
      "timestamp": -1
    }
  ]
}
```

Figure 108. List OS Messages of a Partition: Response

Delete Partition OS Message

The Delete Partition OS Message operation deletes a single OS message.

HTTP method and URI

POST /api/partitions/{*partition-id*}/operations/delete-os-message

In this request, the URI variable {*partition-id*} is the object ID of the target partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
sequence-number	Long	Required	The sequence-number property of the OS message on the partition to delete.

Description

This operation deletes a specific partition OS message. The OS message to be deleted is uniquely identified by the combination of the {*partition-id*} variable in the URI and the **sequence-number** in the request body.

The URI path must designate an existing partition and the API user must have object-access permission to it; otherwise status code 404 (Not Found) is returned.

The request body must designate an existing OS message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the Operating System Messages task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by {*partition-id*}.
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission for the Operating System Messages task.
404 (Not Found)	1	The object ID in the URI { <i>partition-id</i> } does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	336	The sequence-number in the request body does not designate an existing OS message on the partition.

HTTP error status code	Reason code	Description
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/4767c72e-b00d-3a1c-ac89-87f821404a0b/operations/delete-os-message
HTTP/1.1
x-api-session: 6d1q521ruuym4dhqubv3m77678qbjx7c68wpdpv5v75ut8n1iq
content-type: application/json
content-length: 22
{
  "sequence-number":0
}
```

Figure 109. Delete Partition OS Message: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 15 Nov 2018 17:21:00 GMT

<No response body>
```

Figure 110. Delete Partition OS Message: Response

Get ASCII Console WebSocket URI

The Get ASCII Console WebSocket URI operation returns a new WebSocket URI for the ASCII console exposed by the operating system running in this partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/get-ascii-console-websocket-uri
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
force-takeover	Boolean	Optional	Whether the ASCII console session for this partition opened from another HMC should be forcefully closed and taken over by this HMC. Default: False Note: This flag does not affect the ASCII console session for this partition opened from the same HMC.

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body is a JSON object with the following fields:

Field name	Type	Description
websocket-uri	String	The WebSocket URI that should be used to connect to the ASCII console exposed for the operating system running in this partition.

Description

The `Get ASCII Console WebSocket URI` operation returns the URI that a WebSocket client should use to connect to the ASCII console of a partition's operating system. The API client should prepend the secure protocol ('wss'), the WebSocket host, and port information to the URI before using a WebSocket client to connect to it. The HMC is the WebSocket host that accepts incoming WebSocket connection requests.

It is possible that another client has already connected to the ASCII console of a partition from the same HMC or from a different HMC. When the client opens a WebSocket connection with the WebSocket URI obtained by specifying the force-takeover flag as **true**, the existing connection from a different HMC is broken and the same ASCII console session is taken over by this client. It is not possible to break or take over an existing connection to the ASCII console of a partition from the same HMC, even if the force-takeover flag was specified as **true**.

A maximum of 50 WebSocket URIs may be associated with any given API session. If there are already 50 WebSocket URIs associated with the API session, a 409 (Conflict) status code is returned.

Once returned by this operation, a WebSocket URI remains associated with the API session until the corresponding WebSocket is opened and closed.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object to which the API user has object-access permission. If the API user does not have action/task permission for the **Integrated ASCII Console** task, a 403 (Forbidden) status code is returned. If the partition is in any other state than "active" or if there is already an active connection to this partition's ASCII console, a 409 (Conflict) status code is returned unless the active connection is from a different HMC and the force-takeover flag was specified as **true**.

Note: This operation does not create a connection to the partition's ASCII console. A connection does not exist until the client specifically opens a connection using the WebSocket URI returned by this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/Task permission to the **Integrated ASCII Console** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in ["Response body contents" on page 279](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the Integrated ASCII Console task.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The Object ID in the URI <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	100	There is already a connection to this partition's ASCII console opened from this HMC.
	101	There are already 50 WebSocket URIs associated with this API session.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/partitions/dc63eeac-ce0d-11e7-8b57-00106f0dc513/operations/
  get-ascii-console-websocket-uri HTTP/1.1
x-api-session: 4a4f1hj12hldmm26brcpfnydk663gt6gtyxq4iwto26g2r6wq1
content-type: application/json
content-length: 24
{
  "force-takeover":true
}
```

Figure 111. Get ASCII Console WebSocket URI: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 12 Dec 2017 14:50:15 GMT
content-type: application/json;charset=UTF-8
content-length: 85
{
  "websocket-uri":"/api/websock/4a4f1hj12hldmm26brcpfnydk663gt6gtyxq4iwto26g2r6wq1/1"
}
```

Figure 112. Get ASCII Console WebSocket URI: Response

Inventory service data

Information about the partitions managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for partition objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by inventory class, implicitly via a containing category, or by default) that objects of the class **"partition"** are to be included. An entry for a particular partition is included only if the API user has object-access permission to that object as described in the Get Partition Properties operation.

For each partition to be included, the inventory response array includes the following:

- An array entry for the Partition object itself. This entry is a JSON object with the same contents as is specified in the response body contents section for [“Get Partition Properties”](#) on page 195. That is, the data provided is the same as would be provided if a Get Partition Properties operation were requested targeting this object.

- An array entry for each NIC associated with the partition. For each such NIC, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for [“Get NIC Properties”](#) on page 237.
- An array entry for each virtual function associated with the partition. For each such virtual function, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for [“Get Virtual Function Properties”](#) on page 227.
- An array entry for each HBA associated with the partition. For each such HBA, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for [“Get HBA Properties”](#) on page 267.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single partition. This object would appear as a sequence of array entries in the response array:

```
{
  "acceptable-status": [
    "active"
  ],
  "access-basic-counter-set": false,
  "access-basic-sampling": false,
  "access-coprocessor-group-set": false,
  "access-crypto-activity-counter-set": false,
  "access-diagnostic-sampling": false,
  "access-extended-counter-set": false,
  "access-global-performance-data": false,
  "access-problem-state-counter-set": false,
  "auto-start": false,
  "autogenerate-partition-id": true,
  "boot-configuration-selector": 0,
  "boot-device": "test-operating-system",
  "boot-ftp-host": null,
  "boot-ftp-insfile": null,
  "boot-ftp-username": null,
  "boot-iso-image-name": null,
  "boot-iso-ins-file": null,
  "boot-logical-unit-number": "",
  "boot-network-device": null,
  "boot-os-specific-parameters": "",
  "boot-record-lba": "0",
  "boot-removable-media": null,
  "boot-removable-media-type": null,
  "boot-storage-device": null,
  "boot-timeout": 60,
  "boot-world-wide-port-name": "",
  "class": "partition",
  "cp-absolute-processor-capping": false,
  "cp-absolute-processor-capping-value": 1.0,
}
```

Figure 113. Partition object: Sample inventory data - Response (Part 1)


```

"cp-processing-weight-capped":false,
"cp-processors":1,
"crypto-configuration":null,
"current-cp-processing-weight":1,
"current-ifl-processing-weight":1,
"degraded-adapters":[],
"description":"",
"has-unacceptable-status":false,
"hba-uris":[],
"ifl-absolute-processor-capping":false,
"ifl-absolute-processor-capping-value":1.0,
"ifl-processing-weight-capped":false,
"ifl-processors":0,
"initial-cp-processing-weight":100,
"initial-ifl-processing-weight":100,
"initial-memory":4096,
"is-locked":false,
"maximum-cp-processing-weight":999,
"maximum-ifl-processing-weight":999,
"maximum-memory":4096,
"minimum-cp-processing-weight":1,
"minimum-ifl-processing-weight":1,
"name":"Demo_DPM",
"nic-uris":[],
"object-id":"2812231a-048c-11e7-a266-00106f0d81cb",
"object-uri":"/api/partitions/2812231a-048c-11e7-a266-00106f0d81cb",
"os-name":"",
"os-type":"Linux",
"os-version":"3.10.0",

```

Figure 114. Partition object: Sample inventory data - Response (Part 2)

```

"parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
"partition-id":"03",
"permit-aes-key-import-functions":true,
"permit-cross-partition-commands":false,
"permit-des-key-import-functions":true,
"processor-management-enabled":false,
"processor-mode":"shared",
"reserve-resources":false,
"reserved-memory":0,
"short-name":"DEMOPM",
"ssc-boot-selection":null,
"ssc-dns-servers":[],
"ssc-host-name":null,
"ssc-ipv4-gateway":null,
"ssc-ipv6-gateway":null,
"ssc-master-pw":null,
"ssc-master-userid":null,
"status":"active",
"threads-per-processor":1,
"type":"linux",
"virtual-function-uris":[]
}

```

Figure 115. Partition object: Sample inventory data - Response (Part 3)

Adapter object

An Adapter object represents a single adapter for a DPM-enabled CPC. The Adapter object APIs provide access to the set of adapters that are managed by a CPC that is enabled for DPM. APIs exist to query adapters, update selected properties of physical adapters, and create, delete, and update HiperSockets. APIs also exist to query and update properties of the ports of the adapters and to get a list of the partitions for which an adapter is allocated to provide I/O and virtual functions.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, including the operational-status-related properties, with the following class-specific specializations:

Table 126. Adapter object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path for an Adapter object is of the form <code>/api/adapters/{adapter-id}</code> where <code>{adapter-id}</code> is the value of the object-id property of the Adapter object.
object-id	—	String (36)	The unique identifier for the adapter instance.
parent	—	String/ URI	The parent of an adapter is conceptually its owning CPC, and so the parent value is the canonical URI path for the CPC.
class	—	String (7)	The class of an Adapter object is "adapter" .
name	(w)(pc)	String (1-64)	The display name specified for the adapter. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other configured adapters and created HyperSockets for the CPC.
description	(w)(pc)	String (0-1024)	Arbitrary text providing more descriptive information about the adapter.
status	(sc)	String Enum	The status of the adapter, which must be one of the following values: • "active" - The adapter is configured and functioning normally. • "not-active" - The adapter is configured as not active. • "not-detected" - The previously configured adapter is no longer detected; usually indicates that the card has been physically removed from the system. Not applicable if type is "hipersockets". • "exceptions" - The adapter is configured but unavailable due to an error affecting the adapter hardware. The adapter is not active. • "service" - The adapter is configured but unavailable because it is being serviced.
additional-status	—	String Enum	This property is not supported for adapters and will not be included in its data model.
acceptable-status	—	Array of String Enum	This property is not supported for adapters and will not be included in its data model.

Class specific additional properties

In addition to the properties defined via included schema, this object includes the following additional class-specific properties:

Table 127. Adapter object: class-specific properties

Name	Qualifier	Type	Description	Supported "adapter family" values
type	(pc)	String Enum	The type of the adapter. Values: • "crypto" - Cryptographic adapter. • "fcp" - A Fibre Channel attached storage resource. • "hipersockets" - A HiperSockets adapter. • "osd" - OSA Direct Express. • "osm" - OSA-Express for zManager. • "roce" - RDMA over Converged Ethernet. • "zedc" - zEnterprise Data Compression. • "fc" - Fibre Connection attached storage resource. • "cna" - Cloud Network Adapter • "not-configured" - The adapter is not configured. Currently this applies only to adapters of adapter-family "ficon". This value will change when the adapter type is updated via the Change Adapter Type operation.	All
adapter-id	—	String (3)	ID of the adapter; three character lower-case hex string. This string is the PCHID of the physical adapter or the VCHID of the virtual HiperSockets adapter	All
adapter-family	—	String Enum	The family of the card detected for this adapter at the time of its creation. Values: • "hipersockets" - A virtual HiperSockets card. • "osa" - An OSA card. • "ficon" - A FICON card. • "roce" - An RDMA over Converged Ethernet card. • "crypto" - A cryptographic card. • "accelerator" - Adapter card which provides acceleration capability. • "cna" - A Cloud Network Adapter card.	All

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
detected-card-type	(pc)	String Enum	The type detected for the card with the adapter-id of this Adapter object. Values: • "hipersockets" - A virtual HiperSockets card • "osa-express-4s-1gb" - An OSA Express4S Gigabit Ethernet LX/SX. • "osa-express-4s-10gb" - An OSA Express4S Gigabit Ethernet LR/SR. • "osa-express-4s-1000base-t" - An OSA Express4S 1000BASE-T Ethernet. • "osa-express-5s-1gb" - An OSA Express5S Gigabit Ethernet LX/SX. • "osa-express-5s-10gb" - An OSA Express5S Gigabit Ethernet LR/SR. • "osa-express-5s-1000base-t" - An OSA Express5S 1000BASE-T Ethernet. • "osa-express-6s-1gb" - An OSA Express6S Gigabit Ethernet LX/SX. • "osa-express-6s-10gb" - An OSA Express6S Gigabit Ethernet LR/SR. • "osa-express-6s-1000base-t" - An OSA Express6s 1000BASE-T Ethernet. • "osa-express-7s-25gb" - An OSA Express7S 25 Gigabit Ethernet Short Reach • "10gbe-roce-express" - 10 GbE RoCE Express. • "roce-express-2" - A 10 GbE RoCE Express2 • "roce-express-2-25gb" - A 25 GbE RoCE Express2 • "crypto-express-5s" - Crypto Express5S. • "crypto-express-6s" - Crypto Express6S. • "ficon-express-8" - FICON Express8 10KM LX (2, 4, 8 Gbps) or FICON Express8 SX (2, 4, 8 Gbps). • "ficon-express-8s" - FICON Express8S 10KM LX (2, 4, 8 Gbps) or FICON Express8S SX (2, 4, 8 Gbps). • "ficon-express-16s" - FICON Express16S 10KM LX (4, 8, 16 Gbps) or FICON Express16S SX (4, 8, 16 Gbps). • "ficon-express-16s-plus" - FICON Express 16S+ LX 10KM (4, 8, 16 Gbps) or FICON Express16S+ SX (4, 8, 16 Gbps). • "ficon-express-16sa" - FICON Express 16SA LX 10KM (8, 16 GFC) or FICON Express16SA SX (8, 16 GFC) • "fcp-express-32s" - FCP Express32S LX (16, 32 Gbps) or FCP Express32S SX (16, 32 Gbps) • "zedc-express" - zEnterprise Data Compression (FPGA Corsa A5). • "cloud-network-x5" - Cloud Network Adapter - X5. • "unknown" - The detected-card-type could not be determined.	All

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
card-location	—	String (14 or 20)	Location of the physical I/O card. 14 or 20 characters: "<i>www-xxx-J.yy</i>" for single-port adapters, or "<i>www-xxx-J.yy-J.zz</i>" for two-port adapters, where: <i>www</i> - ID of the cage in which the card is installed. <i>xxx</i> - ID of the slot in the cage in which the card is installed. <i>yy</i> - ID of the first jack/port on the card. <i>zz</i> - ID of the second jack/port on the card.	ficon, osa, roce, crypto, accelerator, cna
port-count	—	Integer	Number of ports on the adapter.	ficon, osa, roce, hipersockets, cna
network-port-uris	—	Array of String/ URI	List of network ports for this adapter. Each element in this array is the canonical URI path of a Network Port object. The number of entries in this list matches the value of the port-count property.	osa, roce, hipersockets, cna
storage-port-uris	—	Array of String/ URI	List of storage ports for this adapter. Each element in this array is the canonical URI path of a Storage Port object. The number of entries in this list matches the value of the port-count property.	ficon
state	(pc)	String Enum	The current state of the adapter, which must be one of the following values: "online" - The adapter is online. "stand-by" - The adapter has been configured off. "reserved" - The adapter has been configured into service mode. "unknown" - The state of the adapter cannot be determined because of a communications issue.	All
maximum-transmission-unit-size	(w)(pc)	Integer Enum	The maximum transmission unit size of the virtual switch. The maximum frame size is implied by this value. Values: 8 - 8 KB MTU size and 16 KB maximum frame size. 16 - 16 KB MTU size and 24 KB maximum frame size. 32 - 32 KB MTU size and 40 KB maximum frame size. 56 - 56 KB MTU size and 64 KB maximum frame size.	hipersockets

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
configured-capacity	(pc)	Integer	For FCP adapters (type is "fcp"), this value is the number of host bus adapters (HBAs) that are configured, but not necessarily allocated, to the partitions assigned to the adapter. For channel-based network adapters (type is "osd" or "hipersockets"), this value is the number of subchannels that are configured, but not necessarily allocated, to partitions assigned to the adapter. For Single Root I/O Virtualization (SR-IOV) adapters (type is "roce", or "zedc"), this value is the number of virtual PCI functions that are configured, but not necessarily allocated, to partitions assigned to the adapter. For adapters with type values of "osm", or "not-configured", the value will always be 0. configured-capacity may be larger than allowed-capacity or maximum-total-capacity, which indicates that the adapter is over-committed, and therefore, all partitions that include this adapter in their configuration cannot be active at the same time.	ficon, osa, roce, hipersockets, accelerator, cna
used-capacity	(pc)	Integer	For FCP adapters (type is "fcp"), this value is the number of host bus adapters (HBAs) that are allocated to partitions assigned to the adapter. For channel-based network adapters (type is "osd" or "hipersockets"), this value is the number of subchannels that are allocated to partitions assigned to the adapter. For Single Root I/O Virtualization (SR-IOV) adapters (type is "roce", or "zedc"), this value is the number of virtual PCI functions that are allocated to partitions assigned to the adapter. For adapters with type value "osm", the value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
allowed-capacity	(w)(pc)	Integer	For FCP adapters (type is "fcp"), this value is the maximum number of host bus adapters (HBAs) that may be allocated to the partitions assigned to the adapter. For FCP adapters, this property is writable and must be an integer from 0 to maximum-total-capacity. For channel-based network adapters (type is "osd" or "hipersockets"), this value is the maximum number of subchannels that may be allocated to partitions assigned to the adapter. For these adapters, this property is read-only and its value will be automatically defined by the system when the adapter is configured. For Single Root I/O Virtualization (SR-IOV) adapters (type is "roce", or "zedc"), this value is the maximum number of virtual PCI functions that may be allocated to partitions assigned to the adapter. For these adapters, this property is read-only and its value will be automatically defined by the system when the adapter is configured. For adapters with type value "osm", this property is read-only and its value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna
maximum-total-capacity	(pc)	Integer	This is the largest permitted value of allowed-capacity. For adapters with type value "osm", the value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna
channel-path-id	(w)(pc)	String (2)	Channel path ID (CHPID) used by the adapter's partition; two character hex string. This value will be null when the value of type is "not-configured". channel-path-id cannot be written if the adapter is configured to any partition.	ficon, osa, hipersockets

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
physical-channel-status	(pc)	String Enum	Status of the physical channel. Values: • "operating" - The channel path is operating. Maps to status value "active". • "no-power" - The power is off for the hardware that supports the channel path, the channel path is not operating. Maps to status value "not-active". • "service" - The path is in single channel service (SCS) mode and is not in the active I/O configuration. The channel path is not operating. Maps to status value "service". • "stopped" - The channel path is not operating. Maps to status value "not-active". • "not-defined" - The channel path is not defined in the active IOCDS. The channel path is not operating. Maps to status value "not-active". • "definition-error" - The channel path that is specified in the active input/output configuration data set (IOCDS) does not match the characteristics of the installed channel, or the channel type is incompatible with the current storage allocation, or the level of the installed channel hardware does not support the definition in the IOCDS. The channel path is not operating. Maps to status value "exceptions". • "suspended" - The channel path is suspended. The channel path is not operating. Maps to status value "service". • "check-stopped" - The channel path is unavailable due to a permanent machine error affecting the channel hardware. The channel path is not operating. Maps to status value "exceptions". • "wrap-block" - A wrap block is installed on the channel path's channel interface. Note: Wrap blocks are used during special diagnostic tests performed on the channel. Wrap blocks must be removed before system initialization to allow the channel to initialize completely. The channel path is not operating. Maps to status value "exceptions". • "permanent-error" - The channel path is unavailable due to a permanent outboard error. The channel path is not operating. Maps to status value "exceptions". • "initializing" - The firmware is being loaded into the channel card and then the channel card is starting. Maps to status value "not-active".	all

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
			• "loss-of-signal" - The channel path detected a link-signal error. The level of the signal on the link is below the value specified for reliable communication. Maps to status value "exceptions". • "loss-of-synchronization" - The channel path detected a link-signal error. The bit synchronization with the signal was lost. The channel path is not operating. Maps to status value "exceptions". • "not-operational-link" - The channel path detected a link failure due to a non-operational sequence. The channel path is not operating. Maps to status value "exceptions". • "sequence-time-out" - The channel path detected a link failure due to a sequence timeout. The channel path is not operating. Maps to status value "exceptions". • "sequence-not-permitted" - The channel path detected a link failure due to an illegal sequence for a link. The channel path is not operating. Maps to status value "exception". • "terminal-condition" - The channel path is not available due to an interface-hung condition. This condition can occur after an interface or channel error if the control unit or device fails to disconnect from the interface when requested by the channel. The channel path is not operating. Maps to status value "exceptions". • "offline-signal-received" - The channel path detected an offline sequence, indicating that the sender is in offline mode and subsequent link-signal errors that are detected by the channel path are not to be reported. For an ES conversion channel, this condition can occur only when the channel is wrongly attached to another channel, switch, or control unit instead of an ESCON Converter. The channel path is not operating. Maps to status value "exceptions". • "fabric-login-sequence-failure" - This condition means that the channel detected a failure during the Fabric Login procedure. Maps to status value "exceptions". • "port-login-sequence-failure" - This condition means that the channel detected a failure during the registration procedure. In order for a FICON channel to communicate with devices on a control unit, it must perform a Port Login with that control unit. Maps to status value "exceptions". • "state-change-registration-failure" - This condition means that the channel detected a failure during the registration procedure. A FICON channel is required to register with the switch to receive state change notification. Maps to status value "exceptions".	

Table 127. Adapter object: class-specific properties (continued)

Name	Qualifier	Type	Description	Supported "adapter family" values
			• "invalid-attachment-failure" - Occurs when the channel determines that it is connected to a switch, but the IOCDS specifies that it should be directly connected to a control unit or the contrary. Maps to status value "exceptions". • "test-mode" - The channel path is in test mode. The channel path is not operating. Maps to status value "not-active". • "bit-error-threshold-exceeded" - The number of bit errors the channel path detected while receiving or sending data is more than the threshold set for its bit error counter. The channel path is not operating. Maps to status value "exceptions". • "ifcc-threshold-exceeded" - The number of interface control checks (IFCCs) the channel path detected is more than the threshold set for its IFCC counter. IFCCs may continue to occur; the error logs will not be created and sent to the Support Element. Maps to status value "exceptions". • "io-suppressed" - The channel path input/output (I/O) suppression is active. I/O suppression prevents the channel subsystem from selecting any device and fetching the first channel command word (CCW) of a channel program. The channel path is not operating. Maps to status value "not-active".	
crypto-number	(pc)	Integer	Identifier of the crypto adapter in the range 0-15. Crypto number must be unique to other configured crypto adapters for the CPC.	crypto
crypto-type	(pc)	String Enum	Crypto type. Values: • "accelerator" - Crypto Express5S Accelerator. • "cca-coprocessor" - Crypto Express5S Coprocessor. • "ep11-coprocessor" - Crypto Express5S EP11 Coprocessor.	crypto
udx-loaded	(pc)	Boolean	true if the configured crypto was UDX-loaded; false otherwise. udx-loaded cannot be read, and a null is returned, if any of the following conditions are true: • The CPC that contains the cryptographic adapter is not started. • The adapter's status is "exceptions". • The adapter's status is "not-active". • The adapter's state is not "online".	crypto

Table 127. Adapter object: class-specific properties (continued)				
Name	Qualifier	Type	Description	Supported "adapter family" values
tke-commands-enabled	(w)(pc)	Boolean	true if the crypto permits TKE commands; false otherwise. tke-command-enabled cannot be read or written, and a null is returned on a read, if any of the following conditions are true: - The CPC that contains the cryptographic adapter is not started. - The adapter's status is "exceptions". - The adapter's status is "not-active". - The adapter's state is not "online".	crypto

Network Port element object

A Network Port element object defines the index and description associated with a network adapter port.

Table 128. Network Port element object properties				
Name	Qualifier	Type	Description	
element-id	—	String (1-2)	Unique ID for the network port within the scope of the containing adapter.	
element-uri	—	String/ URI	The canonical URI path for the network port object, of the form <code>/api/adapters/{adapter-id}/network-ports/{network-port-id}</code> , where <code>{adapter-id}</code> is the object-id of the adapter, and <code>{network-port-id}</code> is the element-id of this network port.	
parent	—	String/ URI	The parent of a network port is its owning adapter, so the parent value is the canonical URI path for the adapter.	
class	—	String (12)	The class of a network port object is "network-port" .	
index	—	Integer	The index of the port. Adapters whose type is "osd" or "hipersockets" use a base index of zero while adapters whose type is "roce" use a base index of one. If adapter type is "osd" or "hipersockets", values may range from zero to port-count. Values may go from one to (port-count + 1) if adapter type is "roce".	
name	—	String (1-64)	The name of the network port, which is currently always of the form "Port #", where # is the index of the network port. This form is subject to change in the future.	
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the port.	

Storage Port element object

A Storage Port element object defines the index and description associated with a storage adapter port.

Table 129. Storage Port element object properties

Name	Qualifier	Type	Description	Supported adapter "type" values
element-id	—	String (1-2)	Unique ID for the storage port within the scope of the containing adapter.	All
element-uri	—	String/ URI	The canonical URI path for the storage port object, of the form <code>/api/adapters/{adapter-id}/storage-ports/{storage-port-id}</code> , where <code>{adapter-id}</code> is the object-id of the adapter, and <code>{storage-port-id}</code> is the element-id of this storage port.	All
parent	—	String/ URI	The parent of a storage port is its owning adapter, so the parent value is the canonical URI path for the adapter.	All
class	—	String (12)	The class of a storage port object is "storage-port" .	All
index	—	Integer	The index of the port. Adapters whose type is "fcp" use a base index of zero. If adapter type is "fcp" , values may range from zero to port-count .	All
name	—	String (1-64)	The name of the storage port, which is currently always of the form "Port #", where # is the index of the storage port. This form is subject to change in the future.	All
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the port.	All
fabric-id	(pc)	String (16)	The World Wide Name (WWN) of the uplink Fibre Channel switch. When an adapter is not configured into a storage network, null is returned.	fcp
connection-endpoint-uri	(w)(pc)	String/ URI	The canonical URI of the storage switch or storage subsystem to which this adapter is connected. This value may be null , which indicates that the adapter identified by connection-endpoint-uri is not connected to anything.	fc

Table 129. Storage Port element object properties (continued)

Name	Qualifier	Type	Description	Supported adapter "type" values
connection-endpoint-class	—	String Enum	The class of the object to which this adapter is connected. Values: • "storage-subsystem" – The adapter is directly connected to a storage subsystem. • "storage-switch" – The adapter is connected to a storage switch. This value will be null when connection-endpoint-uri is null and non-null when connection-endpoint-uri is non-null.	fc

List Adapters of a CPC

The List Adapters of a CPC operation lists the adapters managed by the CPC with the given identifier.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/adapters
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern to limit returned objects to those that have a matching name property.
adapter-id	String	Optional	Filter pattern to limit returned objects to those that have a matching adapter-id property.
adapter-family	String Enum	Optional	Filter string to limit returned objects to those that have a matching adapter-family property. Value must be a valid adapter adapter-family property value.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid adapter type property.
status	String Enum	Optional	Filter string to limit returned objects to those that have a matching status property. Value must be a valid adapter status property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
adapters	Array of adapter-info objects	Array of adapter-info objects, described in the next table. Returned array may be empty.

Each nested adapter-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the Adapter object.
name	String	The name property of the Adapter object.
adapter-id	String	The adapter-id property of the Adapter object.
adapter-family	String Enum	The adapter-family property of the Adapter object.
type	String Enum	The type property of the Adapter object.
status	String Enum	The status property of the Adapter object.

Description

This operation lists the adapters that are managed by the identified CPC. The **object-uri**, **name**, **adapter-id**, **adapter-family**, **type**, and **status** are provided for each.

If the **object-id** *{cpc-id}* does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the CPC identified by *{cpc-id}* is not enabled for DPM, an empty list is returned.

If the **name** or **adapter-id** query parameters are specified, the returned list is limited to those adapters that have a same-named property matching the specified filter pattern. If the **name** or **adapter-id** parameter is omitted, this filtering is not done for the corresponding property.

If the **adapter-family**, **type** or **status** query parameters are specified, the parameter is validated to ensure it is a valid value for the same-named property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those adapters that have a corresponding property matching the specified value. If the **adapter-family**, **type** or **status** parameter is omitted, this filtering is not done for the corresponding property.

An adapter is included in the list only if the API user has object-access permission for that object. If the specified CPC is a manager of an adapter but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no adapters are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is specified in the request URI.
- Object-access permission to any adapter object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 295](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 130. List Adapters of a CPC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	A CPC with the object ID {cpc-id} does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29/adapters
x-api-session:3aonctd6ljeg2k4amfb4sz2rb6yo2esgah7dtlpgofor3rzgk1

<no request body>
```

Figure 116. List Adapters of a CPC: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:30:36 GMT",
"content-type":"application/json; charset=UTF-8",
"content-length":"884",
{
  "adapters":[
    {
      "adapter-family":"osa",
      "adapter-id":"18C",
      "name":"OSD 018C Z15B-04",
      "object-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338",
      "status":"active",
      "type":"osd"
    },
    {
      "adapter-family":"crypto",
      "adapter-id":"1C4",
      "name":"Crypto 01C4 Z15B-21",
      "object-uri":"/api/adapters/f7956dc4-c930-11e5-a978-020000000338",
      "status":"not-active",
      "type":"crypto"
    },
    {
      "adapter-family":"ficon",
      "adapter-id":"141",
      "name":"FCP 0141 Z22B-07",
      "object-uri":"/api/adapters/d71902a4-c930-11e5-a978-020000000338",
      "status":"active",
      "type":"fcp"
    }
  ]
}
```

Figure 117. List Adapters of a CPC: Response

Get Adapter Properties

The Get Adapter Properties operation retrieves the properties of a single Adapter object that is designated by its **object-id**.

HTTP method and URI

```
GET /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object for which properties are to be obtained.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Adapter object as defined in the [“Data model” on page 284](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the adapter object as defined in the [“Data model” on page 284](#).

If the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose **object-id** is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 298](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	An adapter with object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/adapters/d71902a4-c930-11e5-a978-020000000338
x-api-session:
  34ccosly7i88xnu1jbp19ekpdnar1hwxujg62ots98p3rrvjoy

<no request body>
```

Figure 118. Get Adapter Properties: Request

```
200 OK
{"server":"zSeries management console API web server / 2.0",
 "cache-control":"no-cache",
 "date":"Wed, 10 Feb 2016 19:34:11 GMT",
 "content-type":"application/json; charset=UTF-8",
 "content-length":"670",
 {
  "adapter-family":"ficon",
  "adapter-id":"141",
  "allowed-capacity":32,
  "card-location":"Z22B-D207-J.01",
  "channel-path-id":"01",
  "class":"adapter",
  "configured-capacity":1,
  "description":"",
  "detected-card-type":"ficon-express-8",
  "maximum-total-capacity":255,
  "name":"FCP 0141 Z22B-07",
  "object-id":"d71902a4-c930-11e5-a978-020000000338",
  "object-uri":"/api/adapters/d71902a4-c930-11e5-a978-020000000338",
  "parent":"/api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29",
  "physical-channel-status":"operating",
  "port-count":1,
  "state":"online",
  "status":"active",
  "storage-port-uris":[
    "/api/adapters/d71902a4-c930-11e5-a978-020000000338/storage-ports/0"
  ],
  "type":"fcp",
  "used-capacity":1
 }
```

Figure 119. Get Adapter Properties: Response

Update Adapter Properties

The Update Adapter Properties operation updates one or more of the writable properties of an adapter.

HTTP method and URI

```
POST /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object for which properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the [“Data model” on page 284](#). The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Fields are only valid if they are supported for an adapter of the targeted type. For instance, an adapter with **type** property "**fcp**" may define a **description** property (all types) but not a **maximum-transmission-unit-size** property (HiperSockets only).

Description

This operation updates an adapter's properties with the values specified.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an Adapter object to which the API user has object-access permission. If the adapter **status** is "**definition-error**", a 409 (Conflict) status code is returned; in this case either a valid card must be installed or the adapter must have its configuration removed and a new configuration added.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the adapter's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 131. Update Adapter Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An adapter with the name specified in the request body already exists on its parent CPC.
	18	An updated property is not valid for an adapter's adapter-family .
	420	A crypto adapter for the CPC is already configured with the given crypto-number .
403 (Forbidden)	1	API user does not have action permission to the Adapter Details task.
404 (Not Found)	1	An adapter with the object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.

Table 131. Update Adapter Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	Adapter status is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	Adapter object with the object ID { <i>adapter-id</i> } was busy and request timed out.
	420	The given allowed-capacity value is not valid because it is less than the current used-capacity value.
	421	The tke-commands-enabled property cannot be updated when the CPC has not been started.
	422	The tke-commands-enabled property cannot be updated when the current crypto-type value is " cca-coprocessor ".
	423	The tke-commands-enabled property cannot be updated when the current status value is " exceptions ".
	424	The tke-commands-enabled property cannot be updated when the current status value is " not-active ".
	425	The tke-commands-enabled property cannot be updated when the current adapter-state value is " online ".
	429	The channel-path-id property cannot be updated when the adapter is configured to any partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/adapters/d71902a4-c930-11e5-a978-0200000000338
x-api-session:1nnqtc5mo3yntwilbk96er2jekdekba2nvv3ivphfvmdqym8po
content-type: application/json
content-length: 33
{
  "description":"My FCP adapter"
}
```

Figure 120. Update Adapter Properties: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:35:11 GMT",

<no response body>
```

Figure 121. Update Adapter Properties: Response

Change Crypto Type

The Change Crypto Type operation reconfigures a cryptographic adapter to a different crypto type. This operation is only supported for cryptographic adapters.

HTTP method and URI

POST /api/adapters/{*adapter-id*}/operations/change-crypto-type

In this request, the URI variable {*adapter-id*} is the object ID of the target cryptographic Adapter object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
crypto-type	String Enum	Required	The value to be set as the cryptographic adapter's crypto-type property. Values: • "accelerator" - Crypto Express5S Accelerator. • "cca-coprocessor" - Crypto Express5S Coprocessor. • "ep11-coprocessor" - Crypto Express5S EP11 Coprocessor.
zeroize	Boolean	Optional	Specifies whether the cryptographic adapter will be zeroized when it is reconfigured to a crypto-type of "accelerator" . This field is only valid when crypto-type is "accelerator" . Default: true

Description

This operation reconfigures a cryptographic adapter to a new crypto type. For a cryptographic adapter that is becoming an accelerator, the adapter may optionally be zeroized as part of the reconfiguration process. A cryptographic adapter must be varied offline before its crypto type can be changed.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** {*adapter-id*} does not identify a crypto adapter object to which the API user has object-access permission. If the given crypto-type is not valid, a 400 (Bad Request) status code is returned. If the adapter's **adapter-family** is not **"crypto"**, a 404 (Not Found) status code is returned. If the adapter is online (**state** is **"online"**), a 409 (Conflict) status code is returned. If the CPC that contains the cryptographic adapter has not been started, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the cryptographic adapter's crypto type is changed to the new value specified in the request body.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is {*adapter-id*}
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 132. Change Crypto Type: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The crypto-type value in the request body is not valid.
	15	The zeroize value in the request body is not applicable with this crypto-type .
403 (Forbidden)	1	The API user does not have action/task permission to the Adapter Details task.
404 (Not Found)	1	An adapter with object ID { <i>adapter-id</i> } does not exist on the HMC or the API user does not have object-access permission to it.
	4	The operation does not support an adapter of the given adapter-family .
409 (Conflict)	1	Adapter status is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	Adapter object with ID { <i>adapter-id</i> } is not started.
	421	The CPC containing the adapter with object ID { <i>adapter-id</i> } is not started.
	426	Adapter adapter-state is "online".
	428	The adapter is already configured to be the requested crypto-type .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/adapters/f7956dc4-c930-11e5-a978-020000000338/operations/change-crypto-type
x-api-session: 4polovirzt2a8qts11nt5ea7mr1it9q1bhez3h3ukh7wvnb8e4
content-type: application/json
content-length: 35
{
  "crypto-type": "ep11-coprocessor"
}
```

Figure 122. Change Crypto Type: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:38:40 GMT",

<no response body>
```

Figure 123. Change Crypto Type: Response

Create Hipersocket

The `Create Hipersocket` operation creates and configures a `HiperSockets` adapter (**type** is **"hipersockets"**).

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/adapters
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the adapter's name property. The name value must be unique among all other adapters owned by the targeted CPC.
description	String (0-1024)	Optional	The value to be set as the adapter's description property. Default value: An empty string.
port-description	String (0-1024)	Optional	The value to be set as the description property of the HiperSocket's single network-port. Default value: An empty string.
maximum-transmission-unit-size	Integer Enum	Optional	The value to be set as the adapter's maximum-transmission-unit-size property. Default value: 8

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri of the newly created Adapter object.

Description

This operation creates and configures a HiperSocket with the values specified on the identified CPC and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to **Create HiperSockets Adapter** task, a 403 (Forbidden) status code is returned. If the **object-id** `{cpc-id}` does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the CPC identified by `{cpc-id}` already contains an adapter with the specified **name**, a 400 (Bad Request) status code is

returned. If the CPC identified by *{cpc-id}* is not enabled for DPM, a 409 (Conflict) status code is returned. If the CPC identified by *{cpc-id}* already contains the number of HiperSockets identified by its **maximum-hipersockets** property, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the HiperSocket is created and its properties are configured to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Create HiperSockets Adapter** task.
- Object-access permission to the CPC whose object ID is *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 304.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 133. Create Hipersocket: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An adapter with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the action/task permission to the Create HiperSockets Adapter task.
404 (Not Found)	1	A CPC with object ID <i>{cpc-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	5	A CPC with object ID <i>{cpc-id}</i> is not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	427	A CPC with object ID <i>{cpc-id}</i> already has the maximum number of HiperSockets adapters (identified by its maximum-hipersockets property) defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29/adapters
x-api-session: 3mwtrej0fkg719l3jr5tvxyl2fd6h35que4x8jrukc17l1802
content-type: application/json
content-length: 59
{
  "name": "hiper1",
  "description": "My hipersocket adapter"
}
```

Figure 124. Create Hipersocket: Request

```
201 Created
"server": "zSeries management console API web server / 2.0",
"cache-control": "no-cache",
"date": "Wed, 10 Feb 2016 19:41:57 GMT",
"location": "/api/adapters/542b9406-d033-11e5-9f39-020000000338",
"content-type": "application/json; charset=UTF-8",
"content-length": "67",
{
  "object-uri": "/api/adapters/542b9406-d033-11e5-9f39-020000000338"
}
```

Figure 125. Create Hipersocket: Response

Delete Hipersocket

The Delete Hipersocket operation deletes a HiperSocket adapter.

HTTP method and URI

```
DELETE /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the adapter to be deleted.

Description

This operation deletes a HiperSocket adapter. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to **Delete HiperSocket**, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission. If the adapter is not a HiperSocket (**adapter-family** is not **"hipersockets"**), a 404 (Not Found) is returned. If the adapter is currently configured to any partition, a 409 (Conflict) is returned.

This operation deletes the identified adapter, and removes the HiperSocket adapter from the CPC.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Delete HiperSocket** task.
- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 134. Delete Hipersocket: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Delete Hipersocket task.
404 (Not Found)	1	An adapter with object-id { <i>adapter-id</i> } does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type .
409 (Conflict)	2	Adapter object with ID { <i>adapter-id</i> } was busy and request timed out.
	430	The Hipersocket adapter cannot be deleted because it is currently assigned to one or more partitions.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/adapters/542b9406-d033-11e5-9f39-020000000338
x-api-session:
  62ykfwk1ds6raibwt03j6ta8g146q06hjnevovt2is3lplwl
<no request body>
```

Figure 126. Delete Hipersocket: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:42:51 GMT",
<no response body>
```

Figure 127. Delete Hipersocket: Response

Get Partitions Assigned to Adapter

The `Get Partitions Assigned to Adapter` operation lists the partitions to which the adapter is configured to provide I/O and virtual functions. This operation is not supported for adapters whose **type** is **"osm"**.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/operations/get-partitions-assigned-to-adapter
```

In this request, the URI variable {*adapter-id*} is the object ID of the Adapter object whose partitions should be returned.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern to limit returned Partition objects to those that have a matching name property
status	String Enum	Optional	Filter string to limit returned Partition objects to those that have a matching status property. Value must be a valid partition status property value.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
partitions-assigned-to-adapter	Array of partition-info objects	Array of partition-info objects, described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the Partition object.
name	String	The current value of the name property of the partition.
status	String Enum	The current value of the status property of the Partition object.

Description

The Get Partitions Assigned to Adapter operation lists the partitions to which the adapter is configured. The object URI, display name, and status information are provided for each.

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid value for the **status** property according to the partition data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have the specified **status** value. If the **status** parameter is omitted, this filtering is not done.

A 404 (Not Found) status code is also returned if the **object-id** {*adapter-id*} does not identify an adapter object to which the API user has object-access permission. If the adapter's **type** is "osm" a 404 (Not Found) status code is returned.

If no partitions are to be included in the results due to filtering (or no partitions exist), an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose object ID is {*adapter-id*}.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 308](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 135. Get Partitions Assigned to Adapter: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A status query parameter defines an invalid value.
404 (Not Found)	1	An adapter with object-id { <i>adapter-id</i> } does not exist on the HMC or the API user does not have object-access permission for it.
	4	An adapter with type of "osm" cannot be assigned to any partitions.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/adapters/d71902a4-c930-11e5-a978-020000000338/operations/get-partitions-
assigned-to-adapter
x-api-session:
  2gr3xkfjj6991vvlwpmnkys8814y68i4anucrudo90ytylfuo5
<no request body>
```

Figure 128. Get Partitions Assigned to Adapter: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:47:42 GMT",
"content-type":"application/json;charset=UTF-8",
"content-length":"140",
{
  "partitions-assigned-to-adapter":[
    {
      "name":"MyPartition",
      "object-uri":"/api/partitions/c0430acc-c9c9-11e5-be4f-020000000338",
      "status":"active"
    }
  ]
}
```

Figure 129. Get Partitions Assigned to Adapter: Response

Get Network Port Properties

The Get Network Port Properties operation retrieves the properties of a single Network Port element object.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/network-ports/{network-port-id}
```

In this request, the URI variable {*adapter-id*} is the object ID of the Adapter object and the URI variable {*network-port-id*} is the element ID of the Network Port object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the network port object as defined in the [“Data model” on page 284](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the network port object as defined in the [“Data model” on page 284](#).

A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the **element-id** *{network-port-id}* does not identify a network port in the adapter. If the adapter **type** is not **"osd"**, **"roce"** or **"hipersockets"**, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 310](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 136. Get Network Port Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	An adapter with object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type.
	5	A network port with element-id <i>{network-port-id}</i> does not exist for the adapter on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0
x-api-session:
  54dwyb1qutbd5ck8ultl1mh752r7lfi3ljbk8pt9814e6j1g1wu
<no request body>
```

Figure 130. Get Network Port Properties: Request

```

200 OK
{"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:51:27 GMT",
"content-type":"application/json; charset=UTF-8",
"content-length":"229",
{
  "class":"network-port",
  "description":"",
  "element-id":"0",
  "element-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0",
  "index":0,
  "name":"Port 0",
  "parent":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338"
}
}

```

Figure 131. Get Network Port Properties: Response

Update Network Port Properties

The Update Network Port Properties operation updates one or more of the writable properties of a Network Port object. This operation only supports configured adapters whose **type** is **"osd"**, **"osm"**, **"roce"**, or **"hipersockets"**.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/network-ports/{network-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{network-port-id}* is the element ID of the Network Port object.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the ["Data model"](#) on page 284. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a network port's properties with the values specified.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the element-id *{network-port-id}* does not identify a network port in the adapter. If the adapter **type** is not **"osd"**, **"roce"** or **"hipersockets"**, a 404 (Not Found) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported.

If the request body contents are valid, the network port's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 137. Update Network Port Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the Adapter Details task.
404 (Not Found)	1	An adapter with the object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type.
	5	A network port with element-id <i>{network-port-id}</i> does not exist for the adapter on the HMC.
409 (Conflict)	2	Adapter object with ID <i>{adapter-id}</i> was busy and request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0
x-api-session: 3y0dr1g0lob607lzurmcwggc2lu4vbjjjiskngz26uno8pa8cz
content-type: application/json
content-length: 30
{
  "description": "My OSA port"
}
```

Figure 132. Update Network Port Properties: Request

```
204 No Content
"server": "zSeries management console API web server / 2.0",
"cache-control": "no-cache",
"date": "Wed, 10 Feb 2016 19:52:01 GMT",

<no response body>
```

Figure 133. Update Network Port Properties: Response

Get Storage Port Properties

The Get Storage Port Properties operation retrieves the properties of a single Storage Port element object.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{storage-port-id}* is the element ID of the Storage Port object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the storage port object as defined in the “Data model” on page 284. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage port object as defined in the “Data model” on page 284.

A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the **element-id** *{storage-port-id}* does not identify a storage port in the adapter. If the adapter **type** is not “fcp”, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 313.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 138. Get Storage Port Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	An adapter with the object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	5	A storage port with element-id of <i>{storage-port-id}</i> does not exist for the adapter on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
GET /api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0 HTTP/1.1
x-api-session: 46jg2ldhqd5k27ty3owaiipupfkvmy3bksas8ih3j4wp0exwk
```

Figure 134. Get Storage Port Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 19:07:51 GMT
content-type: application/json;charset=UTF-8
content-length: 361
{
  "class": "storage-port",
  "connection-endpoint-class": "storage-switch",
  "connection-endpoint-uri": "/api/storage-switches/09f4f570-9fe9-11e8-8c0c-fa163e3c2af4",
  "description": "",
  "element-id": "0",
  "element-uri": "/api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
  "index": 0,
  "name": "Port 0",
  "parent": "/api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4"
}
```

Figure 135. Get Storage Port Properties: Response

Update Storage Port Properties

The Update Storage Port Properties operation updates one or more of the writable properties of a storage port. This operation only supports configured adapters whose **type** is **"fcp"**.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{storage-port-id}* is the element ID of the Storage Port object.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage port's properties with the values specified.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the **element-id** *{storage-port-id}* does not identify a storage port in the adapter. If the adapter **type** is not **"fcp"**, a 404 (Not Found) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported.

If the **connection-endpoint-uri** field references a storage subsystem when one or more storage fabrics are defined, or if the **connection-endpoint-uri** field references a storage switch or storage subsystem that does not reside in the same CPC as the target storage port, a 409 (Conflict) status code is returned.

If the request body contents are valid, the storage port's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 139. Update Storage Port Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the Adapter Details task.
404 (Not Found)	1	An adapter with the object ID <i>{adapter-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type.
	5	A storage port with element-id of <i>{storage-port-id}</i> does not exist for the adapter on the HMC.
409 (Conflict)	2	Adapter object with the ID <i>{adapter-id}</i> was busy and the request timed out
	441	The storage switch or storage subsystem referenced by the connection-endpoint-uri field resides in a different CPC than the targeted storage adapter port.
	455	A direct endpoint connection cannot be made to a storage subsystem when storage fabrics are defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/adapters/d71902a4-c930-11e5-a978-0200000000338/storage-ports/0
x-api-session: 4sw8s1089cczsejp5vzonbleyrvwsmx6edkqdudskevrcp0bvf
content-type: application/json
content-length: 30
{
  "description": "My FCP port"
}
```

Figure 136. Update Storage Port Properties: Request

```
204 No Content
"server": "zSeries management console API web server / 2.0",
"cache-control": "no-cache",
"date": "Wed, 10 Feb 2016 19:50:05 GMT",

<no response body>
```

Figure 137. Update Storage Port Properties: Response

Change Adapter Type

The Change Adapter Type operation reconfigures an adapter from one type to another. Currently, only storage adapters can be reconfigured. Storage adapter instances represent daughter cards on a physical storage card. Current storage cards require both daughter cards to be configured to the same protocol, so changing the configuration of the targeted adapter will also change the configuration of the adapter instance that represents the other daughter card on the same physical adapter. API clients that need to determine the related adapter instance can do so by finding the storage adapter card with a matching first 9 characters (card ID and slot ID) of their **card-location** property values.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/operations/change-adapter-type
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
type	String Enum	Required	The value to be set as the adapter's type property. Values: • "fcp" - Fibre Channel attached storage resource. • "fc" - Fibre Connection attached storage. • "not-configured" - The adapter is not configured.

Description

This operation reconfigures a storage adapter to support a new storage protocol.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{adapter-id}* does not identify an Adapter object on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the current value of the adapter's **adapter-family** property is not **"ficon"** or its **detected-card-type**

property value is **"ficon-express-32s"**. If the value of the adapter's **status** property is **"exceptions"**, or if the current value of the adapter's **type** property is already set to the value specified in the **type** field in the request body, or if the adapter is configured to any partition, or the adapter's current **type** is **"fc"** and it, or the adapter on the same physical card, is connected to a storage switch, a 409 (Conflict) status code is returned.

If the request body contents are valid, the adapter's **type** property and the **type** property of the adapter with the same card id and slot id, as identified by the first 9 characters of their **card-location** property values, are updated to the value specified in the **type** field in the request body. The two adapters are reconfigured to the new storage protocol.

Property change notifications for the **type** property of both affected adapters are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is {*adapter-id*}
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 140. Change Adapter Type: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	18	The operation is not allowed for the adapter's adapter-family or detected-card-type .
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	An adapter with object ID { <i>adapter-id</i> } does not exist on the HMC.
409 (Conflict)	1	Adapter status is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	The adapter object with the object-id { <i>adapter-id</i> } was busy and the request timed out.
	488	The adapter's type property is already set to the value of the type field in the request body.
	489	The adapter is configured to at least one partition.
	496	The adapter with object ID { <i>adapter-id</i> }, or the adapter on the same physical card, has an endpoint connection to a storage switch.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/adapters/e835bb78-9fe7-11e8-bc9a-fa163e3c2af4/operations/change-adapter-
type HTTP/1.1
x-api-session: 2fypcb9w0z7kzv9rtq7yfxaypayrdh209nmrrdyv1jetvhxfu0
content-type: application/json
content-length: 15
{
  "type": "fcp"
}
```

Figure 138. Change Adapter Type: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 15 Aug 2018 17:02:33 GMT

<No response body>
```

Figure 139. Change Adapter Type: Response

Inventory service data

Information about the Adapters managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Adapter objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "adapter" are to be included. Information for a particular adapter is included only if the API user has object-access permission to that object.

For each adapter to be included, the inventory response array includes the following:

- An array entry for the Adapter object itself. This entry is a JSON object with the same contents as is specified in the response body contents section for [“Get Adapter Properties”](#) on page 298. That is, the data provided is the same as would be provided if a Get Adapter Properties operation were requested targeting this object.
- An array entry for each network port associated with the adapter. For each such network port, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for [“Get Network Port Properties”](#) on page 309.
- An array entry for each storage port associated with the adapter. For each such storage port, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for [“Get Storage Port Properties”](#) on page 313.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a single adapter. This object would appear as a sequence of array entries in the response array:

```
{
  "adapter-family": "osa",
  "adapter-id": "18C",
  "allowed-capacity": 1920,
  "card-location": "Z15B-D104-J.01",
  "channel-path-id": "10",
  "class": "adapter",
  "configured-capacity": 3,
  "description": "My OSA",
  "detected-card-type": "osa-express-5s-10gb",
  "maximum-total-capacity": 1920,
  "name": "OSD 018C Z15B-04",
  "network-port-uris": [
    "/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0"
  ],
  "object-id": "e77d39f8-c930-11e5-a978-020000000338",
  "object-uri": "/api/adapters/e77d39f8-c930-11e5-a978-020000000338",
  "parent": "/api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29",
  "physical-channel-status": "operating",
  "port-count": 1,
  "state": "online",
  "status": "active",
  "type": "osd",
  "used-capacity": 3
},
{
  "class": "network-port",
  "description": "My OSA port",
  "element-id": "0",
  "element-uri": "/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0",
  "index": 0,
  "name": "Port 0",
  "parent": "/api/adapters/e77d39f8-c930-11e5-a978-020000000338"
},
}
```

Figure 140. Adapter object: Sample inventory data

Virtual Switch object

A Virtual Switch object is a virtualized representation of a CPC's networking adapter and port. Network adapters without a physical port, such as HiperSockets or single port OSAs are virtualized to a single virtual switch. Network adapters with multiple ports are virtualized into multiple virtual switches one for each port. Virtual switches are generated automatically every time a new network adapter is detected and configured. The virtual switch serves as the connection point for network interfaces (VNICs) created by the virtual server administrator.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

This object includes the properties defined in the [“Base managed object properties schema” on page 70](#), but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status. The following class-specific specializations apply to the other base managed object properties:

Table 141. Virtual Switch object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path for a Virtual Switch object is of the form <code>/api/virtual-switches/{vswitch-id}</code> where <code>{vswitch-id}</code> is the value of the object-id property of the Virtual Switch object.

Table 141. Virtual Switch object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
object-id	—	String (36)	The unique identifier for the virtual switch instance.
parent	—	String/ URI	The canonical URI path of the CPC object.
class	—	String (14)	The class of a Virtual Switch object is "virtual-switch" .
name	(w)(pc)	String (1-64)	The display name of the Virtual Switch object. This name must be unique among all of the CPC's virtual switches, and it must conform to the length and character requirements of the name property described in "Base managed object properties schema" on page 70. Default: A string of the form <i>{PCHID}.{portNumber}.{type}</i> where <i>{PCHID}</i> is the PCHID of the backing adapter, <i>{portNumber}</i> is the value of the port property, and <i>{type}</i> is the abbreviated type property folded to uppercase. For example, "019F.0.OSD".
description	(w)(pc)	String (0-1024)	The description of the Virtual Switch object, or an empty string. Default: empty string.

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 142. Virtual Switch object: class specific properties

Name	Qualifier	Type	Description
type	—	String Enum	"hipersockets" - a HyperSockets virtual switch. "osd" - OSA Direct Express virtual switch.
backing-adapter-uri	—	String/ URI	The canonical URI path of the backing Adapter object.
port	—	Integer	Physical port identifier associated with the virtual switch. Valid port numbers are 0 and 1. Network adapters that do not have a physical port (HyperSockets) are considered to have a single "virtual" port that is identified as port number 0.
connected-vnic-uris	(p)(pc)(c)	Array of String/ URI	The list of network interfaces (VNICs) connected to this virtual switch. The list is available via the Get Connected VNICs of a Virtual Switch operation. Each element in this array is a canonical URI path for a NIC element of a Partition object. This array is initially empty when the virtual switch is created.

List Virtual Switches of a CPC

The List Virtual Switches of a CPC operation lists virtual switches that are defined to the CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/virtual-switches
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC for which virtual switches are to be listed.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those objects that have a matching name property
type	String Enum	Optional	Filter string to limit returned objects to those objects that have a matching type property. The value must be a valid type property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
virtual-switches	Array of objects	Array of nested virtual-switch-info objects as described in the next table.

Each nested virtual-switch-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the Virtual Switch object.
name	String	The name property of the Virtual Switch object
type	String Enum	The type property of the Virtual Switch object

Description

This operation lists Virtual Switches defined to the CPC. Some basic properties are provided for each virtual switch that is included in the response.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

If the name query parameter is specified, the returned list is limited to those virtual switches that have a **name** property matching the specified filter pattern. If the name parameter is omitted, no such filtering is performed.

If the type query parameter is specified, the parameter is validated to ensure that it is a valid virtual switch **type** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those virtual switches that have a **type** property matching the specified value. If the type parameter is omitted, no such filtering is performed.

A virtual switch is included in the list only if the API user has object-access permission to the backing adapter of that virtual switch. If there is a virtual switch to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If the CPC is not in DPM mode, or there are no virtual switches defined to the CPC, or no virtual switches are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object identified in the request URI
- Object-access permission to the backing adapter of the Virtual Switch objects to be included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 321](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or it designates a resource for which the API user does not have object-access permission.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef/virtual-switches
x-api-session: 1jgbldxyy2inf0p7aa9a8p87j7awxsrlmstmfnw07hvoaz8da
```

Figure 141. List Virtual Switches of a CPC: Request


```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Mar 2015 06:20:54 GMT
content-type: application/json; charset=UTF-8
content-length: 255
{
  "virtual-switches": [
    {
      "name": "PrimeIQDVSwitch1",
      "object-uri": "/api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058",
      "type": "hipersockets"
    },
    {
      "name": "5F1.P0.OSD",
      "object-uri": "/api/virtual-switches/df0b71c-d491-11e4-a555-020000003058",
      "type": "osd"
    }
  ]
}

```

Figure 142. List Virtual Switches of a CPC: Response

Get Virtual Switch Properties

The Get Virtual Switch Properties operation retrieves the properties of a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

```
GET /api/virtual-switches/{vswitch-id}
```

In this request, the URI variable *{vswitch-id}* is the object ID of the Virtual Switch object for which properties are to be returned.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Virtual Switch object as defined in the “Data model” on page 319. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current property values for the Virtual Switch object specified by *{vswitch-id}*.

On successful execution, all of the current properties as defined in the “Data model” on page 319 for the Virtual Switch object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Virtual Switch object and the API user must have object-access permission to the backing Adapter object of the virtual switch. If either of these conditions is not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified in the request URI

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 323.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the request URI <i>{vswitch-id}</i> does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058
x-api-session: 5an6scz1o7mikkmyeew077vj9yggo5qe781wgegwpw3yilxq4j
```

Figure 143. Get Virtual Switch Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Mar 2015 07:00:51 GMT
content-type: 'application/json; charset=UTF-8'
content-length: 306
{
  "backing-adapter-uri": "/api/adapters/f718c7a0-d490-11e4-a555-020000003058",
  "class": "virtual-switch",
  "description": "",
  "name": "PrimeIQDVSwitch1",
  "object-id": "f6b4c70e-d491-11e4-a555-020000003058",
  "object-uri": "/api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058",
  "parent": "/api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef",
  "port": 0,
  "type": "hipersockets"
}
```

Figure 144. Get Virtual Switch Properties: Response

Get Connected VNICs of a Virtual Switch

The Get Connected VNICs of a Virtual Switch operation retrieves the list of network interfaces (VNICs) connected to a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

```
GET /api/virtual-switches/{vswitch-id}/operations/get-connected-vnics
```

In this request, the URI variable *{vswitch-id}* is the object ID of the Virtual Switch object whose VNIC list is to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the **connected-vnic-uris** property of the Virtual Switch object as defined in the [“Data model” on page 319](#).

Description

This operation returns the list of VNICs connected to the Virtual Switch object specified by *{vswitch-id}*.

On successful execution, the current list of VNICs connected to the Virtual Switch is provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Virtual Switch object and the API user must have object-access permission to the backing Adapter object of the Virtual Switch object. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified by the request URI.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 324](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the request URI (<i>{vswitch-id}</i>) does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058/operations/get-connected-  
vnic  
x-api-session: 5an6scz1o7mikkmyeew077vj9yggo5qe781wgegwpw3yilxq4j
```

Figure 145. Get Connected VNICs of a Virtual Switch: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Mar 2015 07:00:51 GMT
content-type: 'application/json; charset=UTF-8'
content-length: 370
{
  "connected-vnic-uris": [
    "/api/partitions/675fe728-dfc7-11e4-8582-020000003022/nics/0bc27850-dfc9-11e4-a45e-020000003022",
    "/api/partitions/675fe728-dfc8-11e4-8583-020000003022/nics/0bc27850-dfc9-11e4-a45e-020000003022",
    "/api/partitions/684fe825-dfc9-11e4-8681-020000003022/nics/0ba26450-dfc8-11e4-a36e-020000003022"
  ],
}

```

Figure 146. Get Connected VNICs of a Virtual Switch: Response

Update Virtual Switch Properties

The Update Virtual Switch Properties operation updates the properties of a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

POST `/api/virtual-switches/{vswitch-id}`

In this request, the URI variable `{vswitch-id}` is the object ID of the Virtual Switch object for which properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

The Update Virtual Switch Properties operation updates writable properties of the Virtual Switch object specified by `{vswitch-id}`.

The URI path must designate an existing Virtual Switch object, and the API user must have object-access permission to the backing Adapter object of the Virtual Switch object. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Adapters** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in [“Request body contents” on page 326](#). If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified in the request URI
- Action/task permission to the **Manage Adapters** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 143. Update Virtual Switch Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the Manage Adapters task.
404 (Not Found)	1	The object ID <code>{vswitch-id}</code> does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058
x-api-session: rxkrzf6v287nn0klyk4zah94i91r49eskkut8242kdqfa9so
content-type: application/json
content-length: 68
{
  "description": "the neNetwork virtual switch",
  "name": "neVswitch"
}
```

Figure 147. Update Virtual Switch Properties: Request

```
204 No Content
date: Tue, 31 Mar 2015 07:05:50 GMT
server: zSeries management console API web server / 2.0
cache-control: no-cache

<No response body>
```

Figure 148. Update Virtual Switch Properties: Response

Inventory service data

Information about the virtual switches can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Virtual Switch objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by inventory class, implicitly via a containing category, or by default) that objects of the class **"virtual-switch"** to be included. An entry for a particular

virtual switch is included only if the API user has access permission to that object as described in the [Get Virtual Switch Properties](#) operation.

For each Virtual Switch object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Virtual Switch Properties”](#) on [page 323](#). That is, the data provided is the same as would be provided if a [Get Virtual Switch Properties](#) operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the [Get Inventory](#) response to describe a single virtual switch. This object would appear as one array entry in the response array:

```
{
  "backing-adapter-uri": "/api/adapters/df863694-08b1-11e6-91a3-42f2e90df693",
  "class": "virtual-switch",
  "description": "",
  "name": "5F1.P0.OSD",
  "object-id": "dff0b71c-d491-11e4-a555-020000003058",
  "object-uri": "/api/virtual-switches/dff0b71c-d491-11e4-a555-020000003058",
  "parent": "/api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef",
  "port": 0,
  "type": "osd"
}
```

Figure 149. Virtual Switch object: Sample inventory data - Response

Capacity Group element object

A Capacity Group is an element object of a CPC that is in DPM mode. It consists of a set of partitions and specifies the absolute processor cap per processor type for that set of partitions. The defined absolute processor cap for a specific processor type dictates the total amount of the specified processing capacity that the active partitions in the group can consume at any time. The absolute processor cap is specified as a value between 0.01 and 255.0, where a value of 1.00 represents the processing capacity provided by one processor.

The scope of a Capacity Group is within a single CPC and it is required that the CPC is in DPM mode for a Capacity Group to be created.

The absolute processor cap can be modified after the group is created, at which time, the partitions that are already active in the group will be affected.

Data model

The following attributes are identified for the Capacity Group element object:

Table 144. Capacity Group element object properties			
Name	Qualifier	Type	Description of specialization
element-uri	—	String/ URI	The canonical URI path of the Capacity Group element object is of the form <code>/api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}</code> where <code>{capacity-group-id}</code> is the unique identifier for a capacity group within a CPC.
element-id	—	String (36)	The unique identifier for the capacity group. The element-id is in the form of a UUID.
parent	—	String/ URI	The canonical URI path of the parent CPC object.

Table 144. Capacity Group element object properties (continued)

Name	Qualifier	Type	Description of specialization
class	—	String (14)	The class of a Capacity Group object is " capacity-group ".
name	(w)(pc)	String (1-64)	The name of the capacity group. The name must be unique among all capacity groups in the CPC. The length and character requirements on this property are the same as those of the name property described in the " Base managed object properties schema " on page 70.
short-name	(w)(pc)	String (1-8)	The name of the capacity group that will be presented to the guest OS, when capping is enabled on this capacity group. Only alpha-numeric uppercase characters are allowed. The short-name must be unique among all capacity groups in the CPC. The words PHYSICAL, REC, SYSTEM, and PRIM $nnnn$ (where $nnnn$ is a 4-digit number) are reserved and cannot be used.
description	(w)(pc)	String (0-1024)	The description of the capacity group. Default: An empty string.
capping-enabled	(w)(pc)	Boolean	This indicates if capping is enabled for this capacity group or not. If set to true, the partitions in this capacity group are capped according to the values set. If set to false, any specified cap values are not effective and the partitions in this capacity group are not capped. Default: True.
absolute-general-purpose-proc-cap	(w)(pc)	Float (0-255)	The limit on the absolute capacity of general purpose processors that the partitions in this group are allowed to consume at any point in time. It is expressed in the units of processors. The range of valid values is between 0.0 and 255.0 and in the increments of 0.01. Default: 0.0. The value of 0.0 indicates that the general purpose processors are not capped for the partitions in this capacity group. Exactly one of the processor types must have a value other than 0.0.
absolute-ifl-proc-cap	(w)(pc)	Float (0-255)	The limit on the absolute capacity of IFL processors that the partitions in this group are allowed to consume at any point in time. It is expressed in the units of processors. The range of valid values is between 0.0 and 255.0 and in the increments of 0.01. Default: 0.0. The value of 0.0 indicates that the IFL processors are not capped for the partitions in this capacity group. Exactly one of the processor types must have a value other than 0.0.

Table 144. Capacity Group element object properties (continued)

Name	Qualifier	Type	Description of specialization
partition-uris	(c)(pc)	Array of String/ URIs	Array of URIs of the partitions that belong to this capacity group. Each partition in this capacity group must have the type of processor (general purpose or IFL) that is being capped by this capacity group. This property can be modified using the Add Partition to Capacity Group and Remove Partition from Capacity Group operations.

List Capacity Groups of a CPC

The List Capacity Groups of a CPC operation lists the defined capacity groups for the specified CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-groups
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
capacity-groups	Array of capacity-group-info objects	Array of capacity-group-info objects, described in the next table. Returned array may be empty.

Each nested capacity-group-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Capacity Group object. More details about this capacity group can be fetched using this element-uri .
name	String (1-64)	The name of the capacity group.

Description

This operation lists the capacity groups that are defined for the specified CPC. The **element-uri** and **name** are returned for each capacity group.

If the **name** query parameter is specified for the request, the returned list is limited to the Capacity Group elements that have a name matching the specified filter pattern. If no match is found, then the response will be an empty array. If the parameter is not specified, all the Capacity Group elements are returned.

The response could be an empty array, if the CPC does not have a Capacity Group associated with it or if the CPC is not in DPM mode.

The URI path must designate an existing CPC, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the specified CPC.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 330](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not exist or the user does not have object-access permission to it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups HTTP/1.1
x-api-session: 4xjbyzz8iip3fv0j77gyuk3e6r7lp0pl9yb4zfpf1n46u3z7ec
```

Figure 150. List Capacity Groups of a CPC: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 22 Feb 2016 10:00:54 GMT
content-type: application/json; charset=UTF-8
content-length: 363
{
  "capacity-groups": [
    {
      "element-uri": "/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/e0118d40-d088-11e5-a631-42f2e9ef1641",
      "name": "Test18311"
    },
    {
      "element-uri": "/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/d95f6418-d6eb-11e5-92b9-42f2e9ef1641",
      "name": "myNewGroup1"
    }
  ]
}

```

Figure 151. List Capacity Groups of a CPC: Response

Create Capacity Group

The Create Capacity Group operation creates a capacity group for the specified CPC.

HTTP method and URI

POST /api/cpcs/{cpc-id}/capacity-groups

In this request, the URI variable {cpc-id} is the object ID of the CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the Capacity Group's name property.
short-name	String (1-8)	Optional	The value to be set as the Capacity Group's short-name property. If a short-name is not specified, it will be auto-generated and assigned to this capacity group. The name, if specified, must be unique within the CPC.
description	String (0-1024)	Optional	The value to be set as the Capacity Group's description property.
capping-enabled	Boolean	Optional	The value to be set as the capacity group's capping-enabled property.
absolute-general-purpose-proc-cap	Float	Optional	The value to be set as the Capacity Group's absolute-general-purpose-proc-cap property.
absolute-ifl-proc-cap	Float	Optional	The value to be set as the Capacity Group's absolute-ifl-proc-cap property.

Response body contents

On successful completion, HTTP status code 201 (Created) is returned and a JSON object with the following field is also provided. The **element-uri** of the capacity group created is also available in the Location header of the response.

Field name	Type	Description
element-uri	String/ URI	The element-uri property of the created capacity group.

Description

This operation creates a capacity group with the specified attributes. It is required that the CPC designated by *{cpc-id}* is in DPM mode. On successful execution, the **element-uri** of the created capacity group is returned in the response body and in the **Location** response header. An Inventory Change notification is emitted asynchronously to this operation.

Exactly one processor type must have an absolute processor cap other than 0.0 and within the valid range of 0.01 and 255.0. Otherwise, a 400 (Bad Request) status code is returned.

If the *{cpc-id}* does not designate a CPC that is in DPM mode, an HTTP status code 409 (Conflict) is returned.

The URI path must designate an existing CPC, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

If the request body contents fail to validate, a 400 (Bad Request) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 333](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 145. Create Capacity Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permission to the Manage Processor Sharing task.
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not exist or the user does not have object-access permission to the CPC.

Table 145. Create Capacity Group: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	5	The CPC is not in DPM mode.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups HTTP/1.1
x-api-session: jht693kvysnx78mws1i0sacatmvoug2y11jz0e0jmg3fv94
content-type: application/json
content-length: 160
{
  "absolute-general-purpose-proc-cap":5.0,
  "capping-enabled":true,
  "description":"Test Group for CP Procs",
  "name":"Cap group 123",
  "short-name":"CPGRP111"
}
```

Figure 152. Create Capacity Group: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/a4d8826c-
d9fa-11e5-8b15-42f2e9ef1641
cache-control: no-cache
date: Tue, 23 Feb 2016 07:00:57 GMT
content-type: application/json;charset=UTF-8
content-length: 117
{
  "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-
d9fb-11e5-8b3e-42f2e9ef1641"
}
```

Figure 153. Create Capacity Group: Response

Delete Capacity Group

The Delete Capacity Group operation deletes the specified capacity group.

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.

Name	Description
<i>{capacity-group-id}</i>	Element ID of the capacity group.

Description

This operation deletes the capacity group with the specified *{capacity-group-id}*. It is required that the specified capacity group does not contain any partition, at the time of deletion. An Inventory Change notification is emitted asynchronously to this operation.

The URI path must designate an existing Capacity Group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

A 409 (Conflict) status code is returned if the capacity group contains any partitions. The Remove Partition from Capacity Group operation should be used to delete all the partitions from this capacity group before attempting this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 146. Delete Capacity Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Manage Processor Sharing task
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not exist or the user does not have object-access permission to the CPC.
	150	The capacity group identified by <i>{capacity-group-id}</i> does not exist.
409 (Conflict)	110	The capacity group contains one or more partitions, and thus could not be deleted.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/a4d8826c-d9fa-11e5-8b15-42f2e9ef1641 HTTP/1.1
x-api-session: 2saqzhzg358oqoke89osqrgt09k27rqwy4d0mtdp6j2z05ok6q
```

Figure 154. Delete Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:02:43 GMT
<No response body>
```

Figure 155. Delete Capacity Group: Response

Get Capacity Group Properties

The Get Capacity Group Properties operation retrieves the properties of a single capacity group.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Response body contents

On successful completion, an HTTP status code 200 (OK) is returned and a JSON object containing the current values of the properties for the Capacity Group element object as defined in the [“Data model” on page 328](#) is provided as a response body. Field names and data types in the JSON object are the same as the property names and data types that are defined in the data model.

Description

This operation retrieves the current values of the properties of the capacity group.

The URI path must designate an existing Capacity Group, and the API user must have object-access permission to the CPC object specified by the {cpc-id}. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the specified CPC.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and a response body with the current values for the properties of the capacity group is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 147. Get Capacity Group Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID {cpc-id} does not designate an existing CPC object, or the API user does not have object-access permission to it.
	150	The capacity group identified by {capacity-group-id} does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-8b3e-42f2e9ef1641 HTTP/1.1
x-api-session: 487yumkfzjbyns00a5qn7v916s8eq49krcnpfqt4n0kfv4
```

Figure 156. Get Capacity Group Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:14:23 GMT
content-type: application/json;charset=UTF-8
content-length: 504
{
  "absolute-general-purpose-proc-cap":5.0,
  "absolute-ifl-proc-cap":0.0,
  "capping-enabled":true,
  "class":"capacity-group",
  "description":"Test Group for CP Procs",
  "element-id":"1de263e4-d9fb-11e5-8b3e-42f2e9ef1641",
  "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-8b3e-42f2e9ef1641",
  "name":"Cap group 123",
  "parent":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0",
  "partition-uris":[
    "/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
  ],
  "short-name":"CPGRP111"
}
```

Figure 157. Get Capacity Group Properties: Response

Add Partition to Capacity Group

The Add Partition to Capacity Group operation adds a partition to the specified capacity group.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/add-partition
```

URI variables:

Name	Description
<i>{cpc-id}</i>	Object ID of the CPC.
<i>{capacity-group-id}</i>	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object with the following field:

Name	Type	Rqd/Opt	Description
partition-uri	String/ URI	Required	The canonical URI of the Partition object to be added to the capacity group.

Description

This operation adds a partition to an existing capacity group identified by the *{capacity-group-id}*.

A partition can be added to the capacity group, only if the processor type that the partition uses has a nonzero cap value in the specified capacity group. Otherwise, a status code 409 (Conflict) is returned.

A capacity group can only contain partitions with shared processors. If the partition specified in the request is configured for dedicated processors, a status code 409 (Conflict) will be returned.

A partition cannot become a member of more than one capacity group. If the partition specified in the request is currently a member of another capacity group or the capacity group identified by *{capacity-group-id}*, a status code 409 (Conflict) is returned.

If the partition does not belong to the same CPC as the capacity group, a status code 400 (Bad Request) is returned.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

On successful execution, the partition is added to the capacity group and the number of processors that could be used by this partition becomes governed by the absolute proc cap values that are defined for this capacity group.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 148. Add Partition to Capacity Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	160	The partition does not belong to the same CPC as the capacity group.
403 (Forbidden)	1	API user does not have action/task permission to the Manage Processor Sharing task.
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not have object-access permission to the CPC.
	2	The partition designated by the partition-uri in the request body does not exist.
	150	The capacity group identified by <i>{capacity-group-id}</i> does not exist.
409 (Conflict)	120	The partition is currently a member of another capacity group and thus, cannot be added to this capacity group.
	130	The partition is already a member of the capacity group identified by <i>{capacity-group-id}</i> .
	170	The partition is configured with dedicated processors.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-8b3e-42f2e9ef1641/operations/add-partition HTTP/1.1
x-api-session: 3ty54o1c7961o2jt0jyjm5nm8xqxofys7s302syjoguaycbeo5
content-type: application/json
content-length: 73
{
  "partition-uri":"/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
}
```

Figure 158. Add Partition to Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:13:42 GMT
<No response body>
```

Figure 159. Add Partition to Capacity Group: Response

Remove Partition from Capacity Group

The Remove Partition from Capacity Group operation removes a partition from an existing capacity group.

HTTP method and URI

POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/remove-partition

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
partition-uri	String/ URI	Required	The canonical URI of the Partition object to be removed from the capacity group.

Description

This operation removes a partition from a capacity group identified by the {capacity-group-id}. If the partition is not currently a member of the capacity group, a status code 409 (Conflict) is returned.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the {cpc-id}. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

On successful execution, the partition is removed from the capacity group and the amount of processing capacity that could be used by this partition is no longer governed by the absolute proc cap values defined for this capacity group.

If this operation changes the values of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned, and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 149. Remove Partition from Capacity Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permission to the Manage Processor Sharing task.
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not exist or the user does not have object-access permission to the CPC.
	2	The partition designated by the partition-uri in the request body does not exist.
	150	The capacity group identified by <i>{capacity-group-id}</i> does not exist.
409 (Conflict)	140	The partition is not currently a member of the capacity group.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-8b3e-42f2e9ef1641/operations/remove-partition HTTP/1.1
x-api-session: 3581gp14ocfmtodeh93ckw1ita2c3k07mr6xb7ymqgcncjv8k
content-type: application/json
content-length: 73
{
  "partition-uri":"/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
}
```

Figure 160. Remove Partition from Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:16:00 GMT
<No response body>
```

Figure 161. Remove Partition from Capacity Group: Response

Update Capacity Group Properties

The Update Capacity Group Properties operation modifies the writable properties of a capacity group.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
<i>{cpc-id}</i>	Object ID of the CPC.
<i>{capacity-group-id}</i>	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the Capacity Group element specified by the *{capacity-group-id}*.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the [“Data model”](#) on page 328. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The capacity group must be removed of all partitions first, if the cap values are updated such that the capacity group starts capping partitions on a different processor type. Otherwise, a status code 400 (Bad Request) is returned.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 150. Update Capacity Group Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Manage Processor Sharing task.
404 (Not Found)	1	The CPC identified by <i>{cpc-id}</i> does not exist or the user does not have object-access permission to the CPC.
	150	The capacity group identified by <i>{capacity-group-id}</i> does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-8b3e-42f2e9ef1641 HTTP/1.1
x-api-session: 8rws222hts1w985vhnn5vcne4x5z6b3q143z0gx7kqzmnuqlu
content-type: application/json
content-length: 42
{
  "absolute-general-purpose-proc-cap":2.0
}
```

Figure 162. Update Capacity Group Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:18:42 GMT
<No response body>
```

Figure 163. Update Capacity Group Properties: Response

Inventory service data

Information about capacity groups can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Capacity Group objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular capacity group is included only if the API user has access permission to that object as described in the Get Capacity Group Properties operation.

For each Capacity Group object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for “[Get Capacity Group Properties](#)” on page 336. That is, the data provided is the same as would be provided if a `Get Capacity Group Properties` operation were requested targeting this object.

Storage Site object

A Storage Site object represents a single storage site in the FICON storage configuration associated with a DPM-enabled CPC. A storage site describes a location that houses a set of storage switches and storage subsystems. A primary site with a default name of "Primary Site", local to the CPC, exists by default and cannot be deleted. An alternate site, typically at a remote location, can be created. The Storage Site object APIs provide access to the set of storage sites within the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to create and delete alternate storage sites, query storage sites, and update selected properties of storage sites.

Data model

This object includes the properties that are defined in the “[Base managed object properties schema](#)” on page 70, with the class-specific specializations identified in [Table 152 on page 345](#). The Storage Site object does not support the operational status related properties.

Table 151. Storage Site object properties			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Site object is of the form <code>/api/storage-sites/{storage-site-id}</code> where <code>{storage-site-id}</code> is the value of the object-id property of the Storage Site object.
object-id	—	String (36)	The unique identifier for the storage site instance.
class	—	String (12)	The class of the Storage Site object is "storage-site" .
parent	—	String/ URI	The parent of a storage site is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
name	(w)(pc)	String (1-64)	The display name specified for the storage site. The length and character requirements on this property are the same as those of the name property described in the “ Base managed object properties schema ” on page 70. Names must be unique to the other storage sites within the FICON configuration associated with the CPC identified by the URI in the cpc-uris property.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage site. Default value: An empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 152. Storage Site object: class specific properties

Name	Qualifier	Type	Description
cpc-uris	—	Array of String/ URI	The list of CPCs that reside in this storage site. Each element in this array is the canonical URI path of a CPC.
storage-subsystem-uris	(c)(pc)	Array of String/ URI	The list of storage subsystems that reside in this storage site. Each element in this array is the canonical URI path of a Storage Subsystem object. The value of this property will change, and property change notifications will be emitted, when storage subsystem objects are defined or undefined.
storage-switch-uris	(c)(pc)	Array of String/ URI	The list of storage switches that reside in this storage site. Each element in this array is the canonical URI path of a Storage Switch object. The value of this property will change, and property change notifications will be emitted, when storage switch objects are defined or undefined.
type	—	String Enum	The type of this storage site. Values: • "primary" - The storage site is the primary, typically local, location. • "alternate" - The storage site is an alternate, typically remote, location.

List Storage Sites

The List Storage Sites operation lists the storage sites known to the target Console.

HTTP method and URI

GET /api/storage-sites

Query parameters:

Name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching CPC URI in its cpc-uris property.
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-sites	Array of storage-site-info objects	Array of storage-site-info objects, described in the next table. The returned array may be empty.

Each nested storage-site-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Site object.
name	String	The name property of the Storage Site object.
cpc-uris	Array of String/ URI	The cpc-uris property of the Storage Site object.
type	String Enum	The type property of the Storage Site object.

Description

This operation lists the storage sites that are known to the target Console. The object URI, name, type and CPC URI list are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage sites that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid value for the storage site **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage sites that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not done.

If the **cpc_uri** query parameter is specified, the returned list is limited to those storage sites that have a matching CPC URI in its **cpc-uris** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage site is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the API user does not have permission to a storage site, that object is simply omitted from the list but no error status code results.

If no storage sites are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 345](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 153. List Storage Sites: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-sites HTTP/1.1
x-api-session: 1exubjc30uxq0rpf2gyycx230m9voc6rpuqperj8gdya59d422
```

Figure 164. List Storage Sites: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:08:43 GMT
content-type: application/json; charset=UTF-8
content-length: 363
{
  "storage-sites": [
    {
      "cpc-uris": [
        "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
      ],
      "name": "New York",
      "object-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492",
      "type": "primary"
    },
    {
      "cpc-uris": [
        "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
      ],
      "name": "New Jersey",
      "object-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492",
      "type": "alternate"
    }
  ]
}
```

Figure 165. List Storage Sites: Response

Create Storage Site

The Create Storage Site operation creates a new alternate Storage Site object.

HTTP method and URI

```
POST /api/storage-sites
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
cpc-uris	Array of String/ URI	Required	The value to be set as the storage site's cpc-uris property. Currently, this list must contain a single CPC URI, representing the CPC associated with the FICON configuration to which this storage site belongs. This limitation may change in the future.
name	String (1-64)	Required	The value to be set as the storage site's name property.
description	String (0-1024)	Optional	The value to be set as the storage site's description property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the newly created Storage Site object.

Description

This operation creates an alternate storage site with the values specified and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the **cpc-uris** field in the request body contains a URI that does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** field in the request body already contains a storage site with the specified name, a 400 (Bad Request) status code is returned. If the CPC identified by the URI in the **cpc-uris** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled, or if its FICON configuration already contains the maximum number of alternate storage sites, or the **cpc-uris** field does not contain exactly one CPC URI, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage site is created and its properties are set to their corresponding request body content's field's values. The new storage site's **type** property is set to **"alternate"**. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to each CPC whose **object-id** is in the **cpc-uris** list.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 348.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 154. Create Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage site with the name specified in the request body already exists within the FICON configuration associated with the CPC identified by the URI in the cpc-uris field specified in the request body.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	2	A CPC identified by a URI in the cpc-uris field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	5	The CPC identified by the URI in the cpc-uris field is not enabled for DPM.
	13	A CPC identified by a URI in the cpc-uris field does not support the dpm-storage-management feature.
	329	The operation cannot be performed because the CPC identified by the URI in the cpc-uris field is an unmanaged CPC, which is not supported by this operation.
	440	The maximum number of alternate storage sites, defined by the CPC maximum-alternate-storage-sites property, already exist within the FICON configuration associated with the CPC identified by the URI in the cpc-uris field in the request body.
	451	The cpc-uris field does not contain exactly one CPC URI.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-sites HTTP/1.1
x-api-session: 4htyri6wi6g3tr91kv01hqxdn9aiucykq3e0ne21uvepky34yp
content-type: application/json
content-length: 119
{
  "cpc-uris":[
    "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
  ],
  "description":"Alternate site",
  "name":"New Jersey"
}
```

Figure 166. Create Storage Site: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492
cache-control: no-cache
date: Mon, 30 Jul 2018 20:06:13 GMT
content-type: application/json; charset=UTF-8
content-length: 72
{
  "object-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 167. Create Storage Site: Response

Usage notes

Each CPC maintains its own view of the physical storage sites. When a list operation is targeted at a console, there will likely be multiple storage site instances returned that represent the same physical site – one for each CPC to which that physical site is configured. There is no intrinsic storage site property that can be used to correlate storage site instances that represent the same physical site. It is therefore recommended that API clients adopt a naming convention that ensures storage sites that represent the same physical site have the same value of their respective **name** properties.

FICON configurations are currently limited to a single alternate storage site. This may change in the future, in which case a console could be managing different versions of CPC that have different alternate storage site limits. It is recommended that API clients use the **maximum-alternate-storage-sites** property of the CPC object to programmatically determine how many alternate sites can be created.

Delete Storage Site

The Delete Storage Site operation deletes an alternate storage site.

HTTP method and URI

```
DELETE /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the alternate storage site to delete.

Description

This operation deletes an alternate storage site. The storage site must be empty; it cannot contain any storage switches or storage subsystems. An Inventory Change notification for the deleted storage site is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-site-id}* does not identify a storage site object on the HMC. If the value of the target storage site's **type** property value is **"primary"**, a 400 (Bad Request) status code is returned. If the target storage site's **storage-fabric-uris** or **storage-switch-uris** array properties are not empty, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage site is deleted from the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 155. Delete Storage Site: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	440	The storage site object with the object-id { <i>storage-site-id</i> } has a type property value of "primary" .
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage site with the object-id { <i>storage-site-id</i> } does not exist on the HMC.
409 (Conflict)	2	The storage site object with the object-id { <i>storage-site-id</i> } was busy and the request timed out.
	452	The storage site with the object-id { <i>storage-site-id</i> } has a storage-fabric-uris or storage-switch-uris property that is not empty.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: ynuX0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
```

Figure 168. Delete Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 169. Delete Storage Site: Response

Get Storage Site Properties

The Get Storage Site Properties operation retrieves the properties of a single Storage Site object.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the storage site object.

Response body contents

On successful completion, an HTTP status code 200 (OK) is returned and a JSON object containing the current values of the properties for the Storage Site object as defined in the [“Data model”](#) on page 344 is provided as a response body. Field names and data types in the JSON object are the same as the property names and data types that are defined in the data model.

Description

Returns the current values of the properties for the storage site object as defined in the [“Data model”](#) on page 344 .

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-site-id}* does not identify a storage site object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 352.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 156. Get Storage Site Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage site with object-id <i>{storage-site-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492 HTTP/1.1
x-api-session: 46y82ucgax84e1jtyy5w5ugv1l9lvpewu2xp9d865ipz4skwqo
```

Figure 170. Get Storage Site Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:12:31 GMT
content-type: application/json; charset=UTF-8
content-length: 595
{
  "class": "storage-site",
  "cpc-uris": [
    "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
  ],
  "description": "Primary site",
  "name": "New York",
  "object-id": "13ff101c-941f-11e8-a0c0-fa163e27d492",
  "object-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492",
  "parent": "/api/console",
  "storage-subsystem-uris": [
    "/api/storage-subsystems/37af8766-943e-11e8-8ffe-fa163e27d492",
    "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
  ],
  "storage-switch-uris": [
    "/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
    "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
  ],
  "type": "primary"
}
```

Figure 171. Get Storage Site Properties: Response

Update Storage Site Properties

The Update Storage Site Properties operation updates one or more of the writable properties of a storage site.

HTTP method and URI

```
POST /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the Storage Site object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Data model” on [page 344](#). The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage site's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-site-id}* does not identify a Storage Site object on the HMC. If the FICON configuration associated with the CPC identified by the URI in the storage site's

cpc-uris property already contains a storage site with the specified name, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage site's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 157. Update Storage Site Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage site with the name specified in the request body already exists within the FICON configuration associated with the CPC identified by the URI in the cpc-uris property of the storage site with the object-id {storage-site-id}.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage site with the object-id {storage-site-id} does not exist on the HMC.
409 (Conflict)	2	The storage site object with the object-id {storage-site-id} was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.


```

{
  "class": "storage-site",
  "cpc-uris": [
    "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
  ],
  "description": "Primary site",
  "name": "New Jersey",
  "object-id": "0336a208-9434-11e8-9c43-fa163e27d492",
  "object-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492",
  "parent": "/api/console",
  "storage-subsystem-uris": [
    "/api/storage-subsystems/9b15669a-943e-11e8-8ffe-fa163e27d492",
    "/api/storage-subsystems/93791760-943e-11e8-9c43-fa163e27d492"
  ],
  "storage-switch-uris": [
    "/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
    "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492"
  ],
  "type": "alternate"
}

```

Figure 174. Storage Site object: Sample inventory data - Response

Storage Fabric object

A Storage Fabric object represents a single storage fabric in the FICON configuration associated with a DPM-enabled CPC. A storage fabric is a collection of interconnected storage switches. If the storage configuration contains multiple storage sites, a storage fabric can, and typically does, span those sites. The Storage Fabric object APIs provide access to the set of storage fabrics within the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to create and delete storage fabrics, list storage fabrics, query storage fabric properties, and update selected properties of storage fabrics.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, with the class-specific specializations identified in Table 159 on page 357. The storage fabric object does not support the operational status related properties.

Table 158. Storage Fabric object properties

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Fabric object is of the form <code>/api/storage-fabrics/{storage-fabric-id}</code> where <code>{storage-fabric-id}</code> is the value of the object-id property of the Storage Fabric object.
object-id	—	String (36)	The unique identifier for the storage fabric instance.
class	—	String (14)	The class of the Storage Fabric object is "storage-fabric" .
parent	—	String/ URI	The parent of a storage fabric is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.

Table 158. Storage Fabric object properties (continued)

Name	Qualifier	Type	Description of specialization
name	(w)(pc)	String (1-64)	The display name specified for the storage fabric. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage fabrics within the FICON configuration associated with the CPC identified by the URI in the cpc-uris property.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage fabric. Default value: An empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 159. Storage Fabric object: class specific properties

Name	Qualifier	Type	Description
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with the FICON configuration in which this storage fabric object resides.
storage-switch-uris	(c)(pc)	Array of String/ URI	The list of storage switches that comprise this storage fabric. Each element in this array is the canonical URI path of a Storage Switch object. The value of this property will change, and property change notifications will be emitted, when storage switch objects are defined or undefined.
high-integrity	(w)(pc)	Boolean	Indicates whether this fabric had been configured as a high integrity fabric. If this fabric contains switches that exist in different storage sites, this value will be true and cannot be written. Default value: false

List Storage Fabrics

The List Storage Fabrics operation lists the storage fabrics known to the target Console.

HTTP method and URI

GET /api/storage-fabrics

Query parameters:

Name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching cpc-uri property.

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-fabrics	Array of storage-fabric-info objects	Array of storage-fabric-info objects, described in the next table. The returned array may be empty.

Each nested storage-fabric-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Fabric object.
name	String	The name property of the Storage Fabric object.
cpc-uri	String/ URI	The cpc-uri property of the Storage Fabric object.

Description

This operation lists the storage fabrics that are known to the target Console. The object URI, name, and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage fabrics that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage fabrics that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage fabric is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the API user does not have permission to a storage fabric, that object is simply omitted from the list but no error status code results.

If no storage fabrics are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 358](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 160. List Storage Fabrics: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-fabrics HTTP/1.1
x-api-session: 3tmygsxg76gvwqr4bgf2kpm2aj7z8nrjb9ffiw6hhxfmjpm9b
```

Figure 175. List Storage Fabrics: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:22:26 GMT
content-type: application/json; charset=UTF-8
content-length: 325
{
  "storage-fabrics": [
    {
      "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
      "name": "Fabric B",
      "object-uri": "/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492"
    },
    {
      "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
      "name": "Fabric A",
      "object-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
    }
  ]
}
```

Figure 176. List Storage Fabrics: Response

Create Storage Fabric

The Create Storage Fabric operation creates a new Storage Fabric object.

HTTP method and URI

```
POST /api/storage-fabrics
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Required	The value to be set as the storage fabric’s cpc-uri property.
name	String (1-64)	Required	The value to be set as the storage fabric's name property.

Field name	Type	Rqd/Opt	Description
description	String (0-1024)	Optional	The value to be set as the storage fabric's description property.
high-integrity	Boolean	Optional	The value to be set as the storage fabric's high-integrity property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The Object URI of the newly created Storage Fabric object.

Description

This operation creates an alternate storage fabric with the values specified and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the **cpc-uri** field in the request body contains a URI that does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the FICON configuration associated with the CPC identified by the **cpc-uri** field in the request body already contains a storage fabric with the specified name, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled, or if direct connections between storage subsystems and adapter ports exist, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage fabric is created and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 360](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 161. Create Storage Fabric: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage fabric with the name specified in the request body already exists within the FICON configuration associated with the CPC identified by the cpc-uri specified in the request body.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	2	A CPC identified by the cpc-uri field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	5	The CPC identified by the cpc-uri field is not enabled for DPM.
	13	The CPC identified by the cpc-uri field does not support the dpm-storage-management feature.
	329	The operation cannot be performed because the CPC identified by the cpc-uris field is an unmanaged CPC, which is not supported by this operation.
	455	One or more direct physical connections between a storage subsystem and an adapter port exist.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-fabrics HTTP/1.1
x-api-session: 1r6vll1yh6tywdhmm5b7jvgf7zyf1g5r5ofs7tmwcp6a58sx
content-type: application/json
content-length: 81
{
  "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
  "name": "Fabric A"
}
```

Figure 177. Create Storage Fabric: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492
cache-control: no-cache
date: Mon, 30 Jul 2018 20:20:41 GMT
content-type: application/json; charset=UTF-8
content-length: 74
{
  "object-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
}
```

Figure 178. Create Storage Fabric: Response

Usage notes

Each CPC maintains its own view of the physical storage fabric. When a list operation is targeted at a console, there will likely be multiple storage fabric instances returned that represent the same physical fabric – one for each CPC to which that physical fabric is configured. There is no intrinsic storage fabric property that can be used to correlate storage fabric instances that represent the same physical fabric. It is therefore recommended that API clients adopt a naming convention that ensures storage fabrics that represent the same physical fabric have the same value of their respective **name** properties.

Delete Storage Fabric

The Delete Storage Fabric operation deletes a storage fabric.

HTTP method and URI

```
DELETE /api/storage-fabrics/{storage-fabric-id}
```

In this request, the URI variable *{storage-fabric-id}* is the object ID of the alternate storage fabric to delete.

Description

This operation deletes a storage fabric. The storage fabric must be empty; it cannot contain any storage switches. An Inventory Change notification is emitted for the deleted storage fabric asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-fabric-id}* does not identify a storage fabric object on the HMC. If the target storage fabric's **storage-switch-uris** array property is not empty, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage fabric is deleted from the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 162. Delete Storage Fabric: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage fabric with the object-id <i>{storage-fabric-id}</i> does not exist on the HMC.
409 (Conflict)	2	The storage fabric object with the object-id <i>{storage-fabric-id}</i> was busy and the request timed out.
	452	The storage fabric with the object-id <i>{storage-site-id}</i> has a storage-switch-uris property that is not empty.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: ynuX0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
```

Figure 179. Delete Storage Fabric: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 180. Delete Storage Fabric: Response

Get Storage Fabric Properties

The Get Storage Fabric Properties operation retrieves the properties of a single Storage Fabric object.

HTTP method and URI

```
GET /api/storage-fabrics/{storage-fabric-id}
```

In this request, the URI variable *{storage-fabric-id}* is the object ID of the storage fabric object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Fabric object as defined in the [“Data model”](#) on page 356. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Fabric object as defined in the [“Data model”](#) on page 356.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-fabric-id}* does not identify a Storage Fabric object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 364.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 163. Get Storage Fabric Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage fabric with object-id <i>{storage-fabric-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: knksjscdlzcrh51e46x387k3uqk8qkp1tahyacguhq0p7m0d
```

Figure 181. Get Storage Fabric Properties: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:03:05 GMT
content-type: application/json; charset=UTF-8
content-length: 452
{
  "class": "storage-fabric",
  "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
  "description": "Storage fabric A",
  "high-integrity": true,
  "name": "Fabric A",
  "object-id": "08ad557c-9436-11e8-9c43-fa163e27d492",
  "object-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
  "parent": "/api/console",
  "storage-switch-uris": [
    "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
    "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
  ]
}

```

Figure 182. Get Storage Fabric Properties: Response

Update Storage Fabric Properties

The Update Storage Fabric Properties operation updates one or more of the writable properties of a storage fabric.

HTTP method and URI

POST /api/storage-fabrics/{storage-fabric-id}

In this request, the URI variable {storage-fabric-id} is the object ID of the Storage Fabric object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the [“Data model” on page 356](#). The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage fabric's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-fabric-id} does not identify a Storage Fabric object on the HMC. If the FICON configuration associated with the CPC identified by the URI in the storage fabric's **cpc-uri** property already contains a storage fabric with the specified name, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage fabric's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 164. Update Storage Fabric Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage fabric with the name specified in the request body already exists within the FICON configuration associated with the CPC identified by the cpc-uri property of the storage fabric with the object-id {storage-fabric-id}.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage fabric with the object-id {storage-fabric-id} does not exist on the HMC.
409 (Conflict)	2	The storage fabric object with the object-id {storage-fabric-id} was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 3cqqq1phyn2nepyewzvway545dpi8z5tycmowv3gqf5nvbw107
content-type: application/json
content-length: 35
{
  "description": "Storage fabric A"
}
```

Figure 183. Update Storage Fabric Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:25:17 GMT

<No response body>
```

Figure 184. Update Storage Fabric Properties: Response

Inventory service data

Information about the Storage Fabrics managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage fabric objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "storage-fabric" are to be included. Information for a particular storage fabric is included only if the API user has access permission to that object as described in the Get Storage Fabric Properties operation.

For each storage fabric to be included, the inventory response includes an array entry for the Storage Fabric object. This entry is a JSON object with the same contents as is specified in the Response body contents section of "Get Storage Fabric Properties" on page 363. That is, the data provided is the same as would be provided if a Get Storage Fabric Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a storage fabric. This object would appear as one array entry in the response array:

```
{
  "class": "storage-fabric",
  "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
  "description": "Storage fabric A",
  "high-integrity": true,
  "name": "Fabric A",
  "object-id": "08ad557c-9436-11e8-9c43-fa163e27d492",
  "object-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
  "parent": "/api/console",
  "storage-switch-uris": [
    "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
    "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
  ]
}
```

Figure 185. Storage Fabric object: Sample inventory data - Response

Storage Switch object

A Storage Switch object represents a single storage switch in the FICON configuration associated with a DPM-enabled CPC. The Storage Switch object APIs provide access to the set of storage switches in the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to define and undefine storage switches, list storage switches, query storage switch properties, update selected properties of storage switches, and move a storage switch to another storage site or storage fabric.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, with the class-specific specializations identified in Table 166 on page 368. The Storage Switch object does not support the operational status related properties.

Table 165. Storage Switch object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Switch object is of the form <code>/api/storage-switches/{storage-switch-id}</code> where <code>{storage-switch-id}</code> is the value of the object-id property of the Storage Switch object.
object-id	—	String (36)	The unique identifier for the storage switch instance.
class	—	String (14)	The class of the Storage Switch object is "storage-switch" .
parent	—	String/ URI	The parent of a storage site is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
name	(w)(pc)	String (1-64)	The display name specified for the storage switch. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage sites within the FICON configuration of its containing storage site and storage fabric. Default value: Currently of the form "Storage switch {domain-id}", where <code>{domain-id}</code> is the value of the domain-id property. This form is subject to change in the future.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage switch. Default value: An empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 166. Storage Switch object: class specific properties

Name	Qualifier	Type	Description
domain-id	(w)(pc)	String (2)	A two-character lowercase hexadecimal number in the range 01-ef that represents the domain identifier assigned to the storage switch. Domain identifiers must be unique to the other storage switches within the same fabric. The value may never be set to null , but can be null when queried if the switch was imported from an incomplete IOCDs.
port-count	(w)(pc)	Integer (2-256)	The number of ports on the switch. Default value: 256

Table 166. Storage Switch object: class specific properties (continued)

Name	Qualifier	Type	Description
storage-fabric-uri	(pc)	String/ URI	The canonical URI of the storage fabric associated with this storage switch. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage fabric to another via the Move Storage Switch to Storage Fabric operation
storage-site-uri	(pc)	String/ URI	The canonical URI of the storage site associated with this storage switch. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage site to another via the Move Storage Switch to Storage Site operation.

List Storage Switches of a Storage Site

The List Storage Switches of a Storage Site operation lists the storage switches associated with the storage site with the given identifier.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}/storage-switches
```

In this request, the URI variable *{storage-site-id}* is the **object-id** of the Storage Site object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
domain-id	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching domain-id property.
storage-fabric-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching storage-fabric-uri property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-switches	Array of storage-switch-info objects	Array of storage-switch-info objects, described in the next table. The returned array may be empty.

Each nested storage-switch-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Switch object.
name	String	The name property of the Storage Switch object.
domain-id	String	The domain-id property of the Storage Switch object.
storage-fabric-uri	String/ URI	The storage-fabric-uri property of the Storage Switch object.

Description

This operation lists the storage switches that are associated with the identified storage site. The object URI, name, domain ID and associated storage fabric are provided for each.

If the object ID *{storage-site-id}* does not identify a Storage Site object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **domain-id** query parameters are specified, the returned list is limited to those storage switches that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

If the **storage-fabric-uri** query parameter is specified, the returned list is limited to those storage switches that have a matching **storage-fabric-uri** property. If the **storage-fabric-uri** parameter is omitted, this filtering is not done.

A storage switch is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage site is associated with a storage switch but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage switches are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 369](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 167. List Storage Switches of a Storage Site: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage site with the object ID <i>{storage-site-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492/storage-switches
HTTP/1.1
x-api-session: 158gv9ldjh8x0yar13qjx5o7xhemmtku20ejfi92rtk8hb3yln
```

Figure 186. List Storage Switches of a Storage Site: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:39:12 GMT
content-type: application/json;charset=UTF-8
content-length: 424
{
  "storage-switches": [
    {
      "domain-id": "11",
      "name": "Storage switch 11",
      "object-uri": "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
      "storage-fabric-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
    },
    {
      "domain-id": "21",
      "name": "Storage switch 21",
      "object-uri": "/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
      "storage-fabric-uri": "/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492"
    }
  ]
}
```

Figure 187. List Storage Switches of a Storage Site: Response

List Storage Switches of a Storage Fabric

The List Storage Switches of a Storage Fabric operation lists the storage switches associated with the storage fabric with the given identifier.

HTTP method and URI

```
GET /api/storage-fabrics/{storage-fabric-id}/storage-switches
```

In this request, the URI variable `{storage-fabric-id}` is the **object-id** of the Storage Fabric object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
domain-id	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching domain-id property.
storage-site-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching storage-site-uri property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-switches	Array of storage-switch-info objects	Array of storage-switch-info objects, described in the next table. Returned array may be empty.

Each nested storage-switch-info object contains the following fields:

Field name	Type	Description
object-uri	String/URI	Canonical URI path (object-uri) of the Storage Switch object.
name	String	The name property of the Storage Switch object.
domain-id	String	The domain-id property of the Storage Switch object.
storage-site-uri	String / URI	The storage-site-uri property of the Storage Switch object.

Description

This operation lists the storage switches that are associated with the identified storage fabric. The object URI, name, domain ID, and associated site are provided for each.

If the object ID *{storage-fabric-id}* does not identify a Storage Fabric object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **domain-id** query parameters are specified, the returned list is limited to those storage switches that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

If the **storage-site-uri** query parameter is specified, the returned list is limited to those storage sites that have a matching **storage-site-uri** property. If the **storage-site-uri** parameter is omitted, this filtering is not done.

A storage switch is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage fabric is associated with a storage switch but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage switches are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 372](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 168. List Storage Switches of a Storage Fabric: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage fabric with the object ID <i>{storage-fabric-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492/storage-switches
HTTP/1.1
x-api-session: 5iggido8gsjbbun0k77blh9bsfvdxn8swdioyonor0ohpuebfu
```

Figure 188. List Storage Switches of a Storage Fabric: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:08:30 GMT
content-type: application/json;charset=UTF-8
content-length: 416
{
  "storage-switches": [
    {
      "domain-id": "11",
      "name": "Storage switch 11",
      "object-uri": "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
      "storage-site-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
    },
    {
      "domain-id": "10",
      "name": "Storage switch 10",
      "object-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
      "storage-site-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
    }
  ]
}
```

Figure 189. List Storage Switches of a Storage Fabric: Response

Define Storage Switch

The Define Storage Switch operation defines a new Storage Switch object.

HTTP method and URI

```
POST /api/console/operations/define-storage-switch
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Optional	The value to be set as the storage switch's name property.
description	String (0-1024)	Optional	The value to be set as the storage switch's description property.
domain-id	String (2)	Required	The value to be set as the storage switch's domain-id property.
port-count	Integer (2-256)	Optional	The value to be set as the storage switch's port-count property.
storage-fabric-uri	String/ URI	Required	The value to be set as the storage switch's storage-fabric-uri property.
storage-site-uri	String/ URI	Required	The value to be set as the storage switch's storage-site-uri property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the newly defined Storage Switch object.

Description

This operation defines a storage switch with the values specified and then returns its **object-uri** in the response body. An Inventory Change notification and Property Change notifications for the associated storage site and storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the URI specified in the **storage-site-uri** field does not identify a storage site object on the HMC, or the URI specified in the **storage-fabric-uri** field does not identify a storage fabric on the HMC, a 404 (Not Found) status code is returned. If the storage fabric identified by *{storage-fabric-id}* already contains a storage switch with the specified **domain-id**, or if the **storage-site-uri** field references a storage site that does not reside in the same CPC as the storage fabric referenced by the **storage-fabric-uri** field, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage switch with the specified name exists within the same FICON configuration in which the containing storage size and storage fabric exist.

If the request body contents are valid, the storage switch is defined and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 374.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 169. Define Storage Switch: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage switch with the name specified in the request body already exists within the same FICON configuration in which the containing storage site and storage fabric exist.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	2	A storage site with the URI specified in the storage-site-uri field does not exist on the HMC.
	440	A storage fabric with the URI specified in the storage-fabric-uri field does not exist on the HMC.
409 (Conflict)	441	The storage site referenced by the storage-site-uri field and the storage fabric referenced by the storage-fabric-uri field do not reside in the same CPC.
	445	A storage switch with the domain-id specified in the request body already exists within the storage fabric identified by the storage-fabric-uri field in the request body.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/operations/define-storage-switch HTTP/1.1
x-api-session: 27u5v0reohtwg2ncuw2xebonfrtxpkbdst3cmc5w12jemo6iet
content-type: application/json
content-length: 181
{
  "domain-id": "10",
  "storage-fabric-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
  "storage-site-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 190. Define Storage Switch: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:31:45 GMT
content-type: application/json; charset=UTF-8
content-length: 75
{
  "object-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
}
```

Figure 191. Define Storage Switch: Response

Usage notes

Each CPC maintains its own view of the physical storage switches. When a list operation is targeted at a console, there will likely be multiple storage switch instances returned that represent the same physical switch – one for each CPC to which that physical switch is configured. There is no intrinsic storage switch property that can be used to correlate storage switch instances that represent the same physical switch. It is therefore recommended that API clients adopt a naming convention that ensures storage switches that represent the same physical switch have the same value of their respective **name** and **domain-id** properties.

Undefine Storage Switch

The Undefine Storage Switch operation removes a storage switch definition.

HTTP method and URI

```
POST /api/storage-switches/{storage-switch-id}/operations/undefine
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch that is to be undefined.

Description

This operation removes a storage switch definition. If the storage switch contains switch ports, they will be removed as well. An Inventory Change notification and Property Change notifications for the associated storage site's and storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a storage switch object on the HMC.

If any physical connections from adapters or storage subsystems to the storage switch exist, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage switch definition is removed.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 170. Undefine Storage Switch: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage switch with the object ID <i>{storage-switch-id}</i> does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the object ID <i>{storage-switch-id}</i> was busy and the request timed out.
	446	An endpoint connection in an adapter or storage subsystem references the Storage Switch object with the object ID <i>{storage-switch-id}</i> .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492/operations/undefine
HTTP/1.1
x-api-session: ynuX0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
content-type: application/json
```

Figure 192. Undefine Storage Switch: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 193. Undefine Storage Switch: Response

Get Storage Switch Properties

The **Get Storage Switch Properties** operation retrieves the properties of a single Storage Switch object.

HTTP method and URI

```
GET /api/storage-switches/{storage-switch-id}
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Switch object as defined in the “Data model” on page 368. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Switch object as defined in the “Data model” on page 368.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-switch-id}* does not identify a storage switch object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the “Response body contents” on page 378.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 171. Get Storage Switch Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage switch with object-id <i>{storage-switch-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 5067e2d7z01ujz4o5j106jy255f01248enx0p5wlnr8ti98xld
```

Figure 194. Get Storage Switch Properties: Request


```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:55:23 GMT
content-type: application/json; charset=UTF-8
content-length: 411
{
  "class": "storage-switch",
  "description": "",
  "domain-id": "10",
  "name": "Storage switch 10",
  "object-id": "90204662-9437-11e8-9c43-fa163e27d492",
  "object-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
  "parent": "/api/console",
  "port-count": 256,
  "storage-fabric-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
  "storage-site-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}

```

Figure 195. Get Storage Switch Properties: Response

Update Storage Switch Properties

The Update Storage Switch Properties operation updates one or more of the writable properties of a storage switch.

HTTP method and URI

POST /api/storage-switches/{*storage-switch-id*}

In this request, the URI variable {*storage-switch-id*} is the object ID of the Storage Switch object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Data model” on page 368. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage switch's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {*storage-switch-id*} does not identify a Storage Switch object on the HMC. If the containing storage fabric already contains a storage switch with the specified **domain-id** or the **port-count** field value does not allow for the currently configured ports, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage switch with the specified name exists within the same FICON configuration in which the containing storage site and storage fabric exist.

If the request body contents are valid, the storage switch's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 172. Update Storage Switch Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage switch with the name specified in the request body already exists within the FICON configuration in which the containing storage site and storage fabric exist.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage switch with the object-id {storage-switch-id} does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the object-id {storage-switch-id} was busy and the request timed out.
	441	The value of the port-count field does not allow for the currently configured ports on the Storage Switch object with the object-id {storage-switch-id}.
	445	A storage switch with the domain-id specified in the request body already exists within its associated storage fabric.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: z73khdzsrzllklgo1gbn095ot7bgckt4g6ki1mwggezian3
content-type: application/json
content-length: 50
{
  "description": "Switch 10 in New York, Fabric A"
}
```

Figure 196. Update Storage Switch Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:56:44 GMT

<No response body>
```

Figure 197. Update Storage Switch Properties: Response

Move Storage Switch to Storage Site

The Move Storage Switch to Storage Site operation moves a storage switch from its current storage site to a different storage site.

HTTP method and URI

POST /api/storage-switches/{storage-switch-id}/operations/move-storage-site

In this request, the URI variable {storage-switch-id} is the object ID of the storage switch that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
storage-site-uri	String/ URI	Required	The canonical URI path for the storage site to which this switch is to be moved.

Description

This operation moves a storage switch from the storage site to which it is currently associated to a different storage site within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-switch-id} does not identify a Storage Switch object on the HMC, or the URI specified in the **storage-site-uri** field does not identify a Storage Site object on the HMC. If the storage site specified in the **storage-site-uri** field already contains the Storage Switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage site is associated with a different FICON configuration.

If the request body contents are valid, the **storage-site-uri** property of the storage switch identified by {storage-switch-id} is updated to the value specified in the **storage-site-uri** field. The storage switch is removed from the **storage-switch-uris** array property of the original storage site and added to the **storage-switch-uris** property of the new storage site.

Property-change notifications on the switch's **storage-site-uri** property and on both the original and new storage site's **storage-switch-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 173. Move Storage Switch to Storage Site: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	441	The storage site identified by the storage-site-uri field is associated with a different FICON configuration than the current storage site.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage switch with the object-id {storage-switch-id} does not exist on the HMC.
	2	The storage site identified by the storage-site-uri field does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the object-id {storage-switch-id} was busy and the request timed out.
	450	The storage site identified by the storage-site-uri field already contains a storage switch with the object-id {storage-switch-id}.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492/operations/move-  
storage-site HTTP/1.1  
x-api-session: 45bnirh6wcq2qs1a9fqyxnozbufxfv0yzjnnlq0ojtfxgs35mm  
content-type: application/json  
content-length: 79  
{  
  "storage-site-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"  
}
```

Figure 198. Move Storage Switch to Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:43:09 GMT
<No response body>
```

Figure 199. Move Storage Switch to Storage Site: Response

Move Storage Switch to Storage Fabric

The Move Storage Switch to Storage Fabric operation moves a storage switch from its current storage fabric to a different storage fabric.

HTTP method and URI

POST /api/storage-switches/{storage-switch-id}/operations/move-storage-fabric

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
storage-fabric-uri	String/ URI	Required	The canonical URI path for the storage fabric to which this switch is to be moved.

Description

This operation moves a storage switch from the storage fabric to which it is currently associated to a different storage fabric within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a Storage Switch object on the HMC, or the URI specified in the **storage-fabric-uri** field does not identify a Storage Fabric object on the HMC. If the storage fabric specified in the **storage-fabric-uri** field already contains the Storage Switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage fabric is associated with a different FICON configuration.

If the request body contents are valid, the **storage-fabric-uri** property of the storage switch identified by *{storage-switch-id}* is updated to the value specified in the **storage-fabric-uri** field. The storage switch is removed from the **storage-switch-uris** array property of the original storage fabric and added to the **storage-switch-uris** property of the new storage fabric.

Property-change notifications on the switch's **storage-fabric-uri** property and on both the original and new storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 174. Move Storage Switch to Storage Fabric: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	441	The storage fabric identified by the storage-fabric-uri field is associated with a different FICON configuration than the current storage fabric.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage switch with the object-id { <i>storage-switch-id</i> } does not exist on the HMC.
	2	The storage fabric identified by the storage-fabric-uri field does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the object-id { <i>storage-switch-id</i> } was busy and the request timed out.
	450	The storage fabric identified by the storage-fabric-uri field already contains a storage switch with the object-id { <i>storage-switch-id</i> }.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492/operations/move-storage-fabric
HTTP/1.1
x-api-session: 3u41vmbq45215fjupxm3jjuzua6i1sbqhb7fguq1l3zbha7frp
content-type: application/json
content-length: 83
{
  "storage-fabric-uri": "/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
}
```

Figure 200. Move Storage Switch to Storage Fabric: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:45:56 GMT

<No response body>
```

Figure 201. Move Storage Switch to Storage Fabric: Response

Inventory service data

Information about the Storage Switches managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage switch objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "storage-switch" are to be included. Information for a particular storage switch is included only if the API user has access permission to that object as described in the Get Storage Switch Properties operation.

For each storage switch to be included, the inventory response includes an array entry for the Storage Switch object. This entry is a JSON object with the same contents as is specified in the "Response body contents" section for ["Get Storage Switch Properties" on page 377](#). That is, the data provided is the same as would be provided if a Get Storage Switch Properties operation were requested targeting this object.:

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a storage switch. This object would appear as one array entry in the response array:

```
{
  "class": "storage-switch",
  "description": "",
  "domain-id": "21",
  "name": "Storage switch 21",
  "object-id": "0e261690-9438-11e8-8ffe-fa163e27d492",
  "object-uri": "/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
  "parent": "/api/console",
  "port-count": 256,
  "storage-fabric-uri": "/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492",
  "storage-site-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 202. Storage Switch object: Sample inventory data - Response

Storage Subsystem object

A Storage Subsystem object represents a single storage subsystem in the FICON configuration associated with a DPM-enabled CPC. The Storage Subsystem object APIs provide access to the set of storage subsystems in the FICON configuration. APIs exist to define and undefine storage subsystems, list storage subsystems, query storage subsystem properties, update selected properties of storage subsystems, move a storage subsystem to a different site, and to manage a storage subsystem's endpoint connections.

Data model

This object includes the properties that are defined in the ["Base managed object properties schema" on page 70](#), with the class-specific specializations identified in [Table 176 on page 386](#). The Storage Subsystem object does not support the operational status related properties.

Table 175. Storage Subsystem object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Subsystem object is of the form /api/storage-subsystems/{ <i>storage-subsystem-id</i> } where { <i>storage-subsystem-id</i> } is the value of the object-id property of the Storage Subsystem object.
object-id	—	String (36)	The unique identifier for the storage subsystem instance.
class	—	String (17)	The class of the Storage Subsystem object is " storage-subsystem ".
parent	—	String/ URI	The parent of a storage subsystem is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
name	(w)(pc)	String (1-64)	The display name specified for the storage subsystem. The length and character requirements on this property are the same as those of the name property described in the " Base managed object properties schema " on page 70. Names must be unique to the other storage subsystems within the FICON configuration of its containing storage site.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage subsystem. Default value: An empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 176. Storage Subsystem object: class specific properties

Name	Qualifier	Type	Description
connection-endpoints	(c)(pc)	Array of subsystem-connection-endpoint objects	Array of subsystem-connection-endpoint objects, described in the next table. A subsystem connection endpoint describes a switch or adapter to which this storage subsystem is physically connected. There can be at most 64 connection endpoints configured to a single storage subsystem. The value of this property will change, and property change notifications will be emitted, when endpoints are added and removed via the Add Connection Endpoint and Remove Connection Endpoint operations.

Table 176. Storage Subsystem object: class specific properties (continued)

Name	Qualifier	Type	Description
storage-control-unit-uris	(c)(pc)	Array of String/ URI	The list of storage control units defined within the storage subsystem. Each element in this array is the canonical URI path of a Storage Control Unit object. The value of this property will change, and property change notifications will be emitted, when storage control units are defined or undefined via the Define Storage Control Unit and Undefine Storage Control Unit operations
storage-site-uri	(pc)	String/ URI	The canonical URI of the storage site associated with this storage subsystem. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage site to another via the Move Storage Subsystem to Storage Site operation.

A connection endpoint defines a storage switch or adapter to which a storage subsystem is physically connected.

Table 177. subsystem-connection-endpoint nested object properties

Name	Qualifier	Type	Description
endpoint-class	—	String Enum	The class of the object identified by endpoint-uri. Values: • "adapter" – The storage subsystem is directly connected to an adapter. • "storage-switch" – The storage subsystem is connected to a storage switch.
endpoint-uri	—	String/ URI	The canonical URI path for the Storage Switch or Adapter object to which this subsystem is connected.
port-id	—	String (2)	A two-character lowercase hexadecimal number that represents the switch port. This value will be null if endpoint-uri references an Adapter object.

List Storage Subsystems of a Storage Site

The List Storage Subsystems of a Storage Site operation lists the storage subsystems associated with the storage site with the given identifier.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}/storage-subsystems
```

In this request, the URI variable *{storage-site-id}* is the **object-id** of the Storage Site object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-subsystems	Array of storage-subsystem-info objects	Array of storage-subsystem-info objects, described in the next table. The returned array may be empty.

Each nested storage-subsystem-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Subsystem object.
name	String	The name property of the Storage Subsystem object.

Description

This operation lists the storage subsystems that are associated with the identified storage site. The object URI and name are provided for each.

If the object ID *{storage-site-id}* does not identify a storage site object on the HMC, a 404 (Not Found) status code is returned.

If the **name** query parameter is specified, the returned list is limited to those storage subsystems that have a **name** property that matches the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

A storage subsystem is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage site is associated with a storage subsystem but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage subsystems are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 388](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 178. List Storage Subsystems of a Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage site with the object ID { <i>storage-site-id</i> } does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492/storage-subsystems
HTTP/1.1
x-api-session: 2fdcirhybjv0ogapni5r3z0oo86u7qj87id7rlcons4ce2jfiw
```

Figure 203. List Storage Subsystems of a Storage Site: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:29:44 GMT
content-type: application/json;charset=UTF-8
content-length: 216
{
  "storage-subsystems":[
    {
      "name":"DS8886 A",
      "object-uri":"/api/storage-subsystems/37af8766-943e-11e8-8ffe-fa163e27d492"
    },
    {
      "name":"DS8870 A",
      "object-uri":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
    }
  ]
}
```

Figure 204. List Storage Subsystems of a Storage Site: Response

Define Storage Subsystem

The Define Storage Subsystem operation defines a new Storage Subsystem object.

HTTP method and URI

```
POST /api/console/operations/define-storage-subsystem
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the storage subsystem's name property.
description	String (0-1024)	Optional	The value to be set as the storage subsystem's description property.
storage-site-uri	String/ URI	Required	The value to be set as the storage subsystem's storage-site-uri property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the newly defined Storage Subsystem object.

Description

This operation defines a storage subsystem with the specified values and then returns its **object-uri** in the response body. An Inventory Change notification and a property-change notification on the associated storage site's **storage-subsystem-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the URI specified in the **storage-site-uri** field does not identify a storage site object on the HMC, a 404 (Not Found) status code is returned. If another storage subsystem with the specified **name** exists within the same FICON configuration in which the containing storage site exists, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage subsystem is defined and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 390](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 179. Define Storage Subsystem: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage subsystem with the name specified in the request body already exists within the same FICON configuration in which the containing storage site exists.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	2	A storage site with the URI specified in the storage-site-uri field does not exist on the HMC.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/operations/define-storage-subsystem HTTP/1.1
x-api-session: 4mmbozh10i5mvrlosv8r1r5e5hrfd7gc82pkpzqbm0cv6bepx
content-type: application/json
content-length: 99
{
  "name": "DS8870 A",
  "storage-site-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 205. Define Storage Subsystem: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:21:02 GMT
content-type: application/json; charset=UTF-8
content-length: 77
{
  "object-uri": "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
}
```

Figure 206. Define Storage Subsystem: Response

Usage notes

Each CPC maintains its own view of the physical storage subsystems. When a list operation is targeted at a console, there will likely be multiple storage subsystem instances returned that represent the same physical subsystem – one for each CPC to which that physical subsystem is configured. There is no intrinsic storage subsystem property that can be used to correlate storage subsystem instances that represent the same physical subsystems. It is therefore recommended that API clients adopt a naming convention that ensures storage subsystems that represent the same physical subsystem have the same value of their respective **name** properties.

Undefine Storage Subsystem

The Undefine Storage Subsystem operation removes a storage subsystem definition.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/undefine
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem that is to be undefined.

Description

This operation removes a storage subsystem definition. If the storage subsystem contains storage control units or connection endpoints, they will be removed as well. Switch or adapter objects referenced by a deleted endpoint are not deleted. Inventory Change notifications for the removed storage subsystem and for each of its contained storage control units are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC.

If physical connections from adapters or storage switches to the storage subsystem exist, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage subsystem definition, and all its storage control units and connection endpoints, are removed.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 180. Undefine Storage Subsystem: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage subsystem with the object ID <i>{storage-subsystem-id}</i> does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the object ID <i>{storage-subsystem-id}</i> was busy and the request timed out.
	446	The connection-endpoints property of the Storage Subsystem object with the object ID <i>{storage-subsystem-id}</i> is not empty.

Table 180. Undefine Storage Subsystem: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  undefine HTTP/1.1
x-api-session: 67zvvhvf3c5z2uu0euonieit063n2spw26aco76zkvt422iir6a
content-type: application/json
```

Figure 207. Undefine Storage Subsystem: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:20:28 GMT

<No response body>
```

Figure 208. Undefine Storage Subsystem: Response

Get Storage Subsystem Properties

The Get Storage Subsystem Properties operation retrieves the properties of a single Storage Subsystem object.

HTTP method and URI

```
GET /api/storage-subsystems/{storage-subsystem-id}
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the Storage Subsystem object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Subsystem object as defined in the [“Data model”](#) on page 385. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Subsystem object as defined in the [“Data model”](#) on page 385.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 393.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 181. Get Storage Subsystem Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage subsystem with object-id {storage-subsystem-id} does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 3fbndln8qdk7d0vsh10ltfg8fxpx8zqp0emq2xbl8gnuwhrvk
```

Figure 209. Get Storage Subsystem Properties: Request


```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:09:00 GMT
content-type: application/json; charset=UTF-8
content-length: 620
{
  "class": "storage-subsystem",
  "connection-endpoints": [
    {
      "endpoint-class": "storage-switch",
      "endpoint-uri": "/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
      "port-id": "00"
    },
    {
      "endpoint-class": "storage-switch",
      "endpoint-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
      "port-id": "00"
    }
  ],
  "description": "DS8870 in New York",
  "name": "DS8870 A",
  "object-id": "76c30590-943e-11e8-9c43-fa163e27d492",
  "object-uri": "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
  "parent": "/api/console",
  "storage-control-unit-uris": [],
  "storage-site-uri": "/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}

```

Figure 210. Get Storage Subsystem Properties: Response

Update Storage Subsystem Properties

The Update Storage Subsystem Properties operation updates one or more of the writable properties of a storage subsystem.

HTTP method and URI

POST /api/storage-subsystems/{storage-subsystem-id}

In this request, the URI variable {storage-subsystem-id} is the object ID of the Storage Subsystem object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Data model” on page 385. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage subsystem's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-subsystem-id} does not identify a Storage Subsystem object on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage subsystem with the specified **name** exists within the same FICON configuration on which the containing storage site exists.

If the request body contents are valid, the storage subsystem's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 182. Update Storage Subsystem Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage subsystem with the name specified in the request body already exists within the FICON configuration in which the containing storage site exists.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage subsystem with the object-id {storage-subsystem-id} does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the object-id {storage-subsystem-id} was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 1zwn9re00ctj6qu09fxgu6y28lb8la7dm752gbe60m7uyg5z1m
content-type: application/json
content-length: 37
{
  "description": "DS8870 in New York"
}
```

Figure 211. Update Storage Subsystem Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:24:35 GMT

<No response body>
```

Figure 212. Update Storage Subsystem Properties: Response

Move Storage Subsystem to Storage Site

The Move Storage Subsystem to Storage Site operation moves a storage subsystem from its current storage site to a different storage site.

HTTP method and URI

POST /api/storage-subsystems/{storage-subsystem-id}/operations/move-storage-site

In this request, the URI variable {storage-subsystem-id} is the object ID of the storage subsystem that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
storage-site-uri	String/ URI	Required	The canonical URI path for the storage site to which this subsystem is to be moved.

Description

This operation moves a storage subsystem from the storage site to which it is currently associated to a different storage site within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-subsystem-id} does not identify a Storage Subsystem object on the HMC, or the URI specified in the **storage-site-uri** field does not identify a Storage Site object on the HMC. If the storage site specified in the **storage-site-uri** field already contains the Storage Subsystem, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage site is associated with a different FICON configuration.

If the request body contents are valid, the **storage-site-uri** property of the storage subsystem identified by {storage-subsystem-id} is updated to the value specified in the **storage-site-uri** field. The storage subsystem is removed from the **storage-subsystem-uris** array property of the original storage site and added to the **storage-subsystem-uris** property of the new storage site.

Property change notifications on the subsystem's **storage-site-uri** property and on both the original and new storage site's **storage-subsystem-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 183. Move Storage Subsystem to Storage Site: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	441	The storage site identified by the storage-site-uri field is associated with a different FICON configuration than the current storage site.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage subsystem with the object-id { <i>storage-subsystem-id</i> } does not exist on the HMC.
	2	The storage site identified by the storage-site-uri field does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the object-id { <i>storage-subsystem-id</i> } was busy and the request timed out.
	450	The storage site identified by the storage-site-uri field already contains a storage subsystem with the object-id { <i>storage-subsystem-id</i> }.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/  
  move-storage-site HTTP/1.1  
x-api-session: 3uyn7nr37w44fg2j59hytrtc4ogmw46e5khw1mcbi3undieovv  
content-type: application/json  
content-length: 79  
{  
  "storage-site-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"  
}
```

Figure 213. Move Storage Subsystem to Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:27:40 GMT

<No response body>
```

Figure 214. Move Storage Subsystem to Storage Site: Response

Add Connection Endpoint

The Add Connection Endpoint operation adds a connection endpoint to a storage subsystem.

HTTP method and URI

POST /api/storage-subsystems/{storage-subsystem-id}/operations/add-connection-endpoint

In this request, the URI variable {storage-subsystem-id} is the object ID of the storage subsystem to which a connection endpoint is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
endpoint-uri	String/ URI	Required	The canonical URI path for the Storage Switch or Adapter object to which this subsystem is connected.
port-id	String (2)	Optional	A two-character lowercase hexadecimal number that represents the switch port. This value is required if endpoint-uri references a Storage Switch object and is prohibited if endpoint-uri references an Adapter object.

Description

This operation adds a connection endpoint definition to a storage subsystem. A Property Change notification for the storage subsystem's **connection-endpoints** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-subsystem-id} does not identify a storage subsystem object on the HMC, or if the endpoint URI does not identify an adapter or storage switch on the HMC.

If the specified connection endpoint already exists on the storage subsystem, or if the port specified by **port-id** does not exist on the target storage switch, or if the port specified by **port-id** is already configured as a connection endpoint of another storage subsystem, or if there are already 64 endpoints defined for the target storage subsystem, or **endpoint-uri** identifies a switch in a different storage site, or **endpoint-uri** identifies an adapter port when one or more fabrics exists, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **port-id** field is specified for an adapter endpoint.

If the request body contents are valid, the new connection endpoint is added to the **connection-endpoints** property of the storage subsystem.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 184. Add Connection Endpoint: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	15	The port-id field is present in the request body when the endpoint-uri field references an Adapter object
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage subsystem with the object-id {storage-subsystem-id} does not exist on the HMC.
	2	The adapter or storage switch referenced by the endpoint-uri field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage subsystem object with the object-id {storage-subsystem-id} was busy and the request timed out.
	442	The port identified by port-id is not defined on the endpoint switch.
	443	The specified connection endpoint already exists on the storage subsystem object with the object-id {storage-subsystem-id}.
	455	A new endpoint cannot be created to an adapter port when storage fabrics are defined.
	456	A new endpoint cannot be created to a storage switch that exists in a different storage site.
	457	The switch port identified by the port-id field is already configured as a connection endpoint of another storage subsystem.
	486	A new endpoint cannot be created for the storage subsystem with the object-id {storage-subsystem-id} because the maximum number of endpoints (64) for that storage subsystem are already defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  add-connection-endpoint HTTP/1.1
x-api-session: 4ks9l9ak7cj3eah3fiqwm3arl003hdrqi7uy68bdso7g3z2iopj
content-type: application/json
content-length: 95
{
  "endpoint-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
  "port-id": "00"
}
```

Figure 215. Add Connection Endpoint: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:44:11 GMT

<No response body>
```

Figure 216. Add Connection Endpoint: Response

Remove Connection Endpoint

The Remove Connection Endpoint operation removes a connection endpoint from a storage subsystem.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/remove-connection-endpoint
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem from which the connection endpoint is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
endpoint-uri	String/ URI	Required	The canonical URI path for the Storage Switch or Adapter object from which this subsystem is to be disconnected.
port-id	String (2)	Optional	A two-character lowercase hexadecimal number that represents the switch port. This value is required if endpoint-uri references a Storage Switch object and is ignored if endpoint-uri references an Adapter object.

Description

This operation removes a connection endpoint definition from a storage subsystem. A Property Change notification for the storage subsystem's **connection-endpoints** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not

Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, or if **endpoint-uri** does not identify an adapter or storage switch on the HMC.

If the specified connection endpoint does not exist on the storage subsystem, or the connection endpoint is used by a storage path, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the specified connection endpoint is removed from the **connection-endpoints** property of the storage subsystem.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 185. Remove Connection Endpoint: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage subsystem with the object-id <i>{storage-subsystem-id}</i> does not exist on the HMC.
	2	The adapter or storage switch referenced by the endpoint-uri field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage subsystem object with the object-id <i>{storage-subsystem-id}</i> was busy and the request timed out.
	444	The specified connection endpoint does not exist on the storage subsystem object with the object-id <i>{storage-subsystem-id}</i> .
	446	The specified connection endpoint is configured in one or more of the storage paths defined in the storage subsystem object with the object-id <i>{storage-subsystem-id}</i> .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/  
  remove-connection-endpoint HTTP/1.1  
x-api-session: 1e1n4xd5sqegdqh5t1cvwp7quls9q55uuu7ahbc8ue3ywfva0g  
content-type: application/json  
content-length: 95  
{  
  "endpoint-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",  
  "port-id":"00"  
}
```

Figure 217. Remove Connection Endpoint: Request

```
204 No Content  
server: Hardware management console API web server / 2.0  
cache-control: no-cache  
date: Mon, 30 Jul 2018 21:43:47 GMT  
  
<No response body>
```

Figure 218. Remove Connection Endpoint: Response

Inventory service data

Information about the Storage Subsystems managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage subsystem objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "storage-subsystem" are to be included. Information for a particular storage subsystem is included only if the API user has access permission to that object as described in the [Get Storage Subsystem Properties](#) operation.

For each storage subsystem to be included, the inventory response array includes the following:

- An array entry for the storage subsystem object. This entry is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Subsystem Properties”](#) on [page 393](#). That is, the data provided is the same as would be provided if a Get Storage Subsystem Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a storage subsystem. This object would appear as one array entry in the response array:

```

{
  "class": "storage-subsystem",
  "connection-endpoints": [
    {
      "endpoint-class": "storage-switch",
      "endpoint-uri": "/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
      "port-id": "00"
    },
    {
      "endpoint-class": "storage-switch",
      "endpoint-uri": "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
      "port-id": "00"
    }
  ],
  "description": "",
  "name": "DS8886 B",
  "object-id": "9b15669a-943e-11e8-8ffe-fa163e27d492",
  "object-uri": "/api/storage-subsystems/9b15669a-943e-11e8-8ffe-fa163e27d492",
  "parent": "/api/console",
  "storage-control-unit-uris": [],
  "storage-site-uri": "/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}

```

Figure 219. Storage Subsystem object: Sample inventory data - Response

Storage Control Unit object

A Storage Control Unit object represents a single storage control unit in the FICON configuration associated with a DPM-enabled CPC. The Storage Control Unit object APIs provide access to the set of storage control units within the FICON configuration. APIs exist to define and undefine storage control units, list storage control units, query storage control unit properties, update selected properties of storage control units, and managed volume ranges. APIs also exist to create, delete, query and update the storage path elements of a storage control unit.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, with the class-specific specializations identified in Table 187 on page 405. The Storage Subsystem object does not support the operational status related properties.

Table 186. Storage Control Unit object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Control Unit object is of the form /api/storage-control-units/{ <i>storage-control-unit-id</i> } where { <i>storage-control-unit-id</i> } is the value of the object-id property of the Storage Control Unit object.
object-id	—	String (36)	The unique identifier for the storage control unit instance.
class	—	String (20)	The class of the Storage Control Unit object is " storage-control-unit ".
parent	—	String/ URI	The parent of a storage control unit is conceptually its owning storage subsystem, and so the parent value is the canonical URI path for the subsystem.

Table 186. Storage Control Unit object: base managed object properties specializations (continued)			
Name	Qualifier	Type	Description of specialization
name	(w)(pc)	String (1-64)	The display name specified for the storage control unit. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage control units within the parent storage subsystem. Default value: Currently of the form Control unit {<i>logical-address</i>}, where {<i>logical-address</i>} is the value of the logical-address property. This form is subject to change in the future.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage control unit. Default value: An empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 187. Storage Control Unit object: class specific properties			
Name	Qualifier	Type	Description
logical-address	(w)(pc)	String (2)	A two-character lowercase hexadecimal number that represents the logical address that uniquely identifies the storage control unit within its parent storage subsystem. Logical addresses must be unique to the other storage control units within the parent storage subsystem.
storage-path-uris	(c)(pc)	Array of String/URI	The list of paths that connect this storage control unit optionally to a port on a storage switch, and ultimately to an adapter. Each element in this array is the canonical URI path of a Storage Path element object, described Table 189 on page 406. There can be a maximum of 8 storage paths that Configure a single control unit, so the maximum number of entries in this array is 8. The value of this property will change, and property change notifications will be emitted, when storage paths are added and removed via the Create Storage Path and Delete Storage Path operations.
volume-ranges	(c)(pc)	Array of volume-range objects	Array of volume-range objects, described in Table 188 on page 406. A volume range describes a contiguous set of storage volume unit addresses that are managed by the storage control unit. The value of this property will change, and property change notifications will be emitted, when ports are added and removed via the Add Volume Range and Remove Volume Range operations.

A volume range defines a contiguous set of base or alias volumes within a storage control unit.

Table 188. Storage Control Unit object: volume-range nested object properties

Name	Qualifier	Type	Description
starting-volume	—	String (2)	A two-character lowercase hexadecimal number that represents the first unit address in the volume range.
ending-volume	—	String (2)	A two-character lowercase hexadecimal number that represents the last unit address in the volume range. Default value: The ending volume will be the same as the starting volume, thus giving a range of one volume.
type	—	String Enum	The volume type. Values: • "base" – The volumes in the range are base volumes. • "alias" – The volumes in the range are alias volumes. Default value: "base"

Storage Path element object

A storage path defines a communications path from a storage control unit to an adapter, optionally through one or two storage switches.

Table 189. Storage Control Unit object: storage path element object properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for the storage path element object, of the form /api/storage-control-units/{ <i>storage-control-unit-id</i> }/storage-paths/{ <i>storage-path-id</i> }, where { <i>storage-control-unit-id</i> } is the object-id of the containing storage control unit, and { <i>storage-path-id</i> } is the element-id of this storage path.
element-id	—	String (36)	The unique identifier for the storage path instance.
class	—	String (12)	The class of a storage path element object is "storage-path" .
parent	—	String/ URI	The parent of a storage path is its owning storage control unit, so the parent value is the canonical URI path for the storage control unit.
adapter-port-id	(w)(pc)	String/ URI	The canonical URI path for the storage adapter port at the opposite end of the storage path.
exit-switch-uri	(w)(pc)	String/ URI	The canonical URI path for the exit switch to which the control unit, and in non-cascaded switch configurations, the adapter, are connected. The value of this property may be null , which indicates the adapter and control unit have a point-to-point connection without any intervening switches. Default value: null

Table 189. Storage Control Unit object: storage path element object properties (continued)

Name	Qualifier	Type	Description
exit-port	(w)(pc)	String (2)	A two-character lowercase hexadecimal number that represents the port on the exit switch to which the control unit is connected. The value of this property will be null when the value of exit-switch-uri is null, and non-null when the value of exit-switch-uri is non-null. Default value: null

List Storage Control Units of a Storage Subsystem

The List Storage Control Units of a Storage Subsystem operation lists the storage control units managed by the storage subsystem with the given identifier.

HTTP method and URI

```
GET /api/storage-subsystems/{storage-subsystem-id}/storage-control-units
```

In this request, the URI variable *{storage-subsystem-id}* is the **object-id** of the Storage Subsystem object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
logical-address	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching logical-address property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-control-units	Array of storage-control-unit-info objects	Array of storage-control-unit-info objects, described in the next table. The returned array may be empty.

Each nested storage-control-unit-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Control Unit object.
name	String	The name property of the Storage Control Unit object.
logical-address	String	The logical-address property of the Storage Control Unit object.

Description

This operation lists the storage control units that are managed by the identified storage subsystem. The object URI, name and logical address are provided for each.

If the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **logical-address** query parameters are specified, the returned list is limited to those storage control units that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

A storage control unit is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage subsystem is the parent of a storage control unit but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage control units are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 407.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 190. List Storage Control Units of a Storage Subsystem: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage subsystem with the object ID <i>{storage-subsystem-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/storage-control-units
HTTP/1.1
x-api-session: 1eiJa5xbm9fj0z3bmiwbwnw37ks9esycx5tkzc41z65c6f1ezib
```

Figure 220. List Storage Control Units of a Storage Subsystem: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:55:39 GMT
content-type: application/json; charset=UTF-8
content-length: 156
{
  "storage-control-units": [
    {
      "logical-address": "50",
      "name": "Control unit 50",
      "object-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
    }
  ]
}

```

Figure 221. List Storage Control Units of a Storage Subsystem: Response

Define Storage Control Unit

The Define Storage Control Unit operation defines a new Storage Control Unit object within a parent storage subsystem.

HTTP method and URI

POST /api/storage-subsystems/{*storage-subsystem-id*}/operations/define-storage-control-unit

In this request, the URI variable {*storage-subsystem-id*} is the **object-id** of the Storage Subsystem object on which a storage control unit is to be defined.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Optional	The value to be set as the storage control unit's name property.
description	String (0-1024)	Optional	The value to be set as the storage control unit's description property.
logical-address	String (2)	Required	The value to be set as the storage control unit's logical-address property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/URI	The object-uri property of the newly defined Storage Control Unit object.

Description

This operation defines a storage control unit with the values specified in the identified storage subsystem and then returns its **object-uri** in the response body. An Inventory Change notification and a Property Change notification on the parent storage subsystem's **storage-control-unit-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the

object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned. If the storage subsystem identified by *{storage-subsystem-id}* already contains a storage control unit with the specified **name**, a 400 (Bad Request) status code is returned. If the storage subsystem identified by *{storage-subsystem-id}* already contains a storage control unit with the specified **logical-address**, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage control unit is defined and its properties are set to their corresponding request body contents field's values. If a field is not found in the request body, its property's value will be defaulted. The new storage control unit's URI is added to the parent storage subsystem's **storage-control-unit-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 409.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

<i>Table 191. Define Storage Control Unit: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage control unit with the name specified in the request body already exists within the parent storage subsystem with the object-id <i>{storage-subsystem-id}</i> .
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage subsystem with the object-id <i>{storage-subsystem-id}</i> does not exist on the HMC.
409 (Conflict)	447	A storage control unit with the logical-address specified in the request body already exists within the parent storage subsystem with the object-id <i>{storage-subsystem-id}</i> .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  define-storage-control-unit HTTP/1.1
x-api-session: 557u9y6zn6qwei7239zy3roucdrvzkr8ubni9ijykoe6eq12tu
content-type: application/json
content-length: 25
{
  "logical-address": "50"
}
```

Figure 222. Define Storage Control Unit: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:21:54 GMT
content-type: application/json; charset=UTF-8
content-length: 80
{
  "object-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}
```

Figure 223. Define Storage Control Unit: Response

Undefine Storage Control Unit

The Undefine Storage Control Unit operation removes a storage control unit definition.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/operations/undefine
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit to be removed.

Description

This operation removes a storage control unit definition. If the storage control unit contains storage paths or volume ranges, they will be removed as well. Inventory Change notifications on the removed storage control unit and each of its contained storage paths, and a Property Change notification on the parent storage subsystem's **storage-control-unit-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC.

If the storage control unit is mapped to any storage volume, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage control unit's storage paths and volume ranges are deleted and its URI is removed from the parent storage subsystem's **storage-control-unit-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 192. Undefine Storage Control Unit: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with the object ID <i>{storage-control-unit-id}</i> does not exist on the HMC.
409 (Conflict)	2	The storage control unit with the object-id <i>{storage-control-unit}</i> was busy and the request timed out.
	446	The storage control unit with the object-id <i>{storage-control-unit-id}</i> is mapped to one or more volumes in one or more storage groups.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/
  undefine HTTP/1.1
x-api-session: 4er4rp2aw9y5mgokxtwb8wz37v1mx4nyyj2iqx5gct66ewhcb4
content-type: application/json
```

Figure 224. Undefine Storage Control Unit: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:20:54 GMT

<No response body>
```

Figure 225. Undefine Storage Control Unit: Response

Get Storage Control Unit Properties

The Get Storage Control Unit Properties operation retrieves the properties of a single Storage Control Unit object.

HTTP method and URI

```
GET /api/storage-control-units/{storage-control-unit-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Control Unit object as defined in the [“Data model”](#) on page 404. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Control Unit object as defined in the [“Data model”](#) on page 404.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 413.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 193. Get Storage Control Unit Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with object-id <i>{storage-control-unit-id}</i> does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492 HTTP/1.1
x-api-session: 1skejiyvview4b70zzxk6y2s802fh4gpugbkvy4yddsambc5c8
```

Figure 226. Get Storage Control Unit Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:51:56 GMT
content-type: application/json;charset=UTF-8
content-length: 946
{
  "class": "storage-control-unit",
  "description": "LCU 50 in DS8870 A",
  "logical-address": "50",
  "name": "Control unit 50",
  "object-id": "69bf384a-94d5-11e8-8ffe-fa163e27d492",
  "object-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
  "parent": "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
  "storage-path-uris": [
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4eab9c-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4f02d6-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
  ],
  "volume-ranges": [
    {
      "ending-volume": "04",
      "starting-volume": "00",
      "type": "base"
    },
    {
      "ending-volume": "ff",
      "starting-volume": "fc",
      "type": "alias"
    }
  ]
}
```

Figure 227. Get Storage Control Unit Properties: Response

Update Storage Control Unit Properties

The Update Storage Control Unit Properties operation updates one or more of the writable properties of a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Data model” on page 404. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage control unit's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, or if the specified **logical-address** already exists within the parent storage subsystem, or if the **logical-address** field is present when target storage control unit is mapped to a storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the specified name already exists within the parent storage subsystem.

If the request body contents are valid, the storage subsystem's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 194. Update Storage Control Unit Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage control unit with the name specified in the request body already exists within its parent storage subsystem.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage control unit with the object-id <i>{storage-control-unit-id}</i> does not exist on the HMC.
409 (Conflict)	2	The Storage Control Unit object with the object-id <i>{storage-control-unit-id}</i> was busy and the request timed out.
	446	The logical-address field is present in the request body when the storage control unit with the object-id <i>{storage-control-unit-id}</i> is mapped to one or more volumes in a storage group.
	447	A storage control unit with the logical-address specified in the request body already exists within its parent storage subsystem.

Table 194. Update Storage Control Unit Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492 HTTP/1.1
x-api-session: 23qo36u6e3csdzyx6uroamwcduxrt2l3hk9phe8t9n763eeszl
content-type: application/json
content-length: 37
{
  "description": "LCU 50 in DS8870 A"
}
```

Figure 228. Update Storage Control Unit Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:23:28 GMT

<No response body>
```

Figure 229. Update Storage Control Unit Properties: Response

Add Volume Range

The Add Volume Range operation adds a volume range to a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/operations/add-volume-range
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit to which a volume range is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
starting-volume	String (2)	Required	The value to be set as the volume range's starting-volume property.
ending-volume	String (2)	Optional	The value to be set as the volume range's ending-volume property. If present, the value of the ending-volume field must be greater than or equal to the value of starting-volume .
type	String Enum	Optional	The value to be set as the volume range's type property.

Description

This operation adds a volume range to a storage control unit. A property-change notification for the storage control unit's **volume-ranges** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC.

If the storage control unit is mapped to any storage volume, or if the volume range overlaps with any volume range that already exists on the storage control unit, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may be because the value of the **ending-volume** field is less than the value of the **starting-volume** field.

If the request body contents are valid, the volume range is added to the **volume-ranges** property of the storage control unit.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 195. Add Volume Range: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	15	The value of the ending-volume field is less than the value of the starting-volume field.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with the object-id <i>{storage-control-unit-id}</i> does not exist on the HMC.
409 (Conflict)	2	The Storage Control Unit object with the object-id <i>{storage-control-unit-id}</i> was busy and the request timed out.
	446	The storage control unit with the object-id <i>{storage-control-unit-id}</i> is mapped to one or more volumes in a storage group.
	448	The volume range overlaps with one that already exists on the storage control unit object with the object-id <i>{storage-control-unit-id}</i> .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/
  add-volume-range HTTP/1.1
x-api-session: 3zu31yqjcjwa7plkqpsceibkye5gmssmmkxbf51ok5ve8kbjnw
content-type: application/json
content-length: 64
{
  "ending-volume": "04",
  "starting-volume": "00",
  "type": "base"
}
```

Figure 230. Add Volume Range: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:27:41 GMT

<No response body>
```

Figure 231. Add Volume Range: Response

Remove Volume Range

The Remove Volume Range operation removes a volume range from a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/operations/remove-volume-range
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit from which the volume range is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
starting-volume	String (2)	Required	The starting-volume property value of the volume range that is to be removed.

Description

This operation removes the entire volume range with the specified starting volume from a storage control unit. A Property Change notification for the storage control unit's **volume-ranges** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC.

If the storage control unit is mapped to any storage volume, or if a volume range with the specified **starting-volume** value does not exist on the storage control unit, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the volume range with the specified **starting-volume** is removed from the **volume-ranges** property of the storage control unit.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 196. Remove Volume Range: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with the object-id {storage-control-unit-id} does not exist on the HMC.
409 (Conflict)	2	The Storage Control Unit object with the object-id {storage-control-unit-id} was busy and the request timed out.
	446	The storage control unit with the object-id {storage-control-unit-id} is mapped to one or more volumes in a storage group.
	449	A volume range with the specified starting-volume value does not exist on the storage control unit object with the object-id {storage-control-unit-id}.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/  
  remove-volume-range HTTP/1.1  
x-api-session: 2guwrshqajcumqql9zxk5iighh2kxxzb5tkxqnuh9hct4etk30  
content-type: application/json  
content-length: 25  
{  
  "starting-volume": "00"  
}
```

Figure 232. Remove Volume Range: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:27:21 GMT

<No response body>
```

Figure 233. Remove Volume Range: Response

Create Storage Path

The Create Storage Path operation creates a new Storage Path element object within a parent storage control unit.

HTTP method and URI

POST /api/storage-control-units/{storage-control-unit-id}/storage-paths

In this request, the URI variable {storage-control-unit-id} is the **object-id** of the parent Storage Control Unit object of the new storage path.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
adapter-port-uri	String/ URI	Required	The value to be set as the storage path's adapter-port-uri property.
exit-switch-uri	String/ URI	Optional	The value to be set as the storage path's exit-switch-uri property.
exit-port	String (2)	Optional	The value to be set as the storage path's exit-port property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
element-uri	String/ URI	The element URI of the newly created Storage Path element object.

Description

This operation creates a storage path with the values specified on the identified storage control unit and then returns its element URI in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification and a Property Change notification on the parent storage control unit's **storage-path-uris** property are emitted asynchronously to this operation.

A storage path defines a logical connection between a storage control unit and an adapter port that flows between elements that are physically connected. If the storage path defines a direct connection between a storage control unit and an adapter port, the **endpoint-connection-uri** property of the adapter port must reference the storage control unit's storage subsystem. If the storage path references a switch, the storage subsystem must contain an endpoint connection to that switch and an endpoint connection must exist between the adapter port and any switch that resides in the same storage fabric.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the

object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC, or if **adapter-port-uri** or **exit-switch-uri** do not identify an adapter port or storage switch on the HMC, or if the port identified by the **exit-port** field is not defined on the storage switch referenced by the **exit-switch-uri** field, a 404 (Not Found) status code is returned. If the storage control unit identified by *{storage-control-unit-id}* already contains a storage path with the same set of property values, a 400 (Bad Request) status code is returned. If the storage control unit is mapped to any storage volume, or if the maximum number of storage ports is already defined for the control unit, or the specified exit port is not defined on the specified switch, or if no physical path connects the targeted storage control unit's storage subsystem to the specified adapter port through the specified storage switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **exit-port-id** field is present when the **exit-switch-uri** field is omitted, or the **exit-port-id** field is omitted when the **exit-switch-uri** field is present

If the request body contents are valid, the storage path is created and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted. The new storage path's URI is added to the parent storage control unit's **storage-path-uris** list property.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose port is identified by *{adapter-port-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 420.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 197. Create Storage Path: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage path with the same property values as those defaulted or specified in the request body already exists within the parent storage control unit with the object-id <i>{storage-control-unit-id}</i> .
	442	A corequisite condition on the presence of the exit-port-id field with respect to the presence of the exit-switch-uri field has been violated.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.

Table 197. Create Storage Path: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
404 (Not Found)	1	A storage control unit with the object-id {storage-control-unit-id} does not exist on the HMC.
	2	The adapter port referenced by adapter-port-uri does not exist on the HMC or the API user does not have object-access permission for it.
	441	The storage switch referenced by exit-switch-uri does not exist on the HMC.
	442	The port identified by exit-port is not defined on the storage switch referenced by exit-switch-uri .
409 (Conflict)	2	The storage control unit object with the object-id {storage-control-unit-id} was busy and the request timed out.
	442	The port identified by exit-port is not defined on the storage switch referenced by exit-switch-uri .
	446	The parent storage control unit with the object-id {storage-control-unit-id} is mapped to one or more volumes in a storage group.
	486	A new storage path cannot be created for the storage control unit with the object-id {storage-control-unit-id} because the maximum number of storage paths (8) for that control unit are already defined.
	497	An endpoint connection does not exist between the parent storage subsystem of the storage control unit with the object-id {storage-control-unit-id} and the storage switch identified by exit-switch-uri or adapter port identified by adapter-port-uri , or an endpoint connection does not exist between the adapter port identified by adapter-port-id and a storage switch in the storage fabric in which the storage switch identified by exit-switch-uri resides.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths
HTTP/1.1
x-api-session: 5iiqgy1nfzrlgjqlbuz3s06mqr8u2e0d3g5a7xf7kih4jb2cyb
content-type: application/json
content-length: 190
{
  "adapter-port-uri": "/api/adapters/20633658-941f-11e8-8625-fa163e27d492/storage-ports
/0",
  "exit-port": "00",
  "exit-switch-uri": "/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492"
}
```

Figure 234. Create Storage Path: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
8746e9c8-94d8-11e8-a5c5-fa163e3fe47d
cache-control: no-cache
date: Tue, 31 Jul 2018 15:43:53 GMT
content-type: application/json;charset=UTF-8
content-length: 132
{
  "element-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/
storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
}
```

Figure 235. Create Storage Path: Response

Delete Storage Path

The Delete Storage Path operation deletes a storage path.

HTTP method and URI

```
DELETE /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object and the URI variable *{storage-path-id}* is the element ID of the Storage Path element object to delete.

Description

This operation deletes a storage path. An Inventory Change notification and a Property Change notification on the parent storage control unit's **storage-path-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC, or if the element ID *{storage-path-id}* does not identify a storage path in the storage control unit.

If the storage control unit is mapped to any storage volume, a 409 (Conflict) status code is returned.

If the request body contents are valid, the identified storage path is deleted from the parent storage control unit's **storage-path-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 198. Delete Storage Path: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with the object-id {storage-control-unit-id} does not exist on the HMC.
	5	A storage path with element-id {storage-path-id} does not exist for the storage control unit.
409 (Conflict)	2	The storage control unit object with the object-id {storage-control-unit-id} was busy and the request timed out.
	446	The parent storage control unit with the object-id {storage-control-unit-id} is mapped to one or more volumes in a storage group.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: 6a7jbbd1iipk5mayxii3p742ujjzawofuthw3yyadlonn33gip
```

Figure 236. Delete Storage Path: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:40:41 GMT

<No response body>
```

Figure 237. Delete Storage Path: Response

Get Storage Path Properties

The Get Storage Path Properties operation retrieves the properties of a single Storage Path element object.

HTTP method and URI

```
GET /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
```

In this request, the URI variable {storage-control-unit-id} is the object ID of the Storage Control Unit object and the URI variable {storage-path-id} is the element ID of the Storage Path element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Path object as defined in the “Data model” on page 404. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Path element object as defined in the [Table 189 on page 406](#).

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, or if the element ID *{storage-path-id}* does not identify a storage path in the storage control unit, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the “Response body contents” on page 425.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 199. Get Storage Path Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	A storage control unit with object-id <i>{storage-control-unit-id}</i> does not exist on the HMC.
	5	A storage path with element-id <i>{storage-path-id}</i> does not exist for the storage control unit.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: 2bgudgv3iuzlc6ny30fs9v3pb0w43snsewp9dtt5h0qls29dpt
```

Figure 238. Get Storage Path Properties: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:44:39 GMT
content-type: application/json;charset=UTF-8
content-length: 466
{
  "adapter-port-uri":"/api/adapters/20633658-941f-11e8-8625-fa163e27d492/storage-ports/
  0",
  "class":"storage-path",
  "element-id":"8746e9c8-94d8-11e8-a5c5-fa163e3fe47d",
  "element-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/
  storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d",
  "exit-port":"00",
  "exit-switch-uri":"/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
  "parent":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}

```

Figure 239. Get Storage Path Properties: Response

Update Storage Path Properties

The Update Storage Path Properties operation updates one or more of the writable properties of a storage path.

HTTP method and URI

POST /api/storage-control-units/{*storage-control-unit-id*}/storage-paths/{*storage-path-id*}

In this request, the URI variable {*storage-control-unit-id*} is the object ID of the Storage Control Unit object and the URI variable {*storage-path-id*} is the element ID of the Storage Path element object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Data model” on page 404. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage path's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {*storage-control-unit-id*} does not identify a Storage Control Unit object on the HMC, or if the element ID {*storage-path-id*} does not identify a storage path in the storage control unit, or if **adapter-port-uri** or **exit-switch-uri** do not identify an adapter port or storage switch on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **exit-port-id** field is present when the **exit-switch-uri** field is omitted and the current value of the **exit-switch-uri** property is **null**, or the **exit-port-id** field is omitted and their current property values are **null** when the **exit-switch-uri** field is present, or because the updated storage path properties would be identical to an existing storage port in the parent storage control unit.

If the storage control unit is mapped to any storage volume, or the specified exit switch or port do not exist a 409 (Conflict) status code is returned.

If the request body contents are valid, the storage path's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
- Object-access permission to the adapter whose port is identified by *{adapter-port-id}*. This requirement only applies when updating the **adapter-port-id** property.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 200. Update Storage Path Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	The update would result in a storage path with the same property values as a storage path that already exists within the parent storage control unit with the object-id <i>{storage-control-unit-id}</i> .
	442	The update would put the storage port object into a state that would violate a corequisite condition on the presence of the exit-port-id property with respect to the presence of the exit-switch-uri property.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage – Storage Administrator tasks.
404 (Not Found)	1	The storage control unit with the object-id <i>{storage-control-unit-id}</i> does not exist on the HMC.
	2	The adapter port referenced by adapter-port-uri does not exist on the HMC or the API user does not have object-access permission for it.
	5	A storage path with element-id <i>{storage-path-id}</i> does not exist for the storage control unit on HMC.
	441	The storage switch identified by the exit-switch-uri field does not exist on the HMC.
	442	The port identified by exit-port is not defined on the storage switch referenced by exit-switch-uri .

Table 200. Update Storage Path Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	2	The Storage Control Unit object with the object-id {storage-control-id} was busy and the request timed out.
	446	The parent storage control unit with the object-id {storage-control-unit-id} is mapped to one or more volumes in a storage group.
	497	The new values specified in the adapter-port-uri or exit-switch-uri fields would put the storage path into a state where it would not flow over a physical path defined by the endpoint connections between the storage subsystem and the referenced exit switch and adapter port.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: bucvsrejregx2gv3p5l9rc5ylsupn4mr2o49vtlbuxkp3c3d6
content-type: application/json
content-length: 81
{
  "exit-switch-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
}
```

Figure 240. Update Storage Path Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:48:26 GMT

<No response body>
```

Figure 241. Update Storage Path Properties: Response

Inventory service data

Information about the Storage Control Units managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage control unit objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "storage-control-unit" are to be included. Information for a particular storage control unit is included only if the API user has access permission to that object as described in the `Get Storage Control Unit Properties` operation.

For each storage control unit to be included, the inventory response array includes the following:

- An array entry for the storage control unit object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Control Unit Properties”](#)

on page 413. That is, the data provided is the same as would be provided if a Get Storage Control Unit Properties operation were requested targeting this object.

- An array entry for each storage path associated with the storage control unit. For each such storage path, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for “Get Storage Path Properties” on page 424.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the Get Inventory response to describe a storage control unit. These objects would appear as multiple array entries in the response array:

```
{
  "class": "storage-control-unit",
  "description": "LCU 50 in DS8870 A",
  "logical-address": "50",
  "name": "Control unit 50",
  "object-id": "69bf384a-94d5-11e8-8ffe-fa163e27d492",
  "object-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
  "parent": "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
  "storage-path-uris": [
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4eab9c-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4f02d6-94d9-11e8-917c-fa163e3fe47d",
    "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
  ],
  "volume-ranges": [
    {
      "ending-volume": "04",
      "starting-volume": "00",
      "type": "base"
    },
    {
      "ending-volume": "ff",
      "starting-volume": "fc",
      "type": "alias"
    }
  ]
},
{
  "adapter-port-uri": "/api/adapters/1699fc1a-941f-11e8-8625-fa163e27d492/storage-ports/0",
  "class": "storage-path",
  "element-id": "7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
  "element-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
  "exit-port": "00",
  "exit-switch-uri": "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
  "parent": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}
```

Figure 242. Storage Control Unit object: Sample inventory data - Response

Storage Group object

A Storage Group object represents a single storage group associated with a DPM-enabled CPC. Storage groups define a set of FCP or FICON storage volume resources that can be attached to partitions. Storage group properties that are common to the group are represented by the Storage Group object. Each storage volume in the group is represented by the Storage Volume element object. When a storage group is attached to a partition, the group's fulfilled resources are virtualized and the partition's view of them is represented by a set of Virtual Storage Resource element objects. The Storage Group object APIs provide access to the set of storage groups that are associated with a CPC that is enabled for DPM. APIs exist to create and modify storage groups and their volumes, and delete and query storage group properties. APIs

also exist to query and fulfill the storage volume elements, and to query and update selected properties of the virtual storage resource elements of storage groups.

Rather than creating storage groups from scratch each time, a set of one or more storage templates can be defined and used when creating a new storage group. When a storage group is created from a storage template, the properties predefined in the template are copied into the new group, including any storage template volumes. Template property values can be overridden when creating the new group or by modifying the new group after it is created. See [“Storage Template object” on page 534](#) for more information on storage templates.

A storage group transitions through a number of states in its lifecycle. It is complicated by the fact that some of its attributes cannot be realized without changes to the configuration of the Storage Area Network (SAN) in which the storage resources defined in the group reside. Creation or modification of a storage group by a system administrator requires a subsequent fulfillment action by the SAN administrator before the storage resources in that group can be used by a partition. The nature of the actions required of the SAN administrator to fulfill a storage request differ depending on whether the storage group is defining FCP versus FICON resources. Fulfillment of FCP resources can be auto-detected, whereas fulfillment of FICON resources requires the SAN administrator to explicitly map the selected ECKD volumes to the requested volumes in the storage group.

Modifications to a group that require a fulfillment cycle are particularly complex because the new property values must be saved in addition to the active values, which are the ones that remain in effect if the storage group is currently active or activated before the modifications are fulfilled. To accommodate these needs, the storage group class contains two versions of each property that depends on fulfillment: a base version that represents the configured value and an active version (identified by an **"active-"** prefix to the property name), which represents the value of those properties for storage groups attached to partitions that are currently active, or when they are next activated. In a new group or volume that has not yet been fulfilled, the active property values will be **null** and the base property values will contain the values of that properties that were specified, or defaulted, during group creation. Once the resources in the group are all fulfilled, the base value will be copied into the active value. If a group or volume is modified, the new values specified in the update operation will be placed into the same-named base properties. The active values continue to represent the state of the storage in partitions that are currently active, or become active before the modifications are fulfilled.

A storage group has a **fulfillment-state** property that indicates a storage group's current fulfillment state. Storage volumes also have their own **fulfillment-state** property that indicates the fulfillment state of the individual storage volume. The table below lists important steps in a storage group's lifecycle, and their effect on the group's **fulfillment-state** and other properties, and on the partitions to which they are attached.

Lifecycle Step	Comment
A group is created	The fulfillment-state property of the group and each volume is set to "pending" . The base property values will be those specified or defaulted in the create operation, and the active property values will be null . An email is sent to the SAN administrator requesting fulfillment of the new group's storage resources. The storage group may be attached to a partition, but if activated, the group's storage resources will not be available to the partition.

Lifecycle Step	Comment
The group is fulfilled	The fulfillment-state property of the group and each volume is set to "complete". The base property values are copied to their corresponding active properties. The group's storage resources are now available to a partition. If the storage group is currently attached to an active partition, the resources will be dynamically made available to the partition. Note that there is no guarantee that all of a storage group's resource will be fulfilled at the same time, so it is possible for a storage group to be in a state of partial fulfillment. In that case, some of the properties will have been fulfilled, indicated by the base and active property values being equal, and some have not. The group's fulfillment-state property will be updated to "complete" only after all its resources have been fulfilled.
The group is modified	If volumes are added, or any group or volume property for which an active property exists is modified, the group fulfillment-state property is set to "pending". The base property values will be those specified in the update operation. The active property values will remain unchanged. New and deleted volumes URIs are added to or remove from the storage-volume-uris property of the group. If the storage group is attached to a partition that is currently active or subsequently activated, it will continue to see the original storage resources. The fulfillment-state property of any new volume, or any volume with a modified base property, is set to "pending". If storage volumes are created or deleted, or any group or volume property for which an active property exists is modified, an email is sent to the SAN administrator requesting fulfillment of the new or modified resources. The fulfillment state for deleted FCP storage volumes changes to "deleting". Deleted FCP volumes are removed from their parent storage group only after the system detects that the backing SAN resources have been deconfigured. Deleted FICON volumes are removed immediately and do not required action by the SAN administrator. Deleted volumes are included in the email to notify the SAN administrator that those storage resources may now be recovered.
The group's new and modified resources are fulfilled	The fulfillment-state property of the group and each volume is set to "complete" . The updated base property values are copied to the active properties. The group's new or updated storage resources are now available to a partition. If attached to an active partition, its resources will be dynamically changed.
The group is attached to a partition	If the partition is currently active, the group's storage fulfilled resources are dynamically added to the partition. A group can be attached to a partition any time after it is created.
The group is detached from a partition	If the partition is currently active, the group's storage resources are dynamically removed from the partition.
The group is deleted	The group must be detached from all partitions to which it is attached before it can be deleted. The delete of the group also deletes all its storage volumes.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, with the class-specific specializations identified in Table 202 on page 432. The Storage Group object does not support the operational status related properties.

Table 201. Storage Group object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Group object is of the form <code>/api/storage-groups/{storage-group-id}</code> where <code>{storage-group-id}</code> is the value of the object-id property of the Storage Group object.
object-id	—	String (36)	The unique identifier for the storage group instance.
parent	—	String/ URI	The parent of a storage group is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (13)	The class of a Storage Group object is "storage-group" .
name	(w)(pc)	String (1-64)	The display name specified for the storage group. The character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage groups associated to the same CPC.
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the storage group. Default value: If the storage group is created from a template, the value of the template's description property; otherwise an empty string.

Class specific additional properties

In addition to the properties defined via included schema, this object includes the following additional class-specific properties:

Table 202. Storage Group object: class specific properties

Name	Qualifier	Type	Description	Supported "type" values
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this storage group object.	All
type	—	String Enum	The type of storage resources managed by the storage group. Values: • "fcp" - Fibre Channel Protocol • "fc" - Fibre Connection Default value: If a storage template is not specified when creating a storage group, the type property is required; otherwise the value defaults to the value of the template's type property.	All

Table 202. Storage Group object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
shared	(w)(pc)	Boolean	true if this storage group can be attached to more than one partition; false if this storage group is dedicated to a single partition. This property's value cannot be changed from true to false if the group is attached to more than one partition. This property's value must be "false" when the value of the max-partitions property is 1. Default value: If a storage template is not specified when creating a storage group, the shared property defaults to true; otherwise the value defaults to the value of the template's shared property.	All
fulfillment-state	(pc)	String Enum	The current fulfillment state of the storage group. Values: • "complete" – All resources in this storage group have been fulfilled. • "pending" – The storage group has been created or modified , but has not yet been completely fulfilled by the SAN administrator. • "pending-with-mismatches" – One or more of the volumes of this storage group have a fulfillment-state property value of "pending-with-mismatches", "configuration-error" or "overprovisioned". This fulfillment state is only applicable if the type property value is "fcp". • "checking-migration" – The storage group was created automatically when the dpm-storage-management feature was enabled for the Defined CPC. Volumes that were configured for the virtual storage resources before the feature enablement are being identified. This fulfillment state is only applicable if the type property value is "fcp". • "incomplete" – The storage group migration is complete, but at least one of its storage volumes cannot be detected. This fulfillment state is only applicable if the type property value is "fcp". Default value: "pending"	All

Table 202. Storage Group object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
storage-volume-uris	(c)(pc)	Array of String/ URI	The list of storage volumes in this storage group. Each element in this array is the canonical URI path of a Storage Volume object. This value will change as storage volumes are added and removed from this storage group via the Modify Storage Group Properties operation. Default value: If the storage group is created from a template, a set of storage volumes based on the template volumes in the template's storage-template-volume-uris property; otherwise an empty array.	All
virtual-storage-resource-uris	(c)(pc)	Array of String/ URI	The list of virtual storage resources for this storage group. Each element in this array is the canonical URI path of a Virtual Storage Resource object. This value will change, and property change notification emitted, when the storage group has fulfilled resources and is attached to or detached from partitions, or when the storage group is already attached to a partition and resources are fulfilled again due to a size change. Default value: An empty array.	fcp
connectivity	(w)(pc)	Integer (1-255)	The number of adapters to utilize for this storage group. When the type value is "fc", the maximum value for the connectivity property is the lesser of 8 or the number of ports on FICON adapters of type "fc". When the type value is "fcp", the maximum value is the lesser of 255 or the number of ports on FICON adapters of type "fcp". Default value: If a storage template is not specified when creating a storage group, the connectivity property defaults to 2 for storage groups of type "fcp"; or to the maximum value for storage groups of type "fc"; otherwise the value defaults to the value of the template's connectivity property.	All

Table 202. Storage Group object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
active-connectivity	(pc)	Integer (1-255)	The value of connectivity that applies when the group is attached to partitions that are currently active or when the partitions are next activated. If the value of the fulfillment-state property is "pending" for newly created storage groups, this property value will be null, indicating it is not applicable. When the type value is "fc", the maximum value for the active-connectivity property is the lesser of 8 or the number of ports on FICON adapters of type "fc". When the type value is "fcp", the maximum value is the lesser of 255 or the number of ports on FICON adapters of type "fcp". This value will change to the value of connectivity, and property change notifications emitted, when the value of the connectivity property decreases, or after the value of the connectivity property increases and the new or modified storage group is fulfilled.	fcp
max-partitions	(w)(pc)	Integer	The maximum number of partitions to which this storage group can be attached. The value of max-partitions cannot exceed 1 if the value of the shared field is false. The value of max-partitions must be greater than 1 if the value of the shared field is true. The value of max-partitions cannot be decreased when the storage group is attached to any partition. The value of max-partitions cannot be decreased to a value that is less than the total number of partitions to which this storage group is currently attached. The minimum value for the max-partitions property is 1; the maximum is the value of the CPC object's maximum-partitions property. Default value: If a storage template is not specified when creating a storage group, the max-partitions property defaults to 2; otherwise the value defaults to the value of the template's max-partitions property.	fcp

Table 202. Storage Group object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
active-max-partitions	(pc)	Integer	When the value of the max-partitions property changes, the limit on the number of partitions to which the partition can be attached changes immediately. However, if the value increases, the storage resources for the additional partitions require fulfillment. When the storage group is attached to more than the active-max-partitions number of partitions, the most recently attached partitions will not see the storage group's storage resources until that fulfillment occurs. If the value of the fulfillment-state property is "pending" for newly created storage groups, this property value will be null, indicating it is not applicable. This value will change to the value of max-partitions, and property change notifications emitted, when the value of the max-partitions property decreases, or after the value of the max-partitions property increases and the new or modified storage group is fulfilled.	fcv
candidate-adapter-port-uris	(c)(pc)	Array of String/ URI	The list of adapter ports that are candidates for use in fulfilling connections to FCP devices. Each element in this array is an instance of the canonical URI path of a storage adapter port. This value will change, and property change notifications emitted, when candidate adapter port are added and removed via the Add Candidate Adapter Ports to an FCP Storage Group or Remove Candidate Adapter Ports from an FCP Storage Group operations.	fcv

Table 202. Storage Group object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
direct-connection-count	(w)(pc)	Integer (0-1000)	The number of additional virtual storage resource connections for the host that can be directly assigned to a guest virtual machine. A value of zero indicates this feature is disabled. The direct-connection-count property cannot be enabled (greater than zero) when the shared property is true. Default value: If a storage template is not specified when creating a storage group, the direct-connection-count property defaults to 0; otherwise the value defaults to the value of the template's direct-connection-count property.	fcv
unassigned-world-wide-port-names	(c)(pc)	Array of world-wide-port-name info	The list of information about the World Wide Port Names (WWPNs) that have been allocated to support this FCP storage group, but have not yet been assigned to a virtual storage resource. Each element in this array is an instance of a world-wide-port-name-info nested object, defined in Table 206 on page 447. This value will change, and property change notifications emitted, when the connectivity or max-partitions properties change, or when virtual storage resources are created or deleted. Default value: An empty array.	fcv

Storage Volume element object

A Storage Volume element object defines the size and usage of a single storage volume within its parent storage group.

Table 203. Storage Volume element object properties

Name	Qualifier	Type	Description	Supported storage group "type" values
element-id	—	String (32)	Unique identifier for the storage volume instance	All
element-uri	—	String/ URI	The canonical URI path for the storage port object, of the form <code>/api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}</code>, where <code>{storage-group-id}</code> is the object-id of the containing storage group, and <code>{storage-volume-id}</code> is the element-id of this storage volume.	All

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
parent	—	String/ URI	The parent of a storage volume is conceptually its owning storage group, and so the parent value is the canonical URI path for the storage group.	All
class	—	String (14)	The class of a storage volume element is "storage-volume" .	All
name	(w)(pc)	String (1-64)	The name of the storage volume. The character requirements on this property are the same as those of the name property described in the "Base managed object properties schema" on page 70. Names must be unique to the other storage volumes associated with the parent storage group. Default value: If the storage volume is created from a template, the value of the template volume's name property; otherwise a string that is currently of the form "# GiB {usage}{index}", where # is the size of the volume, {usage} is the usage, and {index} is a number that may or may not be present to ensure name uniqueness. This form is subject to change in the future.	All
description	(w)(pc)	String (0-100)	Arbitrary text providing additional descriptive information about the volume. Default value: If the storage volume is created from a template, the value of the template volume's description property; otherwise an empty string.	All

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
fulfillment-state	(pc)	String Enum	The current fulfillment state of the storage volume. Values: • "complete" – The storage volume resource has been fulfilled • "configuration-error" – A logical unit has been sensed on some, but not all, of the WWPNs defined to the parent storage group. This fulfillment state is only applicable if the type property value of the parent storage group is "fcp". • "deleting" – The storage volume has been deleted, but the SAN administrator has not yet unconfigured it logical unit. This fulfillment state is only applicable if the type property value of the parent storage group is "fcp". • "incomplete" – Migration for the parent storage group is complete, but this storage volume cannot be detected. This fulfillment state is only applicable if the type property value of the parent storage group is "fcp". • "overprovisioned" -Additional logical units have been sensed beyond what is required to fulfill this volume. This fulfillment state is only applicable if the type property value of the parent storage group is "fcp". • "pending" – The storage volume has been created or modified, but has not yet been fulfilled by the SAN administrator. • "pending-with-mismatches" – No logical unit with a size that exactly matches the size of this volume has been sensed. A logical unit with the closest size has been selected for this volume. This fulfillment state is only applicable if the type property value of the parent storage group is "fcp". If the eckd-type property has a value of "alias", the value of the fulfillment-state property will be null. Default value: "pending"	All

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
size	(w)(pc)	Float	The size in gibibytes (GiB) for this volume. If the value of the eckd-type property is "alias", this property value will be null, indicating it is not applicable, and the property cannot be updated. For volumes in storage group of type "fcp", the size property must be between 1.00 and 1,048,576.00. For volumes in storage groups of type "fc", with a model value other than "EAV", the size property is fixed at the value specified in the table below and cannot be written: Model - Maximum size "1" - 0.88 "2" - 1.76 "3" - 2.64 "9" - 7.92 "27" - 25.93 "54" - 51.86 For volumes with a model value of "EAV", size is writable and must be between 0.88 and 212489.20. Note however that the maximum is a theoretical size that may not be supported by an operating system. The value of the size property cannot be reduced once a storage volume is created. For volumes in storage groups of type "fc", the size value is related to the cylinders property. Specifying the size in GiB will also define the number of cylinders and vice versa. The size and cylinders properties cannot be specified together as fields in a create or modify operation. Default value: If the storage volume is created from a template, the value of the template volume's size property.	All

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
active-size	(pc)	Float	The value of size that applies when the volume is attached to partitions that are currently active or when the partitions are next activated. If the value of the fulfillment-state property is "pending" for a newly created volume, or if the value of the eckd-type property is "alias", this property value will be null, indicating it is not applicable. For volumes in storage group of type "fcp", the active-size property will be between 1.00 and 1,048,576.00 For volumes in storage groups of type "fc", with a model value other than "EAV", the active-size property is fixed at the value specified in the table below and cannot be written: Model - Maximum size "1" - 0.88 "2" - 1.76 "3" - 2.64 "9" - 7.92 "27" - 25.93 "54" - 51.86 For volumes with a model value of "EAV", active-size will be between 0.88 and 212489.20. This value will change to the value of size, and property change notifications emitted, when the new or modified storage volume is fulfilled.	All
usage	(w)(pc)	String Enum	The usage of the storage volume. Values: "boot" – this storage volume will contain a bootable image. "data" – this storage volume will contain data. "not-applicable" – the usage property does not apply to this storage volume. This value may never be used when creating or modifying a storage volume, but may appear as the value for this property in a response body. If the value of the usage property is "not-applicable", the property cannot be written. Default value: If the storage volume is created from a template, the value of the template volume's usage property; otherwise "data".	All
uuid	—	String (16, 32)	The sensed UUID of an FCP storage volume.	fcp

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
model	(w)(pc)	String Enum	The 3390 model designation for the storage. If the value of the eckd-type property is "alias", this property value will be null, indicating it is not applicable, and the property cannot be written. Values: "1" - Model 1 "2" - Model 2 "3" - Model 3 "9" - Model 9 "27" - Model 27 "54" - Model 54 "EAV" - Extended Address Volume The value of the model property cannot be changed to a smaller size once a storage volume is created. Default value: If the storage volume is created from a template, the value of the template volume's model property; otherwise this value is required when creating storage volumes of type "fc".	fc
active-model	(pc)	String Enum	The value of model that applies when the volume is attached to partitions that are currently active or when the partitions are next activated. If the value of the fulfillment-state property is "pending" for a newly created volume, or if the value of the eckd-type property is "alias", this property value will be null, indicating it is not applicable. Values: "1" - Model 1 "2" - Model 2 "3" - Model 3 "9" - Model 9 "27" - Model 27 "54" - Model 54 "EAV" - Extended Address Volume This value will change to the value of model, and property change notifications emitted, when the new or modified storage volume is fulfilled.	fc

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
cylinders	(w)(pc)	Integer (1113-268434453)	The size of the volume in cylinders. If the value of the eckd-type property is "alias", this property value will be null, indicating it is not applicable, and the property cannot be written. If the model value is other than "EAV", the cylinders property is fixed at the value specified in the table below and cannot be written: Model - Maximum cylinders "1" - 1113 "2" - 2226 "3" - 3339 "9" - 10017 "27" - 32760 "54" - 65520 For volumes with a model value of "EAV", cylinders is writable and has a maximum value of 268434453. Note however that this is a theoretical size that may not be supported by an operating system. The value of the cylinders property cannot be reduced once a storage volume is created. The cylinders value is related to the size property. Specifying the size in cylinders will also define the size and vice versa. The cylinders and size properties cannot be specified together as fields in a create or modify operation. Default value: If the storage volume is created from a template, the value of the template volume's cylinders property.	fc
device-number	(w)(pc)	String (4)	A four-byte lower case hexadecimal string defining the device number that is assigned by default when the FICON storage group containing this storage volume is attached to partitions. This value may be null, in which case the system auto-assigns a device number. Default value: If the storage volume is created from a template, the value of the template volume's device-number property; otherwise 0.	fc

Table 203. Storage Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported storage group "type" values
control-unit-uri	(pc)	String/ URI	The canonical URI of the logical control unit (LCU) in which the backing ECKD volume is defined. The value of this property will be null if a new "base" volume has not yet been fulfilled. This value will change, and property change notifications emitted, when the storage volume is fulfilled via the Fulfill FICON Storage Volume operation. Default value: null.	fc
eckd-type	—	String Enum	The type of the backing ECKD volume. Values: • "base" - The volume is an ECKD base volume. • "alias" - The volume is an ECKD alias volume. Default value: "base"	fc
unit-address	(pc)	String (2)	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the control unit. The value of this property will be null if a new "base" volume has not yet been fulfilled. This value will change, and property change notifications emitted, when the storage volume is fulfilled via the Fulfill FICON Storage Volume operation. Default value: null	fc
paths	—	Array of partition-volume-path-info objects	Information about all paths of an FCP storage volume for each partition to which the storage group is attached. Each element in the array is a partition-volume-path-info nested object, described in the next table. This field is present only when all the following conditions are true: • The storage group is attached to one or more partitions. • The fulfillment-state property is "complete" or "pending". • The uuid property is not null.	fcg

Each partition-volume-path-info object contains the following fields:

Table 204. *partition-volume-path-info* nested object

Name	Type	Description
partition-uri	String/ URI	The canonical URI path of the partition to which this virtual storage resource is attached.
device-number	String (4)	A 4-character lowercase hexadecimal string that contains the device number of the virtual resource.
target-world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains a WWPN that uniquely identifies a target port in one of the storage subsystems connected to the CPC that the storage group is associated with.
logical-unit-number	String (16)	A 16-character lowercase hexadecimal string that contains the logical unit number (LUN) representing the storage device configured in the storage controller.

Virtual Storage Resource element object

A virtual storage resource defines the virtualized view of a storage adapter as seen by a partition.

Table 205. *Virtual Storage Resource element object properties*

Name	Qualifier	Type	Description
element-id	—	String (36)	The unique identifier for the virtual storage resource instance.
element-uri	—	String/ URI	The canonical URI path for the Virtual Storage Resource element object, of the form <code>/api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}</code> , where <code>{storage-group-id}</code> is the object-id of the containing storage group, and <code>{virtual-storage-resource-id}</code> is the element-id of this virtual storage resource.
parent	—	String/ URI	The parent of a virtual storage resource is its owning storage group, so the parent value is the canonical URI path for the storage group.
class	—	String (24)	The class of a virtual storage resource element object is "virtual-storage-resource" .

Table 205. Virtual Storage Resource element object properties (continued)

Name	Qualifier	Type	Description
name	(w)(pc)	String (1-64)	The display name specified for the virtual storage resource. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other virtual storage resources within the parent storage group.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the virtual storage resource.
partition-uri	—	String/ URI	The canonical URI path of the partition to which this virtual storage resource is attached
device-number	(w)(pc)	String (4)	Device number of the virtual storage resource. The value must be a 4-digit lower case hexadecimal. The value must be unique across the device numbers of all other virtual storage resource elements and all NIC elements of type "iqd" or "osd" of the partition identified by partition-uri . Default value: Auto-generated
adapter-port-uri	(w)(pc)	String/ URI	The canonical URI path of the adapter Storage Port element to which this virtual storage resource is associated. The type field of the parent adapter must be "fcp" . The value of this property will be null if a candidate adapter has not been discovered to back this virtual storage resource. This value may not be set to null . This property may only be written after the virtual resource has been fulfilled, as indicated by a non- null current value.
world-wide-port-name	—	String (16)	A 16-character lower case hexadecimal string that contains the world wide port name of the FCP virtual storage resource.
world-wide-port-name-info	—	world-wide-port-name-info object	Information about the World Wide Port Name allocated to the virtual storage resource. The element is an instance of a world-wide-port-name-info nested object, as described in Table 206 on page 447.

Table 205. Virtual Storage Resource element object properties (continued)

Name	Qualifier	Type	Description
degraded-reasons	(pc)	Array of String Enum	The list of reasons for the degradation of the virtual storage resource. One or more of the following: • "adapter" - The status of the backing adapter of the virtual storage resource is either "service" or "exceptions". • "storage-configuration" - The virtual storage resource is not able to access some of the volumes. If the status of the partition that this virtual storage resource is associated with is neither "degraded" nor "reservation-error", this list will be empty.

A world-wide-port-name-info object defines properties relating to a single World Wide Port Name (WWPN). Each world-wide-port-name-info object contains the following fields.

Table 206. world-wide-port-name-info object: properties

Name	Type	Description
world-wide-port-name	String (16)	A 16-character lower case hexadecimal string that contains the world wide port name of the FCP virtual storage resource.
status	String Enum	The current status of the World Wide Port Name. Values: • "validated" - The CPC can sense all the storage volumes defined in the storage group using this WWPN. • "not-validated" - The CPC is unable to sense one or more of the storage volumes using this WWPN. • "unknown" - The WWPN status is unknown. This value will only appear in property change notifications for WWPNs that are removed from the parent storage group's unassigned-world-wide-port-names list property. • "incomplete" - The CPC is unable to sense all of the storage volumes defined in the storage group using this WWPN which were sensed and validated before.

List Storage Groups

The List Storage Groups operation lists the storage groups known to the target Console.

HTTP method and URI

GET /api/storage-groups

Query parameters:

Name	Type	Rqd/Opt	Description
cpc-uri	String/URI	Optional	Filter string to limit returned objects to those that have a matching cpc-uri property.

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
fulfillment-state	String Enum	Optional	Filter string to limit returned objects to those that have a matching fulfillment-state property. Value must be a valid storage group fulfillment-state property value.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property value. Value must be a valid storage group type property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-groups	Array of storage-group-info objects	Array of storage-group-info objects, described in the next table. The returned array may be empty.

Each nested storage-group-info object contains the following fields:

Field name	Type	Description
object-uri	String/URI	Canonical URI path (object-uri) of the Storage Group object.
cpc-uri	String/URI	The cpc-uri property of the Storage Group object.
name	String	The name property of the Storage Group object.
fulfillment-state	String Enum	The fulfillment-state property of the Storage Group object.
type	String Enum	The type property of the Storage Group object.

Description

This operation lists the storage groups that are known by the target Console. The object URI, name, fulfillment state, type and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage groups that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

If the **fulfillment-state** or **type** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage group **fulfillment-state** or **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage groups that have a **fulfillment-state** or **type** property matching the specified value. If the **fulfillment-state** or **type** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage groups that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage group is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a storage group, that object is simply omitted from the list but no error status code results.

If no storage groups are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Storage Group object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 448.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 207. List Storage Groups: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-groups HTTP/1.1
x-api-session: 3or8sd53i1g0swx7g3e29tnhzdc44tss6hzg3fr65bm29h1c3p
```

Figure 243. List Storage Groups: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 15:59:10 GMT
content-type: application/json; charset=UTF-8
content-length: 420
{
  "storage-groups": [
    {
      "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
      "fulfillment-state": "pending",
      "name": "FCP Storage Group",
      "object-uri": "/api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492",
      "type": "fcp"
    },
    {
      "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
      "fulfillment-state": "complete",
      "name": "FICON Group",
      "object-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492",
      "type": "fc"
    }
  ]
}

```

Figure 244. List Storage Groups: Response

Create Storage Group

The Create Storage Group operation creates a new storage group object. A storage group can optionally be created based on a storage template. In this case, the values of the specified template's properties are copied into the same-named properties of the new storage group. The template property values may be overridden by the presence of properties in the request body. The volumes created from the template can be overridden using the Modify Storage Group operation after the storage group is created. See the Storage Template Object for more information on storage templates. If a template is not specified during the create operation, the storage group's property values are initialized from the fields in the request body, or defaulted if omitted, as in a standard create operation. Once a storage group is created, it loses all association with the template from which it was created. Subsequent modifications to the template do not affect the storage group.

HTTP method and URI

POST /api/storage-groups

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Required	The value to be set as the storage group's cpc-uri property.
template-uri	String/ URI	Optional	The canonical URI of the storage template on which the new storage group is to be based. The template-uri field is not allowed in the request body if the type field is present.
name	String (1-64)	Required	The value to be set as the storage group's name property.
description	String (1-200)	Optional	The value to be set as the storage group's description property.

Field name	Type	Rqd/Opt	Description
type	String Enum	Required if template-uri is null; Prohibited otherwise	The value to be set as the storage group's type property. The type field is not allowed in the request body if the template-uri field is present.
shared	Boolean	Optional	The value to be set as the storage group's shared property.
connectivity	Integer	Optional	The value to be set as the storage group's connectivity property.
max-partitions	Integer	Optional	The value to be set as the storage group's max-partitions property. The max-partitions field is not allowed in the request body unless the type field is present and has a value of " fcp ", or the type field is omitted and the value of the type field in the referenced template is " fcp ".
direct-connection-count	Integer (0-1000)	Optional	The value to be set as the storage group's direct-connection-count property. The direct-connection-count field is not allowed in the request body unless the type field is present and has a value of " fcp ", or the type field is omitted and the value of the type field in the referenced template is " fcp ". The direct-connection-count field value cannot be greater than 0 if the shared field is omitted or set to true .
storage-volumes	Array of storage-volume-request-info nested objects	Optional	The set of volume properties for each of the storage volumes in the group. An array of one or more storage-volume-request-info nested objects, where each element defines the new property values of a storage volume that is to be created. The operation field of each nested object element must be set to " create ". For storage groups of type "fc" , the storage-volume-request-info nested object is defined in Table 212 on page 462 . For storage groups of type "fcp" , the storage-volume-request-info nested object is defined in Table 213 on page 463 .
email-to-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified via email of the new storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. Default value: null . No email will be sent.

Field name	Type	Rqd/Opt	Description
email-cc-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the new storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The email-cc-addresses field must be null when the email-to-addresses field is null. Default value: null. No one will be copied on the email.
email-insert	String	Optional	Text that is to be inserted in the email notification of the new storage group resources that require fulfillment. The text can include HTML formatting tags. The email-insert field must be null when the email-to-users field is null. Default value: null. An email without a special text insert will be sent.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the newly created Storage Group object.
element-uris	Array of String/ URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the storage-volumes field in the request body. If no volumes were specified when creating the storage group, the element-uris field will be an empty list.

Description

This operation creates a storage group with the values specified and then returns its **object-uri** and the **element-uris** of each storage volume that was created in the response body. The response also includes a Location header that provides the new storage group's URI. Inventory Change notifications for the new storage group and for each new storage volume identified in the **storage-volumes** field are emitted asynchronously to this operation.

A storage group can optionally be created based on a storage template, in which case the template's URI is given in the request body. If a template is specified, the values of its properties are copied to the same-named properties in the new storage group. Template values can be overridden by the explicit presence of a group property field in the request body. If a storage template is not specified, the new group's property values are set from the field values, or take a default value.

If the API user does not have action/task permission to the Configure Storage – System Programmer task, a 403 (Forbidden) status code is returned. If the object ID {cpc-id} does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the CPC identified by the **cpc-uri** field is already associated with a storage group with the specified name, or if two or more of the new storage volumes have the same name, or if the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400

(Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled or is not active, or if restrictions on the values of the **shared** and **max-partitions** property values, or the **shared** and **direct-connection-count** property values are violated, or if no adapters are configured to the specified type, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the **template-uri** field references a storage template that does not reside in the CPC referenced by the **cpc-uri** field.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage group type, or because both of, or neither of, the **size** and **cylinders** fields are defined for a new FICON storage volume.

If the request body contents are valid, the storage group is created and its properties are set to their corresponding template property values or request body content's field's values. If the **storage-volumes** field is present, the field values in each array element are used to define a new storage volume for the group. If a template is specified and the **storage-volumes** field is not present, the volumes in the template's **storage-template-volumes** property are copied into the new storage group. If a template is not specified and a field is not found in the request body, its property's value will be defaulted. The group's and volume's active property values will be set to **null**. If at least one storage volume is being created, or the storage group being created defines a property with a corresponding active property, the create requires action by the SAN administrator and the **fulfillment-state** property of the storage group is set to **"pending"**. Otherwise, the **fulfillment-state** property of the storage group is set to **"complete"**.

If the new storage group's **fulfillment-state** is **"pending"** and the **email-to-addresses** field is present in the request body, an email containing information about the storage group and volume resources that require fulfillment is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the creation of the storage group. An API client should assume that a storage group was created even though the request failed with a 409 (Conflict) status code and 491 reason code. The URI of the new Storage Group object is returned in the **error-details** field in the response body. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the Request Storage Group Fulfillment request.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-id** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 452.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 208. Create Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage group with the name specified in the request body is already associated with the CPC identified by the cpc-uri specified in the request body.
	15	The type field value is "fc" and an element of the storage-volumes field contains both the size and cylinders fields, or neither the size nor cylinders fields.
	18	A supplied property is not valid for a storage group's type.
	451	The email-cc-addresses or email-insert field is present in the request body without the email-to-addresses field.
	453	The name field for at least two of the nested entries in the storage-volumes field are the same.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	2	The CPC identified by the cpc-uri field does not exist on the HMC or the API user does not have object-access permission for it.
	444	The storage template identified by the template-uri field does not exist on the HMC or the API user does not have object-access permission for it.

Table 208. Create Storage Group: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	5	The CPC identified by the cpc-uri field is not enabled for DPM.
	8	The max-partitions field value conflicts with the shared field value.
	13	The CPC identified by the cpc-uri field does not support the dpm-storage-management feature.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	441	The storage template referenced by the template-uri field resides in a different CPC than the one referenced by the cpc-uri field.
	478	The type field is present in the request body when the template-uri field is present, or the type field is not present when the template-uri field is omitted.
	487	No adapters are configured for the storage protocol need to support a storage group with the specified type .
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, a new storage group will have been created. The error-details field of the response body contains a created-object-info object identifying the URI of the new Storage Group object. The created-object-info object is described in the next table.
	493	The email-to-addresses field is present in the request body, but the targeted console is not configured to support the sending of emails.
	495	The direct-connection-count field value conflicts with the shared field value.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Table 209. created-object-info object

Field name	Type	Description
created-object-uri	String/ URI	The URI of the new Storage Group object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups HTTP/1.1
x-api-session: d3dxdh07kglx7sc3wyvefv39ur355eddoghhw190cqc72qa8g
content-type: application/json
content-length: 194
{
  "connectivity":4,
  "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
  "name":"FICON Group",
  "storage-volumes":[
    {
      "model":"1",
      "operation":"create",
      "usage":"boot"
    }
  ],
  "type":"fc"
}
```

Figure 245. Create Storage Group: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492
cache-control: no-cache
date: Thu, 02 Aug 2018 19:26:13 GMT
content-type: application/json;charset=UTF-8
content-length: 202
{
  "element-uris":[
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
    ec738d80-9689-11e8-aa30-fa163e27d492"
  ],
  "object-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492"
}
```

Figure 246. Create Storage Group: Response

Delete Storage Group

The Delete Storage Group operation deletes a storage group.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/delete
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group to delete.

Request body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
email-to-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified via email of the storage group resources that are no longer needed. These email addresses will appear in the "to:" address list in the email that is sent. Default value: null . No email will be sent.

Field name	Type	Rqd/Opt	Description
email-cc-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the storage group resources that are no longer needed. These email addresses will appear in the "cc:" address list in the email that is sent. The email-cc-addresses field must be null when the email-to-addresses field is null. Default value: null. No one will be copied on the email.
email-insert	String	Optional	Text that is to be inserted in the email notification of the storage group resources that are no longer needed. The text can include HTML formatting tags. The email-insert field must be null when the email-to-users field is null. Default value: null. An email without a special text insert will be sent.

Description

This operation deletes a storage group. The storage group must be detached from all partitions before it can be deleted. The storage group's contained storage volume elements are also deleted. Inventory Change notifications for the deleted group and volume element object are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission.

If the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned. If the storage group is still attached to any partition, or if the CPC on which this storage group resource exists is not active, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails.

If the request body contents are valid, the identified storage group, and all its storage volumes, are deleted from the CPC. If the **email-to-addresses** field is present in the request body, an email containing information about the storage group and volume resources that may now be recovered is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present, its contents will be inserted into the email body. Note that a successful completion does not imply that the emails were delivered. A failure to send the email does not rollback the deletion of the storage group. An API client should assume that the storage group was deleted even though the request failed with a 409 (Conflict) status code and 491 reason code. Errors could be encountered at an email server after the request completes, for example due to an unknown email address.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 210. Delete Storage Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	451	The email-insert or email-cc-addresses field is present in the request body without the email-to-addresses field.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id { <i>storage-group-id</i> } does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	2	The storage group object with the object-id { <i>storage-group-id</i> } was busy and the request timed out.
	481	The storage group identified by { <i>storage-group-id</i> } is still attached to at least one partition.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the storage group will have been deleted.
	493	The email-to-addresses field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/delete
HTTP/1.1
x-api-session: 3da1nome5qa8uu9lbikho8ftvkd4lqv1bei7xlla1zcar8cbuq
content-type: application/json
content-length: 50
{
  "email-to-addresses": [
    "SamStorage@company.com"
  ]
}
```

Figure 247. Delete Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:20:54 GMT

<No response body>
```

Figure 248. Delete Storage Group: Response

Get Storage Group Properties

The `Get Storage Group Properties` operation retrieves the properties of a single Storage Group object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}
```

In this request, the URI variable `{storage-group-id}` is the object ID of the storage group object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Group object as defined in the [“Data model”](#) on page 432. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage group object as defined in the [“Data model”](#) on page 432.

If the object ID `{storage-group-id}` does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is `{storage-group-id}`.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 459.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 211. Get Storage Group Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492 HTTP/1.1
x-api-session: 2zf171u0zahnx479crjykbkccjiqxf4rstmhsi56mqgeup7kjs
```

Figure 249. Get Storage Group Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:32:02 GMT
content-type: application/json; charset=UTF-8
content-length: 1067
{
  "class": "storage-group",
  "connectivity": 4,
  "cpc-uri": "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
  "description": "A sample FICON storage group",
  "fulfillment-state": "complete",
  "name": "FICON Group",
  "object-id": "ec638c1e-9689-11e8-aa30-fa163e27d492",
  "object-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492",
  "parent": "/api/console",
  "shared": true,
  "storage-volume-uris": [
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492",
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/22de1d40-968a-11e8-a0a5-fa163e27d492",
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/70322410-968a-11e8-a0a5-fa163e27d492",
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/703e39c6-968a-11e8-a0a5-fa163e27d492",
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/704a5e40-968a-11e8-a0a5-fa163e27d492",
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/705620b8-968a-11e8-a0a5-fa163e27d492"
  ],
  "type": "fc"
}
```

Figure 250. Get Storage Group Properties: Response

Modify Storage Group Properties

The Modify Storage Group Properties operation updates one or more of the writable properties of a storage group.

HTTP method and URI

POST /api/storage-groups/{storage-group-id}/operations/modify

In this request, the URI variable {storage-group-id} is the object ID of the Storage Group object.

Request body contents

Fields for properties whose values are not to be changed by this operation can and should be omitted from the request body.

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Optional	The value to be set as the storage group's name property.
description	String (0-200)	Optional	The value to be set as the storage group's description property.
shared	Boolean	Optional	The value to be set as the storage group's shared property.
connectivity	Integer	Optional	The value to be set as the storage group's connectivity property. If the current value of the storage group's type property is " fcp ", modifying the connectivity property requires fulfillment action from the storage administrator. The storage group's fulfillment-state property will change to " pending ". If the email-to-list is also present in the request body, an email will be sent containing the new connectivity value.
max-partitions	Integer	Optional	The value to be set as the storage group's max-partitions property. The max-partitions field is not allowed in the request body unless the current value of the storage group's type property is " fcp ". Modifying the max-partitions property requires fulfillment action from the storage administrator. The storage group's fulfillment-state property will change to " pending ". If the email-to-list is also present in the request body, an email will be sent containing the new max-partitions value.
direct-connection-count	Integer (0-1000)	Optional	The value to be set as the storage group's direct-connection-count property. The direct-connection-count field is not allowed in the request body unless the current value of the storage group's type property is " fcp ". The direct-connection-count field value cannot be greater than 0 for shared storage groups.

Field name	Type	Rqd/Opt	Description
storage-volumes	Array of storage-volume-request-info nested objects	Optional	An array of storage-volume-request-info nested objects, where each element defines the existing storage volumes that are to be deleted or the new property values of a storage volume that is to be created or modified. The storage-volume-request-info nested object is defined in the tables that follow. If not specified, the storage group's volumes remain unchanged.
email-to-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified via email of the modified storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. Default value: null . No email will be sent.
email-cc-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the modified storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The email-cc-addresses field must be null when the email-to-addresses field is null . Default value: null . No one will be copied on the email.
email-insert	String	Optional	Text that is to be inserted in the email notification of the modified storage group resources that require fulfillment. The text can include HTML formatting tags. The email-insert field must be null when the email-cc-users field is null . Default value: null . An email without a special text insert will be sent.

Each nested storage-volume-request-info object contains the following fields:

Table 212. storage-volume-request-info nested object for "create" operations on "fc" storage volumes			
Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the property values for a new storage volume that is to be created. Value: "create"
name	String (1-64)	Optional	The value to be set as the storage volume's name property.
description	String (0-100)	Optional	The value to be set as the storage volume's description property.

Table 212. storage-volume-request-info nested object for **"create"** operations on **"fc"** storage volumes (continued)

Field name	Type	Rqd/Opt	Description
size	Float (0.88-212489.20)	Optional	The value to be set as the storage volume's size property. If the model is "EAV" , the size or cylinders fields (but not both) must be specified. If the model is not "EAV" , the size field may not be present in the request body.
usage	String Enum	Optional	The value to be set as the storage volume's usage property.
model	String Enum	Required	The value to be set as the storage volume's model property.
cylinders	Integer (1113-268434453)	Optional	The value to be set as the storage volume's cylinders property. If the model is "EAV" , the size or cylinders fields (but not both) must be specified. If the model is not "EAV" , the cylinders field may not be present in the request body.
device-number	String (4)	Optional	The value to be set as the storage volume's device-number property.

Table 213. storage-volume-request-info nested object for **"create"** operations on **"fcp"** storage volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the property values for a new storage volume that is to be created. Value: "create"
name	String (1-64)	Optional	The value to be set as the storage volume's name property.
description	String (0-100)	Optional	The value to be set as the storage volume's description property.
size	Float (1.00-1048576.00)	Required	The value to be set as the storage volume's size property.
usage	String Enum	Optional	The value to be set as the storage volume's usage property.

Table 214. storage-volume-request-info nested object for **"modify"** operations on **"fc"** storage volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the new property values for an existing storage volume that is to be modified. Value: "modify"

Table 214. storage-volume-request-info nested object for **"modify"** operations on **"fc"** storage volumes (continued)

Field name	Type	Rqd/Opt	Description
element-uri	String/ URI	Required	The canonical URI path for the storage volume element object that is being updated.
name	String (1-64)	Optional	The value to be set as the storage volume's name property.
description	String (0-100)	Optional	The value to be set as the storage volume's description property.
size	Float (0.88-212489.20)	Optional	The value to be set as the storage volume's size property. The size field is optional but may not be present in the request body if the cylinders field is present in the request body or if the model value is not "EAV". Modifying the size property requires fulfillment action from the storage administrator. The volume and parent storage group's fulfillment-state properties will change to "pending". If the email-to-addresses is also present in the request body, an email will be sent containing the new size value.
usage	String Enum	Optional	The value to be set as the storage volume's usage property.
model	String Enum	Optional	The value to be set as the storage volume's model property. Modifying the model property requires fulfillment action from the storage administrator. The volume and parent storage group's fulfillment-state properties will change to "pending". If the email-to-addresses is also present in the request body, an email will be sent containing the new model value.
cylinders	Integer (1113-268434453)	Optional	The value to be set as the storage volume's cylinders property. The cylinders field is optional but may not be preset in the request body if the size field is present in the request body or if the model value is not "EAV". Modifying the cylinders property requires fulfillment action from the storage administrator. The volume and parent storage group's fulfillment-state properties will change to "pending". If the email-to-addresses is also present in the request body, an email will be sent containing the new cylinders value.
device-number	String (4)	Optional	The value to be set as the storage volume's device-number property.

Table 215. storage-volume-request-info nested object for **"modify"** operations on **"fcp"** storage volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the new property values for an existing storage volume that is to be modified. Value: "modify"
element-uri	String/ URI	Required	The canonical URI path for the storage volume element object that is being updated.
name	String (1-64)	Optional	The value to be set as the storage volume's name property.
description	String (0-100)	Optional	The value to be set as the storage volume's description property.
size	Float (1.00-1048576.00)	Optional	The value to be set as the storage volume's size property. Modifying the size property requires fulfillment action from the storage administrator. The volume and parent storage group's fulfillment-state properties will change to "pending" . If the email-to-addresses is also present in the request body, an email will be sent containing the new size value.
usage	String Enum	Optional	The value to be set as the storage volume's usage property.

Table 216. storage-volume-request-info nested object for **"delete"** operations on **"fc"** or **"fcp"** storage volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object identifies an existing storage volume that is to be deleted. Value: "delete"
element-uri	String/ URI	Required	The canonical URI path for the storage volume element object that is being deleted.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
element-uris	Array of String/ URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the storage-volumes field in the request body. If the storage-volumes field did not contain any entries with operation equal to "create" , the element-uris field will be an empty list.

Description

This operation updates a storage group's properties with the values specified and then returns the **element-uris** of each storage volume that was created in the response body.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission.

If the group's **fulfillment-state** property value is **"checking-migration"**, or if the CPC on which this storage group resource exists is not active, or if the change would put the storage group into a state where its **shared** and **max-partitions** property values or **shared** and **direct-connection-count** property values conflict, or if the **shared** property is **false** for a storage group that is attached to more than one partition, or if the a storage volume modification would put the volume into a state where its **model**, **size** and **cylinders** property values conflict, or the **model**, **size** or **cylinders** properties are being reduced, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if a storage volume that is assigned as a partition's boot volume is deleted, or its **usage** property is changed.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage group type, or because the parent CPC is already associated with a storage group with the specified name, or because the operation would put the storage group into a state where two or more of its storage volumes would have the same name, or because both of, or neither of, the **size** and **cylinders** fields of a FICON storage volume are defined, or because the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or because any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, the storage group's properties are updated to their corresponding request body content's field's values. Optional fields may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified. The element URIs of each new and deleted storage volume will be added to, or removed from, the storage group's **storage-volume-uris** list property. If at least one property being modified has a corresponding active property, the update requires action by the SAN administrator and the **fulfillment-state** property of the storage group is set to **"pending"**. All active property values remain unchanged.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation. This includes a Property Change notification for the **storage-volume-uris** property if the operation creates or deletes storage volumes.

If the modified storage group's **fulfillment-state** is **"pending"** and the **email-to-addresses** field is present in the request body, an email containing information about the modified storage group and volume resources that require fulfillment is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the modification of the storage group. An API client should assume that the storage group was modified even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the Request Storage Group Fulfillment request.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 465](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 217. Modify Storage Group Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage group with the name specified in the request body is already associated with the CPC identified by the group's cpc-uri property.
	15	The storage group's type property value is "fc" and an element of the storage-volumes field contains both the size and cylinders fields, or an operation value of "create" and neither the size nor cylinders fields.
	18	A supplied property is not valid for a storage group's type.
	450	The storage group's type property value is "fc" and an element of the storage-volumes field has an operation value of "delete" and an element-uri value that references a storage volume with an eckd-type property value of "alias" .
	451	The email-cc-addresses or email-insert field is present in the request body without the email-to-addresses field.
	453	The operation would put the storage group into a state where the name property for at least two of its storage volumes would be the same.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id { <i>storage-group-id</i> } does not exist on the HMC or the API user does not have object-access permission for it.
	5	An element of the storage-volumes field has an element-uri value that is not a member of the storage group's storage-volume-uris array property value.

Table 217. Modify Storage Group Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	2	The storage group object with the object-id {storage-group-id} was busy and the request timed out.
	8	The operation would result in conflicting values for the max-partitions and shared property values.
	471	The value of the fulfillment-state property for the storage group with the object-id {storage-group-id} is "checking-migration" .
	475	The max-partitions field value is less than the number of partitions to which this storage group is currently attached, or the shared property is false when the number of partitions to which the partition is attached is more than one.
	490	The storage group's type property value is "fc" and an element of the storage-volumes field would put the storage volume into a state where its model , size and cylinders property values conflict.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the storage group will have been modified.
	493	The email-to-addresses field is present in the request body, but the targeted console is not configured to support the sending of emails.
	494	The size or cylinders field value is less than the storage volume's current size or cylinders , or the model field value represents a size that is smaller than that of the current model .
	495	The operation would result in conflicting values for the direct-connection-count and shared property values.
	499	A storage volume that is assigned as a partition's boot volume is being deleted, or its usage property is being changed.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, "Invoking API operations,"](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/modify HTTP/1.1
x-api-session: 55q0cufbqgz03k1s2bmz98nxx2ozf4e4sqzygw7q50st3zfqe
content-type: application/json
content-length: 352
{
  "description": "A sample FICON storage group",
  "storage-volumes": [
    {
      "description": "A Model 3 FICON data volume",
      "model": "3",
      "operation": "create"
    },
    {
      "description": "A Model 1 FICON boot volume",
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492",
      "operation": "modify"
    }
  ]
}
```

Figure 251. Modify Storage Group Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:27:45 GMT
content-type: application/json; charset=UTF-8
content-length: 130
{
  "element-uris": [
    "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/22de1d40-968a-11e8-a0a5-fa163e27d492"
  ]
}
```

Figure 252. Modify Storage Group Properties: Response

Resend Request

The Resend Request operation requests fulfillment of the storage group resources that require fulfillment. This operation can be invoked on any storage group with fulfillment state **"pending"**, **"incomplete"**, or **"pending-with-mismatches"**. The request will include the current resources that require fulfillment, and will therefore reflect any subsequent modifications or partial fulfillments that have occurred since the storage group was initially created or modified.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/resend-request
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object for which a fulfillment request should be sent.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
email-to-addresses	Array of String	Required	A set of one or more email addresses of the people that are to be notified via email of the storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent.
email-cc-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. Default value: null . No one will be copied on the email.
email-insert	String	Optional	Text that is to be inserted in the email notification of the storage group resources that require fulfillment. The text can include HTML formatting tags. Default value: null . An email without a special text insert will be sent.

Description

This operation notifies via email the new, deleted or modified storage group resources that require fulfillment action by the SAN administrator. The email will reflect the current fulfillment state of the storage group. The content of the email may differ from the email that was sent as a result of a previous Create Storage Group or Modify Storage Group Properties operation in the following ways:

- If the storage group was modified since the last time it was fulfilled, the email will include the accumulation of all storage group resource changes from the previous modify requests.
- The storage group resources that were part of the previous modify request but are now fulfilled will be omitted from the email.
- Storage group resources that were deleted in a modify request may not be included in the email.

If the API user does not have action/task permission to the **Configure Storage - System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission.

If the current storage group does not contain any new, deleted or modified resources that require fulfillment, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example, due to an unknown email address.

If the request body fails to validate, a 400 (Bad Request) status code is returned. A 400 (Bad Request) status code is also returned if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, an email is created that describes the new, modified, or deleted storage group resources that require fulfillment. If the **email-insert** field is present in the request body, its value is inserted into the body of the email. The email is then sent to the email addresses specified in the **email-to-addresses** field and copied to the email addresses specified in the **email-cc-addresses** field.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 218. Resend Request: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	1	The storage group is not in a valid state to perform the operation (must be in one of the following fulfillment states: "pending" , "incomplete" , or "pending-with-mismatches" , with at least one storage volume in "pending" or "deleting" fulfillment state.
	491	An error occurred when sending the email.
	493	SMTP is not configured on HMC to send email.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/a8856fb6-6d80-11e9-b629-fa163e2274e8/operations/resend-request HTTP/1.1
x-api-session: 4gbb8y8b9o3idupz1l7xcehupxmraxg5y874slliniecvsvx
content-type: application/json
content-length: 50
{
  "email-to-addresses": [
    "roberto.fdmmining@bnkincon.com"
  ]
}
```

Figure 253. Resend Request: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 08 May 2019 08:18:00 GMT

<No response body>
```

Figure 254. Resend Request: Response

Add Candidate Adapter Ports to an FCP Storage Group

The Add Candidate Adapter Ports to an FCP Storage Group operation adds a list of storage adapter ports to a storage group's candidate adapter ports list.

HTTP method and URI

POST /api/storage-groups/{storage-group-id}/operations/add-candidate-adapter-ports

In this request, the URI variable {storage-group-id} is the object ID of the storage group to which the candidate adapter ports are to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
adapter-port-uris	Array of String/URI	Required	A list of the adapter ports that are to be added to the storage group's candidate adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port.

Description

This operation adds a list of storage adapter ports to a storage group's candidate adapter ports list. These adapter ports become candidates for use as backing adapters when creating virtual storage resources when the group is attached to a partition. The adapter ports should have connectivity to the Storage Area Network (SAN). This operation only applies to storage groups of **type "fcp"**. Change notification for the storage group's **candidate-adapter-port-uris** property is emitted asynchronously to this operation.

Candidate adapter port may only be added before the CPC discovers a working communications path, indicated by a **"validated" status** on at least one of the parent storage group's WWPNs. After that point all adapter ports in the group are automatically detected and manually adding them is no longer possible.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-group-id} does not identify a storage group to which the API user has object-access permission, or if an element of the **adapter-port-uris** array does not identify a storage adapter object to which the API user has object-access permission, or if the storage group identified by the object ID {storage-group-id} is not of **type "fcp"**.

If any adapter port in the **adapter-port-uris** list is currently a member of the storage group's candidate adapter ports list, or if at least one WWPN in the group has been discovered, or if any adapter port in the **adapter-port-uris** list references a storage adapter port that does not reside in the target storage group's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because an adapter referenced in the **adapter-port-uris** array is not a FICON adapter.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are added to the storage group's candidate adapter ports list.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to each adapter containing the ports identified in the **adapter-port-uris** array.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 219. Add Candidate Adapter Ports to an FCP Storage Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An adapter referenced in the adapter-port-uris list has an adapter-family property value other than "ficon" .
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	2	An adapter containing a port referenced by a URI in the adapter-port-uris field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the object-id <i>{storage-group-id}</i> has a type value other than "fcp" .
409 (Conflict)	2	The storage group object with the object-id <i>{storage-group-id}</i> was busy and the request timed out.
	441	A storage adapter port in the adapter-port-uris list resides in a different CPC than the targeted storage group.
	471	The status property of at least one WWPN listed in the world-wide-port-names property of the storage group identified by <i>{storage-group-id}</i> has a value of "validated" .
	478	A storage adapter port in the adapter-port-uris list is already a current member of the candidate-adapter-port-uris list of the storage group object with the object-id <i>{storage-group-id}</i> .
	483	The adapter that contains the port referenced by an element in the adapter-port-uris field has a type value other than "fcp" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492/operations/add-candidate-
adapter-ports HTTP/1.1
x-api-session: sf2z7yhhiw90fj9rqevt7z3q7up3qfp62a8lraxrstzkkqv10
content-type: application/json
content-length: 93
{
  "adapter-port-uris": [
    "/api/adapters/13eb6396-941f-11e8-8625-fa163e27d492/storage-ports/0"
  ]
}
```

Figure 255. Add Candidate Adapter Ports to an FCP Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 16:07:37 GMT

<No response body>
```

Figure 256. Add Candidate Adapter Ports to an FCP Storage Group: Response

Remove Candidate Adapter Ports from an FCP Storage Group

The Remove Candidate Adapter Ports from an FCP Storage Group operation removes a list of storage adapter ports from a storage group's candidate adapter ports list.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/remove-candidate-adapter-ports
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group from which the candidate adapter port is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
adapter-port-uris	Array of String/ URI	Required	A list of the adapter ports that are to be removed from the storage group's candidate adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port

Description

This operation removes a list of storage adapter port from a storage group's candidate adapter ports list. This operation only applies to storage groups of **type "fcp"**. A Property Change notification for the storage group's **candidate-adapter-port-uris** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or if an element of the **adapter-port-uris** array does not identify a storage adapter object to

which the API user has object-access permission , or if the storage group identified by the object ID {storage-group-id} is not of **type "fcp"**.

If any adapter port in the **adapter-port-uris** list is not a current member of the storage group's candidate adapter ports list or is referenced by any of the group's virtual storage resources, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because an adapter referenced in the **adapter-port-uris** array is not a FICON adapter.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are removed from the storage group's candidate adapter ports list.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is {storage-group-id}.
- Object-access permission to each adapter containing the ports identified in the **adapter-port-uris** array.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 220. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An adapter referenced in the adapter-port-uris list has an adapter-family property value other than "ficon" .
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
	2	An adapter containing a port referenced by a URI in the adapter-port-uris field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the object-id {storage-group-id} has a type value other than "fcp" .

Table 220. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
409 (Conflict)	2	The storage group object with the object-id {storage-group-id} was busy and the request timed out.
	479	A storage adapter port in the adapter-port-uris list is not a current member of the candidate-adapter-port-uris list of the storage group object with the object-id {storage-group-id}.
	482	A storage adapter port in the adapter-port-uris list is referenced by at least one element of the virtual-storage-resource-uris in the storage group object with the object-id {storage-group-id}.
	488	A storage adapter port referenced in the adapter-port-uris field has been assigned to a storage volume in the storage group object with the object-id {storage-group-id}.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492/operations/remove-
candidate-adapter-ports HTTP/1.1
x-api-session: 4xtwzjxbhelnpvnnl0atkg5ut7cg6rqnlcdcvskvnj3zu8ubz
content-type: application/json
content-length: 93
{
  "adapter-port-uris": [
    "/api/adapters/13eb6396-941f-11e8-8625-fa163e27d492/storage-ports/0"
  ]
}
```

Figure 257. Remove Candidate Adapter Ports from an FCP Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 16:07:37 GMT

<No response body>
```

Figure 258. Remove Candidate Adapter Ports from an FCP Storage Group: Response

List Storage Volumes of a Storage Group

The List Storage Volumes of a Storage Group operation lists the storage volumes of the storage group with the given identifier.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
fulfillment-state	String Enum	Optional	Filter string to limit returned objects to those that have a matching fulfillment-state property. Value must be a valid storage volume fulfillment-state property value.
maximum-size	Integer	Optional	Filter value to limit returned object to those that have a size property that is less than or equal to maximum-size .
minimum-size	Integer	Optional	Filter value to limit returned object to those that have a size property that is greater than or equal to minimum-size .
usage	String Enum	Optional	Filter string to limit returned objects to those that have a matching usage property. Value must be a valid storage volume usage property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-volumes	Array of storage-volume-response-info objects	Array of storage-volume-response-info nested objects, described in the next table. The returned array may be empty.

Each nested storage-volume-response-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	Canonical URI path (element-uri) of the Storage Volume element object.
name	String	The name property of the storage volume element.
fulfillment-state	String Enum	The fulfillment-state property of the storage volume element.
size	Integer	The size property of the storage volume element.
usage	String Enum	The usage property of the storage volume element.

Description

This operation lists the storage volumes that are owned by the identified storage group. The element URI, name, fulfillment state, size and usage are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned

If the **name** query parameter is specified, the returned list is limited to those storage volumes that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **fulfillment-state** or **usage** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage volume **fulfillment-state** or **usage** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage volumes that have a **fulfillment-state** or **usage** property matching the specified value. If the **fulfillment-state** or **usage** parameter is omitted, this filtering is not done for that property name.

If the **minimum-size** query parameter is specified, the returned list is limited to those storage volumes that have a **size** property that is greater than or equal to the specified value. If the **maximum-size** query parameter is specified, the returned list is limited to those storage volumes that have a **size** property that is less than or equal to the specified value. When specified together, the **minimum-size** and **maximum-size** query parameters define a size range on which the volume is filtered. If either of these query parameters are omitted, the size filter is not bounded on one end. If both of these query parameters are omitted, no filtering on the **size** property is done.

If no storage volumes are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 477](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 221. List Storage Volumes of a Storage Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage group with the object ID <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes HTTP/1.1
x-api-session: 3xx8y87o2ajorslkb17weulfnfnyheic1l87kqvdksgngigr1e1r8
```

Figure 259. List Storage Volumes of a Storage Group: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:34:09 GMT
content-type: application/json; charset=UTF-8
content-length: 1279
{
  "storage-volumes": [
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/704a5e40-968a-11e8-a0a5-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "Alias",
      "size": 0.0,
      "usage": "not-applicable"
    },
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/70322410-968a-11e8-a0a5-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "Alias",
      "size": 0.0,
      "usage": "not-applicable"
    },
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/705620b8-968a-11e8-a0a5-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "Alias",
      "size": 0.0,
      "usage": "not-applicable"
    },
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "0.88 GiB Boot",
      "size": 0.88,
      "usage": "boot"
    },
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/22de1d40-968a-11e8-a0a5-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "2.64 GiB Data",
      "size": 2.64,
      "usage": "data"
    },
    {
      "element-uri": "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
        storage-volumes/703e39c6-968a-11e8-a0a5-fa163e27d492",
      "fulfillment-state": "complete",
      "name": "Alias",
      "size": 0.0,
      "usage": "not-applicable"
    }
  ]
}

```

Figure 260. List Storage Volumes of a Storage Group: Response

Get Storage Volume Properties

The Get Storage Volume Properties operation retrieves the properties of a single Storage Volume element object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the Storage Volume element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Volume object as defined in the [“Data model”](#) on page 432. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage volume object as defined in the [“Storage Volume element object”](#) on page 437.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission or if the element ID *{storage-volume-id}* does not identify a storage volume in the storage group.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 480.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 222. Get Storage Volume Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission to it.
	5	A storage volume with element-id <i>{storage-volume-id}</i> does not exist in the storage group on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532/storage-volumes/f3bdfdf0-4bba-11e9-b920-00106f23f532 HTTP/1.1
x-api-session: 5mpfhw1mcjtir9u1t4uyryduw48bp4c3fb95b696k4vex91uh6
```

Figure 261. Get Storage Volume Properties: Request

```

200 OK
server: Hardware management console API web server/2.0
cache-control: no-cache
date: Thu ,04 Apr 2019 19:36:08 GMT
content-type:application/json;charset=UTF-8
content-length: 1186
{
  "paths":[
    {
      "device-number":"0001",
      "logical-unit-number":"4016401400000000",
      "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
      "target-world-wide-port-name":"50050763070b46a6"
    },
    {
      "device-number":"0002",
      "logical-unit-number":"4016401400000000",
      "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
      "target-world-wide-port-name":"50050763070b46a6"
    },
    {
      "device-number":"0001",
      "logical-unit-number":"4016401400000000",
      "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
      "target-world-wide-port-name":"50050763070046a6"
    },
    {
      "device-number":"0002",
      "logical-unit-number":"4016401400000000",
      "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
      "target-world-wide-port-name":"50050763070046a6"
    }
  ],
  "fulfillment-state":"complete",
  "parent":"/api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532",
  "element-uri":"/api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532/storage-volumes/f3bdfdf0-4bba-11e9-b920-00106f23f532",
  "description":"",
  "element-id":"f3bdfdf0-4bba-11e9-b920-00106f23f532",
  "uuid":"6005076307FFC6A60000000000001614",
  "name":"10.00 GiB Boot",
  "usage":"boot", "active-size":10.0,
  "class":"storage-volume",
  "size":10.0
}

```

Figure 262. Get Storage Volume Properties: Response

Fulfill FICON Storage Volume

The Fulfill FICON Storage Volume operation maps the ECKD storage resource that has been configured by a SAN administrator to a storage volume that is part of a request for new or modified FICON storage.

HTTP method and URI

POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-ficon-storage-volume

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the storage volume element.

Request body contents

Field name	Type	Rqd/Opt	Description
control-unit-uri	String/ URI	Required	The canonical URI of the storage control unit in which the backing ECKD volume is defined.

Field name	Type	Rqd/Opt	Description
unit-address	String (2)	Required	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the storage control unit.

Description

This operation provides information about the ECKD storage resource that has been selected by a SAN administrator to fulfill a base storage volume that is part of a request for new or modified FICON storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to that storage resource.

Once a FICON storage volume has been fulfilled, the storage control unit and unit that has been assigned to it cannot be changed. If the **size**, **cylinders** or **model** properties of a volume are modified, the **fulfillment-state** of the volume is changed to **"pending"**, indicating that action may be necessary by the storage administrator to adjust the size of the backing ECKD volume. The **fulfillment-state** of the volume is changed back to **"complete"** by fulfilling the volume again with the exact same **control-unit-uri** and **unit-address**, indicating that the changes have been completed.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or the element ID *{storage-volume-id}* does not identify a storage volume in the storage group, or the **control-unit-uri** in the request body does not identify a storage control unit, or the current value of the group's **type** property is not **"fc"**.

If the storage volume's **fulfillment-state** property value is not **"pending"**, or if the CPC on which this storage group resource exists is not active, or if the storage group's **connectivity** property value does not equal the number of storage paths configured in the target storage control unit, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the identified unit address is not defined on the identified control unit, or if the backing ECKD volume is an alias volume or has already being used to fulfill another storage volume, or if the identified control unit and unit address would remap a previously fulfilled storage volume, or if the backing ECKD volume is in a different CPC than the target storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage volume's **fulfillment-state** property is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. The storage volume's **control-unit-uri** and **unit-address** properties are set to the same-named request body field values. The storage volume's **size** and **model** property values are copied to the **active-size** and **active-model** properties. Change notifications for the storage volume's **fulfillment-state**, **control-unit-uri**, **unit-address**, **active-size**, and **active-model** properties are emitted asynchronously to this operation. Change notification for the parent storage group's **fulfillment-state** property may be also emitted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 223. Fulfill FICON Storage Volume: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – Storage Administrator task.
404 (Not Found)	1	A storage group with the object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
	2	The storage control unit identified by control-unit-uri does not exist on the HMC.
	4	The storage group object with the object-id {storage-group-id} has a type value other than "fc" .
	5	A storage volume with element-id {storage-volume-id} does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	2	The storage group object with the object-id {storage-group-id} was busy and the request timed out.
	441	The storage control unit reference by the control-unit-uri field resides in a different CPC than the targeted storage volume.
	471	The current value of the fulfillment-state property of the storage volume identified by {storage-volume-id} is not "pending" .
	473	The storage volume identified by {storage-volume-id} has been previously fulfilled, and the values of the control-unit-uri and unit-address fields do not exactly match the previous values.
	484	The unit address identified by unit-address is not a member of a volume group on the control unit identified by control-unit-uri .
	485	The unit address identified by unit-address in the control unit identified by control-unit-uri has already been used to fulfill another FICON storage volume.
	486	The unit address identified by unit-address is a member of an alias volume group on the control unit identified by control-unit-uri .
	492	The connectivity property value of the storage volume identified by {storage-volume-id} does not equal the number of storage paths configured in the storage control unit identified by control-unit-uri .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492/operations/fulfill-ficon-storage-volume HTTP/1.1
x-api-session: 4iep1d8dyntia1rmppcfl0pmo8yghd190u1s14fu5133vevc7u
content-type: application/json
content-length: 109
{
  "control-unit-uri": "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
  "unit-address": "00"
}
```

Figure 263. Fulfill FICON Storage Volume: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:29:56 GMT

<No response body>
```

Figure 264. Fulfill FICON Storage Volume: Response

Fulfill FICON Storage Volumes

The Fulfill FICON Storage Volumes maps the ECKD storage resources that have been configured by a SAN administrator for one or more storage volumes that are part of a request for new or modified FICON storage. This operation is similar to the Fulfill FICON Storage Volume operation against a storage volume element, but allows the API client to fulfill multiple volumes with a single request, thus eliminating the overhead of multiple API requests.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/fulfill-ficon-storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

Field name	Type	Rqd/Opt	Description
fulfillment-info	Array of volume-fulfillment-info objects	Required	The information required to fulfill one or more FICON volumes. Each element in the array is a volume-fulfillment-info nested object defined in the next table.

Each nested volume-fulfillment-info object contains the following fields:

Field name	Type	Description
storage-volume-uri	String/URI	The canonical URI of the storage volume that is being fulfilled.
control-unit-uri	String/URI	The canonical URI of the storage control unit in which the backing ECKD volume is defined.
unit-address	String (2)	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the storage control unit.

Description

This operation provides information about the ECKD storage resources that have been selected by a SAN administrator to fulfill one or more base storage volumes that are part of a request for new or modified FICON storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to those storage resources.

Once a FICON storage volume has been fulfilled, the storage control unit and unit that has been assigned to it cannot be changed. If the **size**, **cylinders** or **model** properties of a volume are modified, the **fulfillment-state** of the volume is changed to **"pending"**, indicating that action may be necessary by the storage administrator to adjust the size of the backing ECKD volume. The **fulfillment-state** of the volume is changed back to **"complete"** by fulfilling the volume again with the exact same **control-unit-uri** and **unit-address**, indicating that the changes have been completed.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or any **storage-volume-uri** in a nested object within the **fulfillment-info** request field does not identify a storage volume in the storage group, or any **control-unit-uri** in a nested object within the **fulfillment-info** request field does not identify a storage control unit, or the current value of the group's **type** property is not **"fc"**.

If the storage volume's **fulfillment-state** property value is not **"pending"**, or if the CPC on which this storage group resource exists is not active, or if the storage group's **connectivity** property value does not equal the number of storage paths configured in any identified storage control unit, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if any identified unit address is not defined on the identified control unit, or if any backing ECKD volume is an alias volume or has already being used to fulfill another storage volume, or if any identified control unit and unit address would remap a previously fulfilled storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each of the storage volumes is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. The **control-unit-uri** and **unit-address** properties for each storage volume are set to the same-named values in the nested object that references that volume. The **size** and **model** property values for each storage volume are copied to the **active-size** and **active-model** properties. Change notifications for each storage volume's **fulfillment-state**, **control-unit-uri**, **unit-address**, **active-size**, and **active-model** properties are emitted asynchronously to this operation. Change notification for the parent storage group's **fulfillment-state** property may be also emitted.

If an unexpected error occurs when processing a valid request, a 500 (Server Error) status code is returned. In such a case, the targeted storage volumes may be left in a mixed state, with some being fulfilled and others not. The **error-details** field in the response body will contain an array of storage volume URIs that identify the storage volumes that were not fulfilled.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 224. Fulfill FICON Storage Volumes: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – Storage Administrator task.
404 (Not Found)	1	A storage group with the object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
	2	The storage control unit identified by control-unit-uri in any volume-fulfillment-info nested object does not exist on the HMC.
	4	The storage group object with the object-id {storage-group-id} has a type value other than "fc" .
	445	A storage volume identified by storage-volume-uri in any volume-fulfillment-info nested object does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	2	The storage group object with the object-id {storage-group-id} was busy and the request timed out.
	471	The current value of the fulfillment-state property of the storage volume identified by storage-volume-uri in any volume-fulfillment-info nested object is not "pending" .
	473	A storage volume identified by storage-volume-uri in any volume-fulfillment-info nested object has been previously fulfilled, and the values of the associated control-unit-uri and unit-address nested fields do not exactly match the previous values.
	484	A unit address identified by unit-address in any volume-fulfillment-info nested object is not a member of a volume group on the control unit identified by the associated control-unit-uri .
	485	The unit address identified by unit-address in any volume-fulfillment-info nested object in the control unit identified by the associated control-unit-uri has already been used to fulfill another FICON storage volume.
	486	The unit address identified by unit-address in any volume-fulfillment-info nested object is a member of an alias volume group on the control unit identified by the associated control-unit-uri .
	492	The connectivity property value of a storage volume identified by storage-volume-uri in any volume-fulfillment-info nested object does not equal the number of storage paths configured in the storage control unit identified by the associated control-unit-uri .

Table 224. Fulfill FICON Storage Volumes: HTTP status and reason codes (continued)		
HTTP error status code	Reason code	Description
500 (Server Error)	Various	An unexpected error occurred when processing a valid request. The error-details field in the response body will contain a failed-storage-volume-uris field with an array of canonical URIs of the storage volumes that were not fulfilled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/operations/fulfill-ficon-storage-volumes HTTP/1.1
x-api-session: 1mlog3c15tze37kr5z9q5gugo6c9k959sfyzd4tjr03oy4vfw
content-type: application/json
content-length: 514
{
  "fulfillment-info": [
    {
      "control-unit-uri": "/api/storage-control-units/4423c72a-057e-11ea-82b4-fa163e2a9969",
      "storage-volume-uri": "/api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/storage-volumes/b336cb04-057d-11ea-8f2d-fa163e2a9969",
      "unit-address": "00"
    },
    {
      "control-unit-uri": "/api/storage-control-units/44a4b8b2-057e-11ea-82b4-fa163e2a9969",
      "storage-volume-uri": "/api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/storage-volumes/b3381982-057d-11ea-8f2d-fa163e2a9969",
      "unit-address": "01"
    }
  ]
}
```

Figure 265. Fulfill FICON Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 12 Nov 2019 19:08:28 GMT

<No response body>
```

Figure 266. Fulfill FICON Storage Volumes: Response

Usage notes

There is a limit on the request body size for the Fulfill FICON Storage Volumes operation of 256KB. This puts an effective limit of about 1050 volumes that can be fulfilled with a single request.

Fulfill FCP Storage Volume

The Fulfill FCP Storage Volume operation provides information about the storage resource that has been selected by a SAN administrator to fulfill a boot storage volume that is part of a request for new FCP storage.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-fcp-storage-volume
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the storage volume element.

Request body contents

Field name	Type	Rqd/Opt	Description
world-wide-port-name	String (16)	Required	A 16-character lower case hexadecimal string that contains the world wide port name of the FCP storage controller containing the SCSI boot device.
logical-unit-number	String (16)	Required	A 16-character lower case hexadecimal string that contains the logical unit number (LUN) of the SCSI boot device.
adapter-port-uri	String/URI	Required	The canonical URI of the storage adapter port that is to be assigned to the storage volume.

Description

This operation provides information about the storage resource that has been selected by a SAN administrator to fulfill a storage volume that is part of a request for new FCP storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to that storage resource. Only FCP boot volumes may be manually fulfilled. Other volumes will be automatically fulfilled when the CPC detects communication paths to them. FCP volumes may be manually fulfilled only before the CPC discovers a working communications path, indicated by a **"validated"** status on at least one of the parent storage group's WWPNs. After that point, all volumes in the group are automatically fulfilled and manual fulfillment is no longer possible.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or the element ID *{storage-volume-id}* does not identify a storage volume in the group, or the **adapter-port-uri** field does not identify an adapter object to which the API user has object-access permission or identifies an adapter with a **type** value other than **"fcp"**, or the current value of the group's **type** property is not **"fcp"**, or the current value of the volume's **usage** property is not **"boot"**.

If the **status** property of at least one WWPN in the parent group's **world-wide-port-names** property value is **"validated"**, or if the **adapter-port-uri** field references a storage adapter port that does not reside in the target storage group's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage volume's **fulfillment-state** property is changed to **"complete"**. Change notification for the storage volume's **fulfillment-state** property is emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

- Object-access permission to the adapter containing the storage adapter port identified by the **adapter-port-uri** field.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 225. Fulfill FCP Storage Volume: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – Storage Administrator task.
404 (Not Found)	1	A storage group with the object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
	2	The adapter object containing the storage adapter port referenced in the adapter-port-uri field does not exist on the HMC, or the API user does not have object-access permission to it.
	4	The storage group object with the object-id {storage-group-id} has a type value other than "fcp" .
	5	A storage volume with element-id {storage-volume-id} does not exist in the storage group on the HMC.
	6	The adapter containing the storage adapter port referenced in the adapter-port-uri field has a type value other than "fcp" .
	442	The storage volume object with the object-id {storage-group-id} has a usage value other than "boot" .
409 (Conflict)	2	The parent storage group with object-id {storage-group-id} was busy and the request timed out.
	441	The storage adapter port referenced by the adapter-port-uri field resides in a different CPC than the targeted storage group.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/
2d71eeae-9fe9-11e8-b163-fa163e3c2af4/operations/fulfill-fcp-storage-volume HTTP/1.1
x-api-session: 2315cbdr8401gk09zmkpsfzwykutdinjxz1u1sxqxa0chulv2v
content-type: application/json
content-length: 177
{
  "adapter-port-uri": "/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
  "logical-unit-number": "93e4118a8b42c859",
  "world-wide-port-name": "a1b2c3d4e5f60002"
}
```

Figure 267. Fulfill FCP Storage Volume: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:49:28 GMT

<No response body>
```

Figure 268. Fulfill FCP Storage Volume: Response

Usage notes

It is not required to manually fulfill FCP storage volumes. If the Fulfill FCP Storage Volume request is not made targeting an FCP boot volume, the WWPN and LUN will be assigned algorithmically when the volume is sensed. Manual fulfillment can optionally be performed to speed up the process when the WWPN and LUN information is known, for example when the LUN configuration is done automatically.

Accept Mismatched FCP Storage Volumes

The Accept Mismatched FCP Storage Volumes operation completes the fulfillment process for selected volumes of an FCP storage group that have been flagged as possibly being mismatched or overprovisioned.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/accept-mismatched-storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
storage-volume-uris	Array of String/URI	Required	The canonical URIs of the mismatched storage volumes that are to be accepted.

Description

This operation accepts FCP storage volumes that have been flagged as being mismatched or overprovisioned.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access

permission, or one of the storage volumes identified in the **storage-volume-uris** field in the request body does not identify a storage volume in the storage group, or the current value of the group's **type** property is not **"fcp"**.

If the target storage group is busy, or if one of the storage volumes identified in the **storage-volume-uris** field has a **fulfillment-state** property value that is not **"pending-with-mismatches"** or **"overprovisioned"**, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each storage volume identified in the **storage-volume-uris** field is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. Change notifications for each storage volume's **fulfillment-state** property are emitted asynchronously to this operation. A change notification for the parent storage group's **fulfillment-state** property may be also emitted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 226. Accept Mismatched FCP Storage Volumes: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the object-id <i>{storage-group-id}</i> has a type value other than "fcp" .
	5	A storage volume identified in the storage-volume-uris array field does not exist in the storage group with the object-id <i>{storage-group-id}</i> .
409 (Conflict)	2	The storage group object with the object-id <i>{storage-group-id}</i> was busy and the request timed out.
	471	The current value of the fulfillment-state property of one of the storage volumes identified in the storage-volume-uris array field is not "pending-with-mismatches" or "overprovisioned" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/operations/accept-mismatched-
storage-volumes HTTP/1.1
x-api-session: 5b4tzc4fkakbrg4qsprt1ha84vvwdauch9y2x7a15wew0ann58
content-type: application/json
content-length: 138
{
  "storage-volume-uris": [
    "/api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/storage-volumes/cb61f800-
    e29e-11e8-9879-fa163e9c462b"
  ]
}
```

Figure 269. Accept Mismatched FCP Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 07 Nov 2018 15:42:06 GMT

<No response body>
```

Figure 270. Accept Mismatched FCP Storage Volumes: Response

Reject Mismatched FCP Storage Volumes

The Reject Mismatched FCP Storage Volumes operation rejects selected volumes of an FCP storage group that have been flagged as being possibly mismatched or overprovisioned.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/reject-mismatched-storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
storage-volume-uris	Array of String/ URI	Required	The canonical URIs of the mismatched storage volumes that are to be rejected.
email-to-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified via email of the rejected storage group resources. These email addresses will appear in the "to:" address list in the email that is sent. If not specified, no email will be sent.

Field name	Type	Rqd/Opt	Description
email-cc-addresses	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the rejected storage group resources. These email addresses will appear in the "cc:" address list in the email that is sent. The email-cc-addresses field must be omitted when the email-to-addresses field is omitted.
email-insert	String	Optional	Text that is to be inserted in the email notification of the rejected storage group resources. The text can include HTML formatting tags. The email-insert field must be omitted when the email-to-addresses field is omitted. If not specified, an email without a special text insert will be sent.

Description

This operation rejects FCP storage volumes that have been flagged as being mismatched or overprovisioned. The storage administrator is requested to change the storage configuration to delete the identified volumes.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or one of the storage volumes identified in the **storage-volume-uris** field in the request body does not identify a storage volume in the storage group, or the current value of the group's **type** property is not **"fcp"**.

If the target storage group is busy, or if one of the storage volumes identified in the **storage-volume-uris** field has a **fulfillment-state** property value that is not **"pending-with-mismatches"** or **"overprovisioned"**, or if an email address is specified when the target console does not support sending emails, a 409 (Conflict) status code is returned.

If the request body fails to validate, or if the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each storage volume identified in the **storage-volume-uris** field is changed to **"deleting"**. Change notifications for each storage volume's **fulfillment-state** property are emitted asynchronously to this operation.

If the **email-to-addresses** field is present in the request body, an email containing information about the storage volume resources that have been rejected is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the changes to the volume fulfillment states. An API client should assume that the **fulfillment-state** property of each storage volume was changed to **"deleting"** even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

<i>Table 227. Reject Mismatched FCP Storage Volumes: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	451	The email-cc-addresses or email-insert field is present in the request body without the email-to-addresses field.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage group with the object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the object-id <i>{storage-group-id}</i> has a type value other than "fcp" .
	5	A storage volume identified in the storage-volume-uris array field does not exist in the storage group with the object-id <i>{storage-group-id}</i> .
409 (Conflict)	2	The storage group object with the object-id <i>{storage-group-id}</i> was busy and the request timed out.
	471	The current value of the fulfillment-state property of one of the storage volumes identified in the storage-volume-uris array field is not "pending-with-mismatches" or "overprovisioned" .
	491	An error occurred when sending the email. This failure applies only to the ending of the email. If this reason code is returned, the fulfillment state of all storage volumes will have been modified.
	493	The email-to-addresses field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/operations/reject-mismatched-
storage-volumes HTTP/1.1
x-api-session: 3q85scpr0jy2cxmt86ordrc5typ19zpiy5szvxeb0d4duwzz8
content-type: application/json
content-length: 138
{
  "storage-volume-uris": [
    "/api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/storage-volumes/fe7829bc-
    e29e-11e8-8ee9-fa163e9c462b"
  ]
}
```

Figure 271. Reject Mismatched FCP Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 07 Nov 2018 15:44:38 GMT
<No response body>
```

Figure 272. Reject Mismatched FCP Storage Volumes: Response

List Virtual Storage Resources of a Storage Group

The List Virtual Storage Resources of a Storage Group operation lists the virtual storage resources of the storage group with the given identifier.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/virtual-storage-resources
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
device-number	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching device-number property.
adapter-port-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching adapter-port-uri property. Specify an empty query parameter value to select objects that have an adapter-port-uri property value of null, for example: "adapter-port-uri="
partition-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching partition-uri property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
virtual-storage-resources	Array of virtual-storage-resource-info objects	Array of virtual-storage-resource-info objects, described in the next table. The returned array may be empty.

Each nested virtual-storage-resource-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	The canonical URI path (element-uri) of the virtual Storage Resource element object.
name	String	The name property of the virtual storage resource element.
device-number	String	The device-number property of the virtual storage resource element.
adapter-port-uri	String/ URI	The adapter-port-uri property of the virtual storage resource element.
partition-uri	String/ URI	The partition-uri property of the virtual storage resource element.

Description

This operation lists the virtual storage resources that are owned by the identified storage group. The element URI, name, device number and associated adapter port and partition are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or does not identify an FCP storage group,, a 404 (Not Found) status code is returned

If the **name** or **device-number** query parameter is specified, the returned list is limited to those virtual storage resources that have a **name** or **device-number** property matching the specified filter pattern. If the **name** or **device-number** parameter is omitted, the filtering on the omitted property name is not done.

If the **adapter-port-uri** or **partition-uri** query parameter is specified, the returned list is limited to those virtual storage resources that have a matching **adapter-port-uri** or **partition-uri** property. If the **adapter-port-uri** or **partition-uri** parameter is omitted, the filtering on the omitted property name is not done.

If no virtual storage resources are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 495](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 228. List Virtual Storage Resources of a Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage group with the object ID <i>{storage-group-id}</i> does not exist on the HMC, or is a storage group of the wrong type , or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources
HTTP/1.1
x-api-session: 46zfkryyht0wrf7v8fjxfmza830u42yuob8p0ct727mr2so2x
```

Figure 273. List Virtual Storage Resources of a Storage Group: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:50:54 GMT
content-type: application/json;charset=UTF-8
content-length: 737
{
  "virtual-storage-resources":[
    {
      "adapter-port-uri":"/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/
storage-ports/0",
      "device-number":"0000",
      "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
virtual-storage-resources/4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
      "name":"vhba_FCP Storage Group0",
      "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4"
    },
    {
      "adapter-port-uri":"/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/
storage-ports/0",
      "device-number":"0001",
      "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
virtual-storage-resources/4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
      "name":"vhba_FCP Storage Group1",
      "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4"
    }
  ]
}
```

Figure 274. List Virtual Storage Resources of a Storage Group: Response

Get Virtual Storage Resource Properties

The Get Virtual Storage Resource Properties operation retrieves the properties of a single Virtual Storage Resource element object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group object and the URI variable *{virtual-storage-resource-id}* is the element ID of the virtual storage resource element.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Virtual Storage Resource object as defined in the “Data model” on page 432. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the virtual storage resource object as defined in “Virtual Storage Resource element object” on page 445.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or does not identify an FCP storage group, or if the element ID *{virtual-storage-resource-id}* does not identify a virtual storage resource in the storage group.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the “Response body contents” on page 498.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 229. Get Virtual Storage Resource Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with object-id <i>{storage-group-id}</i> does not exist on the HMC, or is a storage group of the wrong type , or the API user does not have object-access permission to it.
	5	A virtual storage resource with element-id <i>{virtual-storage-resource-id}</i> does not exist in the storage group on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
GET /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
4ce7feea-9fe9-11e8-b6db-fa163e3c2af4 HTTP/1.1
x-api-session: 63naxg5mxrmpugoyatmcwn6nv5zz5lbtb1zf6ga1408yhplxi
```

Figure 275. Get Virtual Storage Resource Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:51:56 GMT
content-type: application/json; charset=UTF-8
content-length: 690
{
  "adapter-port-uri": "/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
  "class": "virtual-storage-resource",
  "degraded-reasons": [
    "adapter"
  ],
  "description": "",
  "device-number": "0001",
  "element-id": "4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
  "element-uri": "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
virtual-storage-resources/4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
  "name": "vhba_FCP Storage Group1",
  "parent": "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
  "partition-uri": "/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4",
  "world-wide-port-name": "a1b2c3d4e5f60003",
  "world-wide-port-name-info": {
    "status": "validated",
    "world-wide-port-name": "a1b2c3d4e5f60003"
  }
}
```

Figure 276. Get Virtual Storage Resource Properties: Response

Update Virtual Storage Resource Properties

The Update Virtual Storage Resource Properties operation updates one or more of the writable properties of a virtual storage resource.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group object and the URI variable *{virtual-storage-resource-id}* is the element ID of the virtual storage resource element.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the “Virtual Storage Resource element object” on page 445. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a virtual storage resource's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access

permission, or does not identify an FCP storage group, or if the element ID *{virtual-storage-resource-id}* does not identify a Virtual Storage Resource element object in the storage group. In addition to object-access to the parent storage group, updates to the **adapter-port-uri** or **device-number** properties also require object-access permission to the partition that is associated with the target virtual storage resource. If these properties appear in the request body and the user does not have object-access to the partition identified in the **partition-uri** property, a 403 (Forbidden) status code is returned. If the CPC on which this virtual storage resource exists is not active, or if the **adapter-port-uri** field references a storage adapter port that does not reside in the target virtual storage resource's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines an **adapter-port-uri** field that references a non-FCP adapter, because the parent storage group already contains a virtual storage resource with the specified name, or because the partition associated with this virtual storage resource already contains a device with the supplied device number.

If the request body contents are valid, the virtual storage resource's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to the partition referenced by the virtual storage resource's **partition-uri** property. This requirement only applies when updating the **adapter-port-uri** or **device-number** properties.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 230. Update Virtual Storage Resource Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A virtual storage resource with the name specified in the request body already exists within the parent storage group.
	18	The request body contains the adapter-port-uri field, but the type property of the parent adapter is not "fcp" .
	452	The new value supplied in the device-number field conflicts with an existing device number for another device attached to the partition associated with this virtual storage resource.

Table 230. Update Virtual Storage Resource Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
	450	The adapter-port-uri or device-number fields are present in the request body and the API user does not have object-access permission to the partition referenced by the current value of the virtual storage resource's partition-uri property
404 (Not Found)	1	A storage group with object-id { <i>storage-group-id</i> } does not exist on the HMC, or is a storage group of the wrong type , or the API user does not have object-access permission to it.
	5	A virtual storage resource with element-id { <i>virtual-storage-resource-id</i> } does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions").
	2	The parent storage group with object-id { <i>storage-group-id</i> } or the partition identified by this virtual storage resource's partition-uri property value was busy and the request timed out.
	6	The status property value for the partition identified by this virtual storage resource's partition-uri property is not valid to perform the operation (must be in one of the following states: "active" , "degraded" , "paused" , "reservation-error" , "stopped" , or "terminated").
	441	The storage adapter port referenced by the adapter-port-uri field resides in a different CPC than the targeted virtual storage resource.
	498	The adapter-port-uri field was present in the request body when the current value of that property in the virtual storage resource element with object-id { <i>virtual-storage-resource-id</i> } is null .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
4ce7feea-9fe9-11e8-b6db-fa163e3c2af4 HTTP/1.1
x-api-session: 3f4ojo8469608kk498sr9e96l05yrsyapyheb1icwn355ws0pd
content-type: application/json
content-length: 44
{
  "description": "vHBA in FCP Storage Group"
}
```

Figure 277. Update Virtual Storage Resource Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:49:28 GMT

<No response body>
```

Figure 278. Update Virtual Storage Resource Properties: Response

Get Partitions for a Storage Group

The Get Partitions for a Storage Group operation lists the partitions to which the storage group with the given identifier is attached.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/operations/get-partitions
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters

Name	Type	Req/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned Partition objects to those that have a matching name property
status	String Enum	Optional	Optional filter string to limit returned Partition objects to those that have a matching status property. Value must be a valid partition status property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
partitions	Array of partition-info objects	Array of partition-info objects, described in the next table. The returned array may be empty.

Each nested partition-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The canonical URI path (object-uri) of the partition object.
name	String	The name property of the object.
status	String Enum	The status property of the object.

Description

This operation lists the partitions to which the identified storage group is attached. The object URI, name and status are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid value for the **status** property according to the partition data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have the specified **status** value. If the **status** parameter is omitted, this filtering is not done.

A partition is included in the list only if the user has object-access permission to it. If no partitions are to be included in the results due to filtering or lack of permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the “Response body contents” on page 502.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 231. Get Partitions for a Storage Group: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	A storage group with object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/get-partitions
HTTP/1.1
x-api-session: 40xd7zji2d7manim658hz4ot8r7x7fp3xuk0nyuced9psqubc8
```

Figure 279. Get Partitions for a Storage Group: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 15:53:46 GMT
content-type: application/json; charset=UTF-8
content-length: 235
{
  "partitions": [
    {
      "name": "Partition 2",
      "object-uri": "/api/partitions/8d1ca9a0-998d-11e8-8352-fa163e27d492",
      "status": "active"
    },
    {
      "name": "Partition 1",
      "object-uri": "/api/partitions/72a0f8b0-998d-11e8-8352-fa163e27d492",
      "status": "stopped"
    }
  ]
}

```

Figure 280. Get Partitions for a Storage Group: Response

Validate LUN Path

The Validate LUN Path operation verifies if a LUN is reachable along the path specified. The path is specified by the host world wide port name, adapter port, target world wide port name, and the LUN to indicate the connections from the CPC to the Storage Subsystem.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/validate-lun-path

In this request, the URI variable {cpc-id} is the object ID of the CPC that contains the host world wide port name and the adapters.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
host-world-wide-port-name	String (16)	Required	A 16-character lower-case hexadecimal string that contains a WWPN that uniquely identifies a host or initiator device in the SAN.
adapter-port-uri	String/URI	Required	The canonical URI path of the Storage Port element object to be used to validate the LUN configuration.
target-world-wide-port-name	String (16)	Required	A 16-character lower-case hexadecimal string that contains a WWPN that uniquely identifies a target port in one of the storage subsystems connected to this CPC.
logical-unit-number	String (1-16)	Required	The lower-case hexadecimal logical unit number (LUN) representing the storage device configured in the storage controller.

Description

This operation verifies if a LUN identified by **logical-unit-number** configured for the **host-world-wide-port-name** through the storage port designated by **adapter-port-uri** and the target port designated by **target-world-wide-port-name**.

The **adapter-port-uri** field in the request body must designate an existing Storage Port of an existing adapter, and the API user must have object-access permission to that adapter. If not, 404 (Not Found) status code is returned

If the **adapter-port-uri** does not designate an existing Storage Port of an existing adapter of **type "fcp"**, a 409 (Conflict) status code is returned.

If the storage controller does not configure the specified **target-world-wide-port-name** for the given combination of **host-world-wide-port-name** and **adapter-port-uri**, a 409 (Conflict) status is returned.

If the storage controller does not configure the specified **logical-unit-number** for the given combination of **host-world-wide-port-name**, **adapter-port-uri** and **target-world-wide-port-name**, a 409 (Conflict) status code is returned.

If the CPC does not have **"dpm-storage-management"** feature enabled, or designates a Storage Port that does not reside in the target CPC, a 409 (Conflict) status is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by {cpc-id}.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 232. Validate LUN Path: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	2	The storage port designated by the adapter-port-uri does not exist on the HMC or the API user does not have object-access permission for the adapter containing the storage port.
409 (Conflict)	13	The operation is not supported when the "dpm-storage-management" feature is not enabled on the CPC.
	441	The storage adapter port referenced by the adapter-port-uri field resides in a different CPC than the targeted storage group.
	483	The adapter designated by the adapter-port-uri is not of type "fcp" .
	484	A configuration for the target-world-wide-port-name does not exist in the storage controller for the given host-world-wide-port-name and adapter-port-uri combination.
	485	A configuration for the logical-unit-number does not exist in the storage controller for the given host-world-wide-port-name , adapter-port-uri and target-world-wide-port-name combination.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e/operations/validate-lun-path
Request Headers : {
  content-type:application/json,
  x-api-session:p5e86fr0btffsspy95sal06vkfp4cbkj4tjq27qmlr3fcx71dd
  content-length:233,
  {
    "adapter-port-uri":"/api/adapters/df4d482e-ce2c-11e7-bca6-00106f0d81cb/
      storage-ports/0",
    "host-world-wide-port-name":"C05076FFE8000002",
    "logical-unit-number":"4013402F00000000",
    "target-world-wide-port-name":"C05076FFE8001001"
  }
}
```

Figure 281. Validate LUN Path: Request

```
204 No Content
cache-control: no-cache,
date:Mon, 30 Jul 2018 17:32:40 GMT,
server:Hardware management console API web server / 2.0,
x-request-id:Sxb597d98c-8bef-11e8-8db3-00106f0d8409.1fca Rxc,
x-wsa-provider-duration-ms:934

<No response body>
```

Figure 282. Validate LUN Path: Response

Start FCP Storage Discovery

The Start FCP Storage Discovery operation triggers asynchronous storage discovery for an FCP storage group. The system will check the connections for the storage group and create a connection report indicating the status of those connections. Because the system periodically performs storage discovery for FCP storage groups and creates a connection report for those storage groups, it is not necessary to use this operation; however, issuing this operation may cause the discovery to be performed sooner than it would have otherwise.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/start-fcp-storage-discovery
```

In this request, the URI variable *{storage-group-id}* is the object ID of the target storage group.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
force-restart	Boolean	Optional	Indicates if there is an in-progress discovery operation for the specified storage group, it should be terminated and started again. If false or there is no in-progress discovery operation for the specified storage group, a new one is started. Default: false

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 508](#). The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The operation triggers Logical Unit Number (LUN) discovery for the target storage group. Once the LUN discovery operation is completed a connection report is generated for the storage group, indicating the status of the connections. Once the connection report is created, the `Get Connection Report` operation can be used to retrieve it. This operation is only valid for the storage groups of **type "fcp"**. The API user must have action/task permission to the **Configure Storage - System Programmer** task or the **Configure Storage - Storage Administrator** task; otherwise, status code 403 (Forbidden) is returned.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission or if the storage group identified by the object ID *{storage-group-id}* is not of **type "fcp"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or the **Configure Storage - Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 507](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 233. Start FCP Storage Discovery: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage - Storage Administrator tasks.
404 (Not Found)	1	A storage group with the object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group with object ID {storage-group-id} has a type value other than "fcp" .
409 (Conflict)	1	The state of the CPC to which the storage group is associated is not valid to perform the operations. It must be one of the following states: "active" , "service-required" , "degraded" , or "exceptions" .
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Table 234. Start FCP Storage Discovery: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	The operation completed successfully
404 (Not Found)	1	The storage group with object-id {storage-group-id} does not exist on the HMC or the API user does not have object-access permission for it.
500 (Server Error)	263	The storage discovery operation failed and there will be no automatic retry.

Example HTTP interaction

```
POST /api/storage-groups/2217a632-56bb-11e9-a434-fa163effe469/operations/
start-fcp-storage-discovery HTTP/1.1
x-api-session: 36lkftjan75oyr72g1bz9bic4r68qk2dtsm0svw2ei9tpa2az4
```

Figure 283. Start FCP Storage Discovery: Request

```

202 Accepted
server: Hardware management console API web server / 2.0
location:/api/jobs/31500aa4-56bb-11e9-ac0a-fa163efc2ef4
cache-control : no-cache
date: Thu, 04 Apr 2019 09:22:38 GMT
content-type:application/json;charset=UTF-8
content-length:60
{
  "job-uri":"/api/jobs/31500aa4-56bb-11e9-ac0a-fa163efc2ef4"
}

```

Figure 284. Start FCP Storage Discovery: Response

Usage notes

- At any point in time, there will be at most one in-progress discovery operation checking connections for a given storage group. While a discovery operation is in progress, any new request for the same storage group from the APIs or the Graphical User Interface will be linked to that in-progress discovery operation. An in-progress discovery operation can be interrupted for various reasons, such as starting a partition to which this storage group is attached, reassigning adapters of the virtual resources of the storage group, or invoking the Start FCP Storage Discovery operation with a **force-restart** flag set to **true**. If an in-progress discovery operation is interrupted for any reason, the operation will be automatically retried and the previous requests associated with that in-progress discovery operation will be linked to the new operation.
- If the request fails for any reason, all requests associated with that in-progress request will fail and an error code will be returned in the asynchronous result.

Get Connection Report

The Get Connection Report operation retrieves the most recent connection report for a storage group. The connection report for a storage group of **type "fc"** is first generated when at least one of the volumes is mapped and the storage group is attached to at least one partition. The connection report for a storage group of **type "fcp"** is first generated when a storage discovery operation completes successfully for the first time.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/operations/get-connection-report
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
last-scan-time	Timestamp	The time the last discovery operation for a storage group completed successfully. This field is only present when the storage group's type property value is "fcp" .
storage-group-uri	String/ URI	The canonical URI path of the storage group with which the connection report is associated.

Field name	Type	Description
fcp-fabrics	Array of fcp-fabric-info objects	The list of information about the distinct storage fabrics that the storage adapters in the system are connected. Each element in the array is an instance of an fcp-fabric-info object, defined in Table 235 on page 510 . This field is only present when the storage group's type property is "fcp" .
fcp-storage-subsystems	Array of fcp-storage-subsystem-info objects	The list of information about the distinct storage subsystems, where the FCP volumes for this storage group are defined. Each element in the array is an instance of fcp-storage-subsystem-info object, defined in Table 238 on page 511 . This field is only present when the storage group's type property is "fcp" .
ficon-storage-subsystems	Array of ficon-storage-subsystem-info objects	The list of information about the distinct storage subsystems, where the FICON volumes for this storage group are defined. Each element in the array is an instance of a ficon-storage-subsystem-info object, defined in Table 240 on page 512 . This field is only present when the storage group's type property is "fc" .

An fcp-fabric-info object contains information about a single fabric configured in the storage area network switches. Each fabric-info object contains the following fields:

Table 235. fcp-fabric-info nested object		
Name	Type	Description
fabric-id	String (16)	The World Wide Name (WWN) of the uplink Fibre Channel Switch.
expected-adapter-count	Integer	The number of adapters expected to be zoned for the World Wide Port Names (WWPNs) of the storage group in this fabric.
accessible-subsystem-ids	Array of String	The list of World Wide Node Names (WWNNs) of the storage subsystems that are accessible for the adapters in this fabric. Each WWNN is a 16-character lowercase hexadecimal string representing the node name defined in the storage subsystem.
world-wide-port-names	Array of world-wide-port-name-zone-info objects	The list of information about the World Wide Port Names (WWPNs) that have been allocated to support this FCP storage group. Each element in the array is an instance of world-wide-port-name-zone-info object, defined in Table 236 on page 511 .
unzoned-adapters	Array of adapter-info objects	The list of information about storage adapters in the system that are connected to this fabric but does not contain a zoning configuration or any of the WWPNs in this storage group. Each element in the array is an instance of adapter-info object, defined in Table 237 on page 511 .

A world-wide-port-name-zone-info object contains information about the World Wide Port Names (WWPNs) of the storage group and the set of adapters that are zoned for it in the fabric which can be used to connect to the storage network. Each world-wide-port-name-zone-info object contains the following fields:

Table 236. world-wide-port-name-zone-info nested object		
Name	Type	Description
world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains the world wide port name used to access the FCP storage volumes defined for the storage group.
zoned-adapters	Array of adapter-info objects	The list of information about storage adapters that are configured in one of the active zones of the storage fabric. Each element in the array is an instance of the adapter-info object, defined in Table 237 on page 511. The world wide port name can be assigned to the virtual storage resource created on the adapter to access the FCP storage volumes defined for the storage group. The sum of the number of distinct adapters zoned for this world wide port name across various fabrics should be greater than or equal to the value of the connectivity field of the storage group.

An adapter-info object contains information about an adapter. Each adapter-info object contains the following fields:

Table 237. adapter-info nested object		
Name	Type	Description
adapter-name	String (1-64)	The display name specified for this adapter
adapter-status	String Enum	The status of the adapter. The adapter can be used to check the connections only when the status is "active".
adapter-uri	String/ URI	The canonical URI path of the Adapter object.

An fcp-storage-subsystem-info object contains information about a storage subsystem that the FCP volumes of the storage group are defined in. Each fcp-storage-subsystem-info object contains the following fields:

Table 238. fcp-storage-subsystem-info nested object		
Name	Type	Description
world-wide-node-names	Array of string	The list of World Wide Node Names (WWNNs) defined for the storage subsystem. Each WWNN is a 16-character lowercase hexadecimal string representing the node name defined in the storage subsystem.
world-wide-port-names-count	Integer	The number of World Wide Port Names (WWPNs) that are zoned and masked correctly in the storage switch and storage subsystem, to access the expected number of FCP volumes for the storage group.
expected-volumes-count	Integer	The number of FCP volumes expected to be discovered in this storage subsystem.

Table 238. fcp-storage-subsystem-info nested object (continued)

Name	Type	Description
storage-configurations	Array of storage-configuration-info objects	The list of information about subsystem configurations for each world wide port name of the storage group. Each element in the array is an instance of storage-configuration-info object, defined in Table 239 on page 512.

A storage-configuration-info object contains information about the subsystem configuration for each WWPN of the storage group. Each storage-configuration-info object contains the following fields:

Table 239. storage-configuration-info nested object

Name	Type	Description
world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains the world wide port name used to access the FCP storage volumes defined for the storage group.
discovered-volumes-count	Integer	The number of volumes the WWPN is configured to access in the storage subsystem.
volumes-configuration-status	String Enum	The configuration status of the WWPN across the subsystems. Indicates if the WWPN is able to access the expected number of volumes in the storage subsystem. Possible values: • "no-volumes" - The WWPN cannot access any volume across the subsystems. • "too-many-volumes" - The WWPN can access more volumes than what was expected for the storage group. • "too-few-volumes" - The WWPN can access fewer volumes than what was expected for the storage group. • "different-volumes" - The WWPN can access one or more volumes not defined to the storage group, or it cannot access one or more of the volumes that are defined to the storage group, but not all. • "correct-volumes" - The WWPN can access all of the defined volumes of the storage group and no other volumes.

A ficon-storage-subsystem-info object contains information about a storage subsystem to which the FICON volumes of the storage group are mapped. Each ficon-storage-subsystem-info object contains the following fields:

Table 240. ficon-storage-subsystem-info nested object

Name	Type	Description
mapped-subsystem-uri	String/ URI	The canonical URI path of the Storage Subsystem object that contains control units mapped to the volumes of the storage group.
mapped-control-unit-uris	Array of String/ URI	The list of storage control units within the storage subsystem that are mapped to the volumes in the storage group. Each element in this array is the canonical URI path of a Storage Control Unit object.

Description

This operation retrieves the most recent connection report of the specified storage group. The API user must have action/task permission to the **Configure Storage – System Programmer** task or the **Configure Storage – Storage Administrator** task; otherwise, status code 403 (Forbidden) is returned.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or the **Configure Storage - Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 509.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 241. Get Connection Report: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer or Configure Storage - Storage Administrator tasks.
404 (Not Found)	1	A storage group with the object-id <i>{storage-group-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group with object ID <i>{storage-group-id}</i> has a type value other than “fcp”.
409 (Conflict)	250	A connection report has not been generated.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
GET /api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/operations/  
get-connection-report HTTP/1.1  
x-api-session: 1wianh609dwilk4lfgyriv8ywx4o2okf4d19kn3jln1p9d1msw
```

Figure 285. Get Connection Report: Request


```

"expected-adapter-count":0,
"fabric-id":"100099901",
"unzoned-adapters":[],
"world-wide-port-names":[
{
"world-wide-port-name":"a1b2c3d4e5f60045",
"zoned-adapters":[
{
"adapter-name":"FCP 0101 Z01B-02",
"adapter-status":"active",
"adapter-uri":"/api/adapters/45b2e5a6-448d-11e9-9536-fa163effe469"
}
]
}
],
},
{
"accessible-subsystem-ids":[
"5005076400c00003",
"5005076400c00002",
"5005076400c00004",
"5005076400c00001",
"5005076400c00000"
],
},
{
"expected-adapter-count":0,
"fabric-id":"100099905",
"unzoned-adapters":[],
"world-wide-port-names":[
{
"world-wide-port-name":"a1b2c3d4e5f60045",
"zoned-adapters":[
{
"adapter-name":"FCP 0130 Z01B-17",
"adapter-status":"active",
"adapter-uri":"/api/adapters/4aaaf260-448d-11e9-9536-fa163effe469"
}
]
}
],
},
{
"accessible-subsystem-ids":[
"5005076400c00003",
"5005076400c00002",
"5005076400c00004",
"5005076400c00001",
"5005076400c00000"
],
},

```

Figure 287. Get Connection Report: Response (Part 2)

```

    "expected-adapter-count":0,
    "fabric-id":"100099904",
    "unzoned-adapters":[],
    "world-wide-port-names":[
      {
        "world-wide-port-name":"a1b2c3d4e5f60045",
        "zoned-adapters":[
          {
            "adapter-name":"FCP 0105 Z01B-03",
            "adapter-status":"active",
            "adapter-uri":"/api/adapters/a5a6bb22-448d-11e9-ad2d-fa163effe469"
          }
        ]
      }
    ],
    "accessible-subsystem-ids":[
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
    ],
    "expected-adapter-count":0,
    "fabric-id":"100099903",
    "unzoned-adapters":[],
    "world-wide-port-names":[
      {
        "world-wide-port-name":"a1b2c3d4e5f60045",
        "zoned-adapters":[
          {
            "adapter-name":"FCP 0100 Z01B-02",
            "adapter-status":"active",
            "adapter-uri":"/api/adapters/3ee50ee8-448d-11e9-9536-fa163effe469"
          }
        ]
      }
    ],
    "accessible-subsystem-ids":[
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
    ],
  ],

```

Figure 288. Get Connection Report: Response (Part 3)

```

    "expected-adapter-count":0,
    "fabric-id":"100099902",
    "unzoned-adapters":[],
    "world-wide-port-names":[
      {
        "world-wide-port-name":"a1b2c3d4e5f60045",
        "zoned-adapters":[
          {
            "adapter-name":"FCP 0104 Z01B-03",
            "adapter-status":"active",
            "adapter-uri":"/api/adapters/a317e23c-448d-11e9-ad2d-fa163effe469"
          }
        ]
      }
    ],
  },
  "fcp-storage-subsystems":[
    {
      "expected-volumes-count":1,
      "storage-configurations":[
        {
          "discovered-volumes-count":5,
          "volumes-configuration-status":"too-many-volumes",
          "world-wide-port-name":"a1b2c3d4e5f60045"
        }
      ],
      "world-wide-node-names":[
        "5005076400c00003",
        "5005076400c00002",
        "5005076400c00004",
        "5005076400c00001",
        "5005076400c00000"
      ],
      "world-wide-port-names-count":1
    }
  ],
  "last-scan-time":1554370171000,
  "storage-group-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469"
}

```

Figure 289. Get Connection Report: Response (Part 4)

Usage notes

- If a connection report has not been generated for a storage group of **type "fcp"**, use the Start FCP Storage Discovery operation to initiate the creation of the connection report.
- If a connection report has not been generated for a storage of **type "fc"**, use the Fulfill FICON Storage Volume operation to map the ECKD storage resources to the volumes that are part of the storage group and the Attach Storage Group to Partition operation to attach the storage group to a partition.

Get Storage Group Histories

The Get Storage Group Histories operation returns a chronological list of actions performed on the storage groups known to the target Console. The response can be filtered according to query parameters, if specified.

HTTP method and URI

GET /api/console/operations/get-storage-group-histories

Query parameters:

Name	Type	Rqd/Optional	Description
begin-time	Timestamp	Optional	A timestamp to filter the entries returned in the storage-group-actions list. Actions with action-time values earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
end-time	Timestamp	Optional	A timestamp to filter the entries returned in the storage-group-actions list. Actions with action-time values later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
cpc-uri	String/URI	Optional	The canonical URI path of a CPC object to limit the returned entries to those that have a matching cpc-uri field. If not specified, then no such filtering is performed.
storage-group-uri	String/URI	Optional	The canonical URI path of a Storage Group object to limit the returned entries to those that have a matching storage-group-uri field. If not specified, then no such filtering is performed.
user-name	String (4-320)	Optional	Filter pattern (regular expression) to limit the entries returned in the storage-group-actions list to those that have a matching user-name field. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
storage-group-histories	Array of storage-group-history-info objects	The list of information about the history of actions performed on each storage group. Each element in the array is an instance of storage-group-history-info object, described in the next table. The returned array may be empty.

Each storage-group-history-info object contains the following fields:

Table 242. storage-group-history-info nested object		
Name	Type	Description of specialization
cpc-uri	String/URI	The canonical URI path of the CPC object with which the storage group is associated.
storage-group-uri	String/URI	The canonical URI path of the Storage Group object with which the storage group history is associated.
storage-group-actions	Array of storage-group-action-info objects	The list of information about the actions performed on the storage group. Each element in the array is an instance of a storage-group-action-info object, described in the next table.

Each storage-group-action-info object contains the following fields:

Table 243. *storage-group-action-info* nested object

Name	Type	Description
user-name	String (4-320)	The name of the user who performed the action on the storage group. Refer to the name property of the User object, defined in Table 325 on page 630 The value is " no user " if the action was performed by the system.
action-time	Timestamp	The time when the action was performed on the storage group.

Table 243. storage-group-action-info nested object (continued)

Name	Type	Description
action	String Enum	The action that was performed on the storage group. Values: • "creation-requested" - Storage group was created and fulfillment is pending. • "modification-requested" - One or more properties of the storage group were modified. Some properties are changed immediately and some are pending fulfillment which is indicated by the field storage-group-fulfillment-state. • "deletion-requested" - Storage group was deleted. • "request-fulfilled" - Storage group request was fulfilled. • "attached-to-partition" - Storage group was attached to a partition. • "detached-from-partition" - Storage group was detached from a partition. • "auto-created" - Storage group was created automatically when the CPC was upgraded to a level where "dpm-storage-management" feature is enabled. • "auto-created-during-migration" - Storage group was created as part of MES upgrade. • "mismatched-volumes-found" - Volumes not matching the storage request for a storage group of type "fcp" were discovered. • "mismatched-volumes-accepted" - Mismatched volumes discovered were accepted by the user. • "mismatched-volumes-rejected" - Mismatched volumes discovered were rejected by the user. • "send-email-failed" - Send email operation failed during creation/modification/deletion of the storage group. • "manually-fulfilled" - A storage volume was manually fulfilled. • "incomplete" - Fulfillment state of storage group changed to Incomplete. • "incomplete-resolved" - Fulfillment state of storage group changed from Incomplete. • "backing-adapter-changed" – Backing adapter of a virtual storage resource belonging to the storage group was changed. • "partition-name-changed" – The name of a Partition to which this storage group is attached was changed. • "device-number-changed" - Device number of virtual storage resource or a storage volume of type "fc" belonging to the storage group was changed. • "ficon-volumes-mapped" - Storage volumes of type "fc" were mapped to control units.

Table 243. *storage-group-action-info* nested object (continued)

Name	Type	Description
storage-group-fulfillment-state	String Enum	The fulfillment state of the storage group, after the action was performed. Values: - Any value of the fulfillment-state property as defined in the Storage Group Data model in Table 202 on page 432. "deleted" - The storage group has been deleted.
storage-group-configuration	storage-group-configuration object	A nested object that provides additional information about the configuration of the storage group that was changed as part of this action. The value is a single instance of a storage-group-configuration object, defined in Table 244 on page 521. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the value of the action property is one of the following: "creation-requested" , "modification-requested" , "deletion-requested" , "backing-adapter-changed" , "mismatched-volumes-found" , "mismatched-volumes-rejected" , or "auto-created-during-migration" .
partition-uri	String/URI	The canonical URI path of the partition which this storage group is attached to or detached from. This field is only present if the action property is either "attached-to-partition" , "detached-from-partition" , or "partition-name-changed" .
old-partition-name	String (1-64)	The old name of the partition to which the storage group is attached. This field is only present if the action property is "partition-name-changed" .
new-partition-name	String (1-64)	The new name of the partition to which the storage group is attached. This field is only present if the action property is "partition-name-changed" .

Each storage-group-configuration object contains the following fields:

Table 244. *storage-group-configuration* nested object

Name	Type	Description
old-name	String (1-64)	The old value for the name property of the storage group.
new-name	String (1-64)	The new value for the name property of the storage group.
old-description	String (0-200)	The old value for the description property of the storage group.
new-description	String (0-200)	The new value for the description property of the storage group.
old-shared	Boolean	The old value for the shared property of the storage group.
new-shared	Boolean	The new value for the shared property of the storage group.

Table 244. storage-group-configuration nested object (continued)

Name	Type	Description
old-connectivity	Integer	The old value for the connectivity property of the storage group.
new-connectivity	Integer	The new value for the connectivity property of the storage group.
old-max-partitions	Integer	The old value for the max-partitions property of the storage group.
new-max-partitions	Integer	The new value for the max-partitions property of the storage group.
old-direct-connection-count	Integer (0-1000)	The old value for the direct-connection-count property of the storage group.
new-direct-connection-count	Integer (0-1000)	The new value for the direct-connection-count property of the storage group.
world-wide-port-names-added	Array of world-wide-port-name-info objects	The list of information about the World Wide Port Names (WWPNs) that have been newly allocated to this storage group. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 206 on page 447. This field is only present when the action resulted in new World Wide Port Names being added for this storage group.
world-wide-port-names-deleted	Array of world-wide-port-name-info objects.	The list of information about the World Wide Port Names (WWPNs) that have been deleted from this storage group. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 206 on page 447. This field is only present when the action resulted in World Wide Port Names being deleted from this storage group.
storage-volumes-added	Array of storage-volume-info objects	The list of information about the storage volumes that were newly added to this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 245 on page 523. This field is only present when the action resulted in new volumes being added to the storage group. When the action is "mismatched-volumes-found", this field contains the volumes that were sensed but did not match the storage request.
storage-volumes-modified	Array of storage-volume-info objects	The list of information about the storage volumes that were modified in this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 245 on page 523. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the action resulted in volumes of this storage group being modified. When the action is "mismatched-volumes-accepted", this field contains the volumes that were accepted by the Accept FCP Storage Volumes operation.

Table 244. storage-group-configuration nested object (continued)

Name	Type	Description
storage-volumes-deleted	Array of storage-volume-info objects	The list of information about the storage volumes that were deleted from this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 245 on page 523. This field is only present when the action resulted in volumes being deleted from this storage group. When the action is "mismatched-volumes rejected", this field contains the volumes that were rejected by the Reject Mismatched FCP Storage Volumes operation.
virtual-storage-resources-added	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were added. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 246 on page 525. This field is only present when the action property is either "attached-to-partition" or "modification-requested" and action resulted in virtual storage resources of this storage group being added.
virtual-storage-resources-modified	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were modified. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 246 on page 525. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the action property is either "device-number-changed", "backing-adapter-changed", or "modification-requested" and action resulted in virtual storage resources of this storage group being modified.
virtual-storage-resources-deleted	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were deleted. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 246 on page 525. This field is only present when the action property is either "detached-from-partition" or "modification-requested" and action resulted in virtual storage resources of this storage group being deleted.

Each storage-volume-info object contains the following fields:

Table 245. storage-volume-info nested object

Name	Type	Description
element-uri	String/URI	The canonical URI path for the storage volume element object that was modified in this action.
old-name	String (1-64)	The old value for the name property of the storage volume.
new-name	String (1-64)	The new value for the name property of the storage volume.

Table 245. storage-volume-info nested object (continued)

Name	Type	Description
old-description	String (0-100)	The old value for the description property of the storage volume.
new-description	String (0-100)	The new value for the description property of the storage volume.
fulfillment-state	String Enum	The value of the fulfillment-state property of the storage volume at the time of this action. This field is always present.
eckd-type	String Enum	The value of the eckd-type property of the storage volume at the time of this action. This field is always present when the storage group's type property is "fc" .
uuid	String (16, 32)	The value of the uuid property of the storage volume at the time of this action. This field is only present when the storage group's type property is "fcp" and the volume has already been sensed.
old-size	Float	The old value for the size property of the storage volume.
new-size	Float	The new value for the size property of the storage volume.
old-usage	String Enum	The old value for the usage property of the storage volume.
new-usage	String Enum	The new value for the usage property of the storage volume.
old-device-number	String (4)	The old value for the device-number property of the storage volume.
new-device-number	String (4)	The new value for the device-number property of the storage volume.
control-unit-uri	String/URI	The value of the control-unit-uri property of the storage volume at the time of this action. This field is always present when the storage group's type property is "fc" . The value will be null if a "base" volume was not yet fulfilled at the time of this action.
unit-address	String (2)	The value of the unit-address property of the storage volume at the time of this action. This field is always present when the storage group's type property is "fc" . The value will be null if a "base" volume was not yet fulfilled at the time of this action.

Each virtual-storage-resource-info object contains the following fields:

Table 246. *virtual-storage-resource-info* nested object

Name	Type	Description
element-uri	String/URI	The canonical URI path for the Virtual Storage Resource element object that was modified in this action.
old-name	String (1-64)	The old value for the name property of the virtual storage resource.
new-name	String (1-64)	The new value for the name property of the virtual storage resource.
old-description	String (0-1024)	The old value for the description property of the virtual storage resource.
new-description	String (0-1024)	The new value for the description property of the virtual storage resource.
partition-uri	String/URI	The value of the partition-uri property of the Virtual Storage Resource element object at the time of this action. This field is always present.
old-device-number	String (4)	The old value for the device-number property of the virtual storage resource.
new-device-number	String (4)	The new value for the device-number property of the virtual storage resource.
old-adapter-port-uri	String/URI	The old value for the adapter-port-uri property of the virtual storage resource.
new-adapter-port-uri	String/URI	The new value for the adapter-port-uri property of the virtual storage resource.
world-wide-port-name	String (16)	The value of the world-wide-port-name property of the Virtual Storage Resource element object at the time of this action. This field is always present.

Description

This operation returns information about the history of actions performed for all the storage groups known to this console. It contains a chronological list of actions performed on storage groups. Information about storage groups to which the API user does not have object-access permission, are not included in the response. The history entry for each storage group contains an array of storage-group-action-info objects representing the actions performed on the storage group.

The order in which the storage-group-history-info objects are returned is not guaranteed and can change for every request.

A storage group's history entries are not deleted when the storage group object is deleted. The history entries are preserved for 30 days from the deletion of the storage group object, after which they are deleted automatically. This allows API clients to query minimal configuration information for the storage group even after they receive notification that the storage group has been deleted. API clients can use this configuration information to undo any configuration done earlier on the SAN in order to fulfill this storage group.

The entries can be limited by specifying explicit filtering criteria on the request. Filtering can occur on two different levels. The **storage-group-uri** and **cpc-uri** query parameters limit the results to a single storage group, or to all storage groups associated with a specific CPC respectively. The **begin-time**, **end-time** and **user-name** query parameters filter the actions that are returned for each storage group.

If the **begin-time** query parameter is specified, then any actions earlier than that time are omitted. If the **end-time** query parameter is specified, then any actions later than that time are omitted.

If the **storage-group-uri** query parameter is specified for the request, the returned list is limited to the storage group history that has a matching **storage-group-uri** value.

If the **cpc-uri** query parameter is specified for the request, the returned list is limited to the storage group history entries that have a matching **cpc-uri** value.

If the **user-name** query parameter is specified for the request, only actions with a **user-name** property matching the specified filter pattern are returned.

If no storage group history entries are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Storage Group object designated by the **storage-group-uri** property of each storage-group-history-info object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 518](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 247. Get Storage Group Histories: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter contains an invalid value.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/get-storage-group-histories?storage-group-  
name=SG_CFI_con_22 HTTP/1.1  
x-api-session: 2x828y1ic1qtyenkg33luihglj9qlnyxdilblc2q14y5bus44s
```

Figure 290. Get Storage Group Histories: Request

```

200 OK
server: Hardware management console API web server/2.0
cache-control: no-cache
date: Thu, 04 Apr 2019 09:41:54 GMT
content-type: application/json; charset=UTF-8
content-length: 5473
{
  "storage-group-histories": [
    {
      "cpc-uri": "/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
      "storage-group-actions": [
        {
          "action": "creation-requested",
          "action-time": 1554370624684,
          "storage-group-configuration": {
            "new-connectivity": 1,
            "new-description": "",
            "new-direct-connection-count": 0,
            "new-max-partitions": 1,
            "new-name": "SG_CFI_con_22",
            "new-shared": false,
            "storage-volumes-added": [
              {
                "element-uri": "/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469/storage-volumes/30bc6360-56bd-11e9-a59c-fa163effe469",
                "fulfillment-state": "pending",
                "new-description": "",
                "new-name": "vol1",
                "new-size": 10.0,
                "new-usage": "boot",
                "sensed-uuid": null
              }
            ],
            "world-wide-port-names-added": [
              {
                "status": "not-validated",
                "world-wide-port-name": "a1b2c3d4e5f60046"
              }
            ]
          },
          "storage-group-fulfillment-state": "pending",
          "user-name": "sysprog"
        }
      ],
    }
  ],
}

```

Figure 291. Get Storage Group Histories: Response (Part 1)

```

{
  "action": "deletion-requested",
  "action-time": 1554370628053,
  "storage-group-configuration": {
    "old-connectivity": 1,
    "old-description": "",
    "old-direct-connection-count": 0,
    "old-max-partitions": 1,
    "old-name": "SG_CFI_con_22",
    "old-shared": false,
    "storage-volumes-deleted": [
      {
        "element-uri": "/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469/storage-volumes/30bc6360-56bd-11e9-a59c-fa163effe469",
        "fulfillment-state": "pending",
        "old-description": "",
        "old-name": "vol1",
        "old-size": 10.0,
        "old-usage": "boot",
        "sensed-uuid": null
      }
    ],
    "world-wide-port-names-deleted": [
      {
        "status": "not-validated",
        "world-wide-port-name": "a1b2c3d4e5f60046"
      }
    ]
  },
  "storage-group-fulfillment-state": "deleted",
  "user-name": "sysprog"
},
{
  "storage-group-name": "SG_CFI_con_22",
  "storage-group-type": "fcp",
  "storage-group-uri": "/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469"
},
{
  "cpc-uri": "/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
  "storage-group-actions": [
    {
      "action": "creation-requested",
      "action-time": 1554370175411,
      "storage-group-configuration": {
        "new-connectivity": 1,
        "new-description": "",
        "new-direct-connection-count": 0,
        "new-max-partitions": 1,
        "new-name": "SG_CFI_con_22",
        "new-shared": false,
        "storage-volumes-added": [

```

Figure 292. Get Storage Group Histories: Response (Part 2)

```

    {
      "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
        24f274b2-56bc-11e9-bcc3-fa163effe469",
      "fulfillment-state": "pending",
      "new-description": "",
      "new-name": "vol1",
      "new-size": 10.0,
      "new-usage": "boot",
      "sensed-uuid": null
    }
  ],
  "world-wide-port-names-added": [
    {
      "status": "not-validated",
      "world-wide-port-name": "a1b2c3d4e5f60045"
    }
  ]
},
"storage-group-fulfillment-state": "pending",
"user-name": "sysprog"
},
{
  "action": "mismatched-volumes-found",
  "action-time": 1554370171053,
  "storage-group-fulfillment-state": "pending-with-mismatches",
  "user-name": "no user"
},
{
  "action": "deletion-requested",
  "action-time": 1554370194985,
  "storage-group-configuration": {
    "old-connectivity": 1,
    "old-description": "",
    "old-direct-connection-count": 0,
    "old-max-partitions": 1,
    "old-name": "SG_CFI_con_22",
    "old-shared": false,
    "storage-volumes-deleted": [
      {
        "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
          26225f50-56bc-11e9-8d03-fa163effe469",
        "fulfillment-state": "overprovisioned",
        "old-description": "",
        "old-name": "Data 48",
        "old-size": 0.0,

```

Figure 293. Get Storage Group Histories: Response (Part 3)

```

    "old-usage": "data",
    "sensed-uuid": "6005076307FFC6A60000000000000001"
  },
  {
    "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/26182224-56bc-11e9-8d03-fa163effe469",
    "fulfillment-state": "overprovisioned",
    "old-description": "",
    "old-name": "Data 47",
    "old-size": 0.0,
    "old-usage": "data",
    "sensed-uuid": "6005076307FFC6A60000000000000002"
  },
  {
    "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/24f274b2-56bc-11e9-bcc3-fa163effe469",
    "fulfillment-state": "pending-with-mismatches",
    "old-description": "",
    "old-name": "vol1",
    "old-size": 10.0,
    "old-usage": "boot",
    "sensed-uuid": "6005076307FFC6A60000000000000004"
  },
  {
    "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/262c85c0-56bc-11e9-8d03-fa163effe469",
    "fulfillment-state": "overprovisioned",
    "old-description": "",
    "old-name": "Data 49",
    "old-size": 0.0,
    "old-usage": "data",
    "sensed-uuid": "6005076307FFC6A60000000000000000"
  },
  {
    "element-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/260cd37e-56bc-11e9-8d03-fa163effe469",
    "fulfillment-state": "overprovisioned",
    "old-description": "",
    "old-name": "Data",
    "old-size": 0.0,

```

Figure 294. Get Storage Group Histories: Response (Part 4)


```

        "old-usage": "data",
        "sensed-uuid": "6005076307FFC6A60000000000000003"
    },
    "world-wide-port-names-deleted": [
        {
            "status": "validated",
            "world-wide-port-name": "a1b2c3d4e5f60045"
        }
    ],
    "storage-group-fulfillment-state": "deleted",
    "user-name": "sysprog"
},
{
    "storage-group-name": "SG_CFI_con_22",
    "storage-group-type": "fcp",
    "storage-group-uri": "/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469"
},
{
    "cpc-uri": "/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
    "storage-group-actions": [
        {
            "action": "creation-requested",
            "action-time": 1554370914499,
            "storage-group-configuration": {
                "new-connectivity": 1,
                "new-description": "",
                "new-direct-connection-count": 0,
                "new-max-partitions": 1,
                "new-name": "SG_CFI_con_22",
                "new-shared": false,
                "storage-volumes-added": [
                    {
                        "element-uri": "/api/storage-groups/dd29557c-56bd-11e9-9081-fa163effe469/storage-volumes/dd7851ea-56bd-11e9-9081-fa163effe469",
                        "fulfillment-state": "pending",
                        "new-description": "",
                        "new-name": "vol1",
                        "new-size": 10.0,
                        "new-usage": "boot",
                        "sensed-uuid": null
                    }
                ]
            },
            "world-wide-port-names-added": [
                {
                    "status": "not-validated",
                    "world-wide-port-name": "a1b2c3d4e5f60047"
                }
            ]
        },
        {
            "storage-group-fulfillment-state": "pending",
            "user-name": "sysprog"
        }
    ],
    "storage-group-name": "SG_CFI_con_22",
    "storage-group-type": "fcp",
    "storage-group-uri": "/api/storage-groups/dd29557c-56bd-11e9-9081-fa163effe469"
}
]
}

```

Figure 295. Get Storage Group Histories: Response (Part 5)

Inventory service data

Information about the Storage Groups managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage group objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "storage-group" are to be included. Information for a particular storage group is included only if the API user has object-access permission to that object.

For each storage group to be included, the inventory response array includes the following:

- An array entry for the storage group object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Group Properties”](#) on page 459. That is, the data provided is the same as would be provided if a `Get Storage Group Properties` operation were requested targeting this object.
- An array entry for each storage volume element associated with the storage group. For each such storage volume, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Volume Properties”](#) on page 479.
- An array entry for each virtual storage resource element associated with the storage group. For each such virtual storage resource, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for [“Get Virtual Storage Resource Properties”](#) on page 498.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage group. These objects would appear as multiple array entries in the response array:

```

{
  "active-connectivity":null,
  "active-max-partitions":null,
  "candidate-adapter-port-uris":[
    "/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
    "/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0"
  ],
  "class":"storage-group",
  "connectivity":2,
  "cpc-uri":"/api/cpcs/3aadfe48-128c-3766-8f51-37738f784591",
  "description":"",
  "direct-connection-count":0,
  "fulfillment-state":"pending",
  "max-partitions":2,
  "name":"FCP Group",
  "object-id":"2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
  "object-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
  "parent":"/api/console",
  "shared":true,
  "storage-volume-uris":[
    "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
    "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/2d71eeae-9fe9-11e8-b163-fa163e3c2af4"
  ],
  "type":"fcp",
  "unassigned-world-wide-port-names":[
    {
      "status":"not-validated",
      "world-wide-port-name":"a1b2c3d4e5f60006"
    },
    {
      "status":"not-validated",
      "world-wide-port-name":"a1b2c3d4e5f60005"
    }
  ],
  "virtual-storage-resource-uris":[
    "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
    "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/4ce7feea-9fe9-11e8-b6db-fa163e3c2af4"
  ]
},
{
  "active-size":null,
  "class":"storage-volume",
  "description":"",
  "element-id":"2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
  "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
  "fulfillment-state":"pending",
  "name":"1.00 GiB Boot",
  "parent-storage-group-uris":[
    "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4"
  ],
  "size":1.0,
  "usage":"boot",
  "uuid":"a1b2c3d4e5f6000325d4677ab2320e54"
},

```

Figure 296. Storage Group object: Sample inventory data - Response (Part 1)

```

{
  "adapter-port-uri": "/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
  "class": "virtual-storage-resource",
  "description": "",
  "device-number": "0000",
  "element-id": "4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
  "element-uri": "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
  "name": "vhba_FCP_Group0",
  "parent": "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
  "partition-uri": "/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4",
  "world-wide-port-name": "a1b2c3d4e5f60002"
  "world-wide-port-name-info": {
    "status": "not-validated",
    "world-wide-port-name": "a1b2c3d4e5f60002"
  }
}

```

Figure 297. Storage Group object: Sample inventory data - Response (Part 2)

Storage Template object

A Storage Template object represents a single storage template associated with a DPM-enabled CPC. A storage template can be used when creating a storage group to initialize the group's property values to the current values of the template's properties. See [Create Storage Group](#). The Storage Template object APIs provide access to the set of storage templates that are associated with a CPC that is enabled for DPM. APIs exist to create and delete storage templates, query storage templates, and modify selected properties of storage templates. APIs also exist to list and query properties of the storage template volume elements of storage templates.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 70, with the class-specific specializations identified in Table 249 on page 535 . The Storage Template object does not support the operational status related properties.

Table 248. Storage Template object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Template object is of the form <code>/api/storage-templates/{storage-template-id}</code> where <code>{storage-template-id}</code> is the value of the object-id property of the Storage Template object.
object-id	—	String (36)	The unique identifier for the storage template instance.
parent	—	String/ URI	The parent of a storage template is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (16)	The class of a Storage Template object is "storage-template" .
name	(w)(pc)	String (1-64)	The display name specified for the storage template. The character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage templates associated to the same CPC.

Table 248. Storage Template object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the storage template. This value will be used to initialize the description property of storage groups that are created using this storage template. Default value: An empty string.

Class specific additional properties

In addition to the properties defined via included schema, this object includes the following additional class-specific properties:

Table 249. Storage Template object: class specific properties

Name	Qualifier	Type	Description	Supported "type" values
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this Storage Template object. This value will be used to initialize the cpc-uri property of the storage groups that are created using this storage template.	All
type	—	String Enum	The type of storage resources managed by the storage template. Values: • "fcp" - Fibre Channel Protocol • "fc" - Fibre Connection This value will be used to initialize the type property of storage groups that are created using this storage template.	All
shared	(w)(pc)	Boolean	true if storage groups created from this template can be attached to more than one partition; false if the storage group is dedicated to a single partition. This property's value must be "false" when the value of the max-partitions property is 1 . This value will be used to initialize the shared property of storage groups that are created using this storage template. Default value: true .	All

Table 249. Storage Template object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
storage-volume-uris	(c)(pc)	Array of String/ URI	The list of storage template volumes for this storage template. Each element in this array is the canonical URI path of a Storage Template Volume object. This value will change when storage template volumes are added or removed from this storage template via the Modify Storage Template Properties operation. This value will be used to initialize the storage-volume-uris property of storage groups that are created using this storage template. Default value: An empty array.	All
connectivity	(w)(pc)	Integer (1-255)	The number of adapters to utilize for storage groups created from this template. When the type value is "fc", the maximum value for the connectivity property is 8. This value will be used to initialize the connectivity property of storage groups that are created using this storage template. Default value: 2 for storage templates of type "fc"; 8 for storage templates of type "fc".	All
max-partitions	(w)(pc)	Integer	The maximum number of partitions to which this storage template can be attached. The value of max-partitions cannot exceed 1 if the value of the shared field is false. The value of max-partitions must be greater than 1 if the value of the shared field is true. This value will be used to initialize the max-partitions property of storage groups that are created using this storage template. The minimum value for the max-partitions property is 1; the maximum is the value of the CPC object's maximum-partitions property. Default value: 2.	fc

Table 249. Storage Template object: class specific properties (continued)

Name	Qualifier	Type	Description	Supported "type" values
direct-connection-count	(w)(pc)	Integer (0-1000)	The number of additional virtual storage resource connections for the host that can be directly assigned to a guest virtual machine for storage groups created from this template. A value of zero indicates this feature is disabled. The direct-connection-count property cannot be enabled (greater than zero) when the shared property is true. This value will be used to initialize the direct-connection-count property of storage groups that are created using this storage template. Default value: 0.	fc

Storage Template Volume element object

A Storage Template Volume element object defines the size and usage of a single storage template volume within its parent storage template.

Table 250. Storage Template Volume element object properties

Name	Qualifier	Type	Description	Supported adapter "type" values
element-id	—	String (36)	Unique identifier for the storage template volume instance	All
element-uri	—	String/ URI	The canonical URI path for the storage template volume element object, of the form <code>/api/storage-templates/{storagetemplate-id}/storage-template-volumes/{storage-template-volume-id}</code> , where <code>{storage-template-id}</code> is the object-id of the containing storage template, and <code>{storage-template-volume-id}</code> is the element-id of this storage template volume.	All
parent	—	String/ URI	The parent of a storage template volume is its owning storage group, so the parent value is the canonical URI path for the storage template.	All
class	—	String (23)	The class of a Storage Template Volume element object is "storage-template-volume" .	All

Table 250. Storage Template Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported adapter "type" values
name	(w)(pc)	String (1-64)	The name of the storage template volume. The character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. Names must be unique to the other storage template volumes associated with the parent storage template. Default value: Currently of the form "# GiB {usage}{index}", where # is the size of the volume, {usage} is the usage, and {index} is a number that may or may not be present to ensure name uniqueness. This form is subject to change in the future.	All
description	(w)(pc)	String (0-100)	Arbitrary text providing additional descriptive information about the template volume. This value will be used to initialize the description property of storage volumes that are created using this storage template volume. Default value: An empty string.	All

Table 250. Storage Template Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported adapter "type" values
size	(w)(pc)	Float	The size in gibibytes (GiB) for this template volume. For volumes in storage templates of type "fcp", the size property must be between 1.00 and 1,048,576.00. For volumes in storage templates of type "fc", with a model value other than "EAV", the size property is fixed at the value specified below and cannot be written: Model - Maximum size "1" - 0.88 "2" - 1.76 "3" - 2.64 "9" - 7.92 "27" - 25.93 "54" - 51.86 For volumes with a model value of "EAV", size is writable and must be between 0.88 and 212489.20. Note however that the maximum is a theoretical size that may not be supported by an operating system. For volumes in storage templates of type "fc", the size value is related to the cylinders property. Specifying the size in GiB will also define the number of cylinders and vice versa. The size and cylinders properties cannot be specified together as fields in a create or modify operation. The value will be used to initialize the size property of storage volumes that are created using this storage template volume.	All
usage	(w)(pc)	String Enum	The usage of the storage volumes created from this template. Values: "boot" – storage volumes created from this template will contain a bootable image. "data" – storage volumes created from this template will contain data. This value will be used to initialize the usage property of storage volumes that are created using this storage template volume. Default value: "data".	All

Table 250. Storage Template Volume element object properties (continued)

Name	Qualifier	Type	Description	Supported adapter "type" values
model	(w)(pc)	String Enum	The 3390 model designation for the storage. Values: "1" - Model 1 "2" - Model 2 "3" - Model 3 "9" - Model 9 "27" - Model 27 "54" - Model 54 "EAV" - Extended Address Volume This value will be used to initialize the model property of storage volumes that are created using this storage template volume.	fc
cylinders	(w)(pc)	Integer (1113-268434453)	The size of the volume in cylinders. If the model value is other than "EAV", the cylinders property is fixed at the value specified below and cannot be written: Model - Cylinders "1" - 1113 "2" - 2226 "3" - 3339 "9" - 10017 "27" - 32760 "54" - 65520 For volumes with a model value of "EAV", cylinders is writable and has a maximum value of 268434453. Note however that this is a theoretical size that may not be supported by an operating system. The cylinders value is related to the size property. Specifying the size in cylinders will also define the size and vice versa. The cylinders and size properties cannot be specified together as fields in a create or modify operation. This value will be used to initialize the cylinders property of storage volumes that are created using this storage template volume.	fc
device-number	(w)(pc)	String (4)	A four-byte lower case hexadecimal string defining the device number that is assigned by default when FICON storage groups containing storage volumes created from this storage template are attached to partitions. This value may be null, in which case the system auto-assigns a device number. This value will be used to initialize the device-number property of storage volumes that are created using this storage template volume. Default value: null.	fc

List Storage Templates

The List Storage Templates operation lists the storage templates known to the target Console.

HTTP method and URI

GET /api/storage-templates

Query parameters:

Name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Optional	Filter string to limit returned objects to those that have a matching cpc-uri property.
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property value. Value must be a valid storage template type property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-templates	Array of storage- template- info objects	Array of storage-template-info objects, described in the next table. The returned array may be empty.

Each nested storage-template-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path (object-uri) of the Storage Template object.
cpc-uri	String/ URI	The cpc-uri property of the Storage Template object.
name	String	The name property of the Storage Template object.
type	String Enum	The type property of the Storage Template object.

Description

This operation lists the storage templates that are known by the target Console. The object URI, name, type and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage templates that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid value for the storage template **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage templates that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage templates that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage template is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a storage template, that object is simply omitted from the list but no error status code results.

If no storage templates are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Storage Template object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 541](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 251. List Storage Templates: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/storage-templates HTTP/1.1
x-api-session: 5jb1l6nadc5oggoh5l2xfwo8e1i24y1s6xude47kv4tkasb2ik
```

Figure 298. List Storage Templates: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:44:23 GMT
content-type: application/json; charset=UTF-8
content-length: 366
{
  "storage-templates": [
    {
      "cpc-uri": "/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
      "name": "FICON Template",
      "object-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
      "type": "fc"
    },
    {
      "cpc-uri": "/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
      "name": "FCP Template",
      "object-uri": "/api/storage-templates/6df0fbfa-e215-11e8-bffc-fa163e9c462b",
      "type": "fcp"
    }
  ]
}

```

Figure 299. List Storage Templates: Response

Create Storage Template

The Create Storage Template operation creates a new Storage Template object.

HTTP method and URI

POST /api/storage-templates

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
cpc-uri	String/ URI	Required	The value to be set as the storage template's cpc-uri property.
name	String (1-64)	Required	The value to be set as the storage template's name property.
description	String (0-200)	Optional	The value to be set as the storage template's description property.
type	String Enum	Required	The value to be set as the storage template's type property.
shared	Boolean	Optional	The value to be set as the storage template's shared property.
connectivity	Integer	Optional	The value to be set as the storage template's connectivity property.
max-partitions	Integer	Optional	The value to be set as the storage template's max-partitions property. The max-partitions field is not allowed in the request body unless the type field value has a value of "fcp" .

Field name	Type	Rqd/Opt	Description
direct-connection-count	Integer (0-1000)	Optional	The value to be set as the storage template's direct-connection-count property. The direct-connection-count field is not allowed in the request body unless the type field value has a value of " fcp ".
storage-template-volumes	Array of storage-template-volume-request-info nested objects	Optional	The set of properties for each of the storage template volumes in the template. An array of one or more storage-template-volume-request-info nested objects, where each element defines the new property values of a storage template volume that is to be created. The operation field of each nested object element must be set to " create ". For storage templates of type "fc" , the storage-template-volume-request-info nested object is defined in Table 212 on page 462 . For storage templates of type "fcp" , the storage-template-volume-request-info nested object is defined in Table 213 on page 463 .

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the newly created Storage Template object.
element-uris	Array of String/ URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the storage-template-volumes field in the request body. If no volumes were specified when creating the storage template, the element-uris field will be an empty list.

Description

This operation creates a storage template with the values specified and then returns its **object-uri** and the **element-uris** of each storage volume that was created in the response body. The response also includes a Location header that provides the new storage template's URI. Inventory Change notifications for the new storage template and for each new storage template volume identified in the **storage-template-volumes** field are emitted asynchronously to this operation.

If the API user does not have action/task permission to the Configure Storage – System Programmer task, a 403 (Forbidden) status code is returned. If the object ID **cpc-uri** field does not identify a CPC object to which the API user has object-access permission, or if the CPC identified by the **cpc-uri** field does not support this operation, a 404 (Not Found) status code is returned. If the CPC identified by the **cpc-uri** field is already associated with a storage template with the specified name, or if two or more of the new storage template volumes have the same name, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM, or if restrictions on the values of the **shared** and **max-partitions** property values, or the **shared** and **direct-connection-count** property values are violated, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage template type. If the request body contents are valid, the storage template and each of the storage template volumes defined in the

storage-template-volumes field are created and their properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 544.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 252. Create Storage Template: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage template with the name specified in the request body is already associated with the CPC identified by the cpc-uri field.
	15	The type field value is "fc" and an element of the storage-template-volumes field contains both the size and cylinders fields, or neither the size nor cylinders fields.
	18	A supplied property is not valid for a storage template's type.
	453	The name field for at least two of the nested entries in the storage-template-volumes field are the same.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	2	The CPC identified by the cpc-uri field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The CPC identified by the cpc-uri field does not support this operation.
409 (Conflict)	5	The CPC identified by the cpc-uri field is not enabled for DPM.
	8	The max-partitions field value conflicts with the shared field value.
	329	The operation cannot be performed because the CPC designated by the cpc-uri is an unmanaged CPC, which is not supported by this operation.
	495	The direct-connection-count field value conflicts with the shared field value.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-templates HTTP/1.1
x-api-session: 4572v9cswxkse3fgdfxclgakl6c9m67h46jrv06bkgoe42dzdv
content-type: application/json
content-length: 206
{
  "connectivity":4,
  "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
  "name":"FICON Template",
  "storage-template-volumes":[
    {
      "model":"1",
      "operation":"create",
      "usage":"boot"
    }
  ],
  "type":"fc"
}
```

Figure 300. Create Storage Template: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b
cache-control: no-cache
date: Tue, 06 Nov 2018 22:09:21 GMT
content-type: application/json; charset=UTF-8
content-length: 217
{
  "element-uris":[
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b"
  ],
  "object-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b"
}
```

Figure 301. Create Storage Template: Response

Delete Storage Template

The Delete Storage Template operation deletes a storage template.

HTTP method and URI

```
DELETE /api/storage-templates/{storage-template-id}
```

In this request, the URI variable *{storage-template-id}* is the object ID of the storage template to delete.

Description

This operation deletes a storage template. All of the template's contained storage template volume elements are also deleted. Inventory Change notifications for the deleted template and element objects are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission.

If the request body contents are valid, the identified storage template, and all its storage template volumes, are deleted from the CPC.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 253. Delete Storage Template: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage template with the object-id <i>{storage-template-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage template object with the object-id <i>{storage-template-id}</i> was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/storage-templates/6df0fbfa-e215-11e8-bffc-fa163e9c462b HTTP/1.1
x-api-session: 2p68kr4e521yrw0rggj9nad4ax541epqnzsihc67gxqrfpwye6
```

Figure 302. Delete Storage Template: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:56:32 GMT

<No response body>
```

Figure 303. Delete Storage Template: Response

Get Storage Template Properties

The Get Storage Template Properties operation retrieves the properties of a single Storage Template object.

HTTP method and URI

GET /api/storage-templates/{storage-template-id}

In this request, the URI variable {storage-template-id} is the object ID of the Storage Template object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Template object as defined in the [“Data model” on page 534](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Template object as defined in the [“Data model” on page 534](#).

If the object ID {storage-template-id} does not identify a Storage Template object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is {storage-template-id}.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents” on page 548](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 254. Get Storage Template Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	A storage template with object-id {storage-template-id} does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b HTTP/1.1
x-api-session: 61dhyf85cfr0vvvn4mt1cxuxbzd1akqx0yizl2kau2vv7e9fjlu
```

Figure 304. Get Storage Template Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:39:09 GMT
content-type: application/json;charset=UTF-8
content-length: 707
{
  "class": "storage-template",
  "connectivity": 4,
  "cpc-uri": "/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
  "creation-timestamp": 1541542161088,
  "description": "A sample FICON storage template",
  "modification-timestamp": 1541543867601,
  "name": "FICON Template",
  "object-id": "9cae8872-e210-11e8-82d0-fa163e9c462b",
  "object-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
  "parent": "/api/console",
  "shared": true,
  "storage-template-volume-uris": [
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b"
  ],
  "type": "fc"
}
```

Figure 305. Get Storage Template Properties: Response

Modify Storage Template Properties

The Modify Storage Template Properties operation updates one or more of the writable properties of a storage template.

HTTP method and URI

```
POST /api/storage-templates/{storage-template-id}/operations/modify
```

In this request, the URI variable *{storage-template-id}* is the object ID of the Storage Template object.

Request body contents

Fields for properties whose values are not to be changed by this operation can and should be omitted from the request body.

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Optional	The value to be set as the storage template's name property.
description	String (0-200)	Optional	The value to be set as the storage template's description property.
shared	Boolean	Optional	The value to be set as the storage template's shared property.

Field name	Type	Rqd/Opt	Description
connectivity	Integer	Optional	The value to be set as the storage template's connectivity property.
max-partitions	Integer	Optional	The value to be set as the storage template's max-partitions property. The max-partitions field is not allowed in the request body unless the type property has a value of "fcp" .
direct-connection-count	Integer (0-1000)	Optional	The value to be set as the storage template's direct-connection-count property. The direct-connection-count field is not allowed in the request body unless the type property has a value of "fcp" .
storage-template-volumes	Array of storage-template-volume-request-info nested objects	Optional	An array of storage-template-volume-request-info nested objects, where each element defines the existing storage template volumes that are to be deleted or the new property values of a storage template volume that is to be created or modified. The storage-template-volume-request-info nested object is defined in the tables that follow. If not specified, the storage template's volumes remain unchanged.

Each nested storage-volume-request-info object contains the following fields:

Table 255. storage-template-volume-request-info nested object for "create" operations on "fc" storage template volumes			
Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the property values for a new storage template volume that is to be created. Value: "create"
name	String (1-64)	Optional	The value to be set as the storage template volume's name property.
description	String (0-100)	Optional	The value to be set as the storage template volume's description property.
size	Float (0.88-212489.20)	Optional	The value to be set as the storage template volume's size property. If the model is "EAV" , the size or cylinders fields (but not both) must be specified. If the model is not "EAV" , the size field may not be present in the request body.
usage	String Enum	Optional	The value to be set as the storage template volume's usage property.
model	String Enum	Required	The value to be set as the storage template volume's model property.

Table 255. storage-template-volume-request-info nested object for **"create"** operations on **"fc"** storage template volumes (continued)

Field name	Type	Rqd/Opt	Description
cylinders	Integer (1113-268434453)	Optional	The value to be set as the storage template volume's cylinders property. If the model is "EAV" , the size or cylinders fields (but not both) must be specified. If the model is not "EAV" , the cylinders field may not be present in the request body.
device-number	String (4)	Optional	The value to be set as the storage template volume's device-number property.

Table 256. storage-template-volume-request-info nested object for **"create"** operations on **"fcp"** storage template volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the property values for a new storage template volume that is to be created. Value: "create"
name	String (1-64)	Optional	The value to be set as the storage template volume's name property.
description	String (0-100)	Optional	The value to be set as the storage template volume's description property.
size	Float (1.00-1048576.00)	Optional	The value to be set as the storage template volume's size property.
usage	String Enum	Optional	The value to be set as the storage template volume's usage property.

Table 257. storage-template-volume-request-info nested object for **"modify"** operations on **"fc"** storage template volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the new property values for an existing storage template volume that is to be modified. Value: "modify"
element-uri	String/URI	Required	The canonical URI path for the storage template volume element object that is being updated.
name	String (1-64)	Optional	The value to be set as the storage template volume's name property.
description	String (0-100)	Optional	The value to be set as the storage template volume's description property.

Table 257. storage-template-volume-request-info nested object for **"modify"** operations on **"fc"** storage template volumes (continued)

Field name	Type	Rqd/Opt	Description
size	Float (0.88-212489.20)	Optional	The value to be set as the storage template volume's size property. The size field is optional but may not be present in the request body if the cylinders field is present in the request body or if the model value is not "EAV" .
usage	String Enum	Optional	The value to be set as the storage template volume's usage property.
model	String Enum	Optional	The value to be set as the storage template volume's model property.
cylinders	Integer (1113-268434453)	Optional	The value to be set as the storage template volume's cylinders property. The cylinders field is optional but may not be present in the request body if the size field is present in the request body or if the model value is not "EAV" .
device-number	String (4)	Optional	The value to be set as the storage template volume's device-number property.

Table 258. storage-template-volume-request-info nested object for **"modify"** operations on **"fcp"** storage template volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object contains the new property values for an existing storage template volume that is to be modified. Value: "modify"
element-uri	String/ URI	Required	The canonical URI path for the storage template volume element object that is being updated.
name	String (1-64)	Optional	The value to be set as the storage template volume's name property.
description	String (0-100)	Optional	The value to be set as the storage template volume's description property.
size	Float (1.00-1048576.00)	Optional	The value to be set as the storage template volume's size property.
usage	String Enum	Optional	The value to be set as the storage template volume's usage property.

Table 259. storage-template-volume-request-info nested object for **"delete"** operations on **"fc"** or **"fcp"** storage template volumes

Field name	Type	Rqd/Opt	Description
operation	String Enum	Required	This nested object identifies an existing storage template volume that is to be deleted. Value: "delete"

Table 259. *storage-template-volume-request-info* nested object for **"delete"** operations on **"fc"** or **"fcp"** storage template volumes (continued)

Field name	Type	Rqd/Opt	Description
element-uri	String/ URI	Required	The canonical URI path for the storage template volume element object that is being deleted.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
element-uris	Array of String /URI	A list of the URIs for the storage template volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the storage-template-volumes field in the request body. If the storage-template-volumes field did not contain any entries with operation equal to "create" , the element-uris field will be an empty list.

Description

This operation updates a storage template's properties with the values specified and then returns the **element-uris** of each storage template volume that was created in the response body.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a Storage Template object to which the API user has object-access permission, or if the **element-uri** field of a modified or deleted template volume does not exist or is not a member of the storage template.

If the change would put the storage template into a state where its **shared** and **max-partitions** property values or **shared** and **direct-connection-count** property values conflict, or if the a storage template volume modification would put the volume into a state where its **model**, **size** and **cylinders** property values conflict, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage template type, or because the parent CPC is already associated with a storage template with the specified name, or because the operation would put the storage template into a state where two or more of its storage template volumes would have the same name, or because both of, or neither of, the **size** and **cylinders** fields of a EAV FICON storage template volume are defined.

If the request body contents are valid, the storage template's properties are updated to their corresponding request body content's field's values. Optional fields may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified. The element URIs of each new and deleted storage template volume will be added to, or removed from, the storage template's **storage-template-volume-uris** list property.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation. This includes a Property Change notification for the **storage-template-volume-uris** property if the operation creates or deletes storage template volumes.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 553.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 260. Modify Storage Template Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A storage template with the name specified in the request body is already associated with the CPC identified by the template's cpc-uri property.
	15	The storage template's type property value is "fc" and an element of the storage-volumes field contains both the size and cylinders fields, or an operation value of "create" and neither the size nor cylinders fields.
	18	A supplied property is not valid for a storage template's type.
	453	The operation would put the storage template into a state where the name property for at least two of its storage template volumes would be the same.
403 (Forbidden)	1	The API user does not have action/task permission to the Configure Storage – System Programmer task.
404 (Not Found)	1	A storage template with the object-id { <i>storage-template-id</i> } does not exist on the HMC or the API user does not have object-access permission for it.
	5	An element of the storage-template-volumes field has an element-uri value that does not exist on the HMC or the API user does not have object-access permission for it, or is not a member of the storage template's storage-template-volume-uris array property value.
409 (Conflict)	2	The storage template object with the object-id { <i>storage-template-id</i> } was busy and the request timed out.
	8	The operation would result in conflicting values for the max-partitions and shared property values.
	490	The storage template's type property value is "fc" and an element of the storage-template-volumes field would put the storage template volume into a state where its model , size and cylinders property values conflict.
	495	The operation would result in conflicting values for the direct-connection-count and shared property values.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/operations/modify HTTP/1.1
x-api-session: 1metap37hjlthezzcnd70br2kpn6aoyh97dgxt7rjm7bgmi04z
content-type: application/json
content-length: 376
{
  "description": "A sample FICON storage template",
  "storage-template-volumes": [
    {
      "description": "A Model 3 FICON data volume",
      "model": "3",
      "operation": "create"
    },
    {
      "description": "A Model 1 FICON boot volume",
      "element-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
      "operation": "modify"
    }
  ]
}
```

Figure 306. Modify Storage Template Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:37:46 GMT
content-type: application/json; charset=UTF-8
content-length: 142
{
  "element-uris": [
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b"
  ]
}
```

Figure 307. Modify Storage Template Properties: Response

List Storage Template Volumes of a Storage Template

The List Storage Template Volumes of a Storage Template operation lists the storage template volumes of the storage template with the given identifier.

HTTP method and URI

```
GET /api/storage-templates/{storage-template-id}/storage-template-volumes
```

In this request, the URI variable *{storage-template-id}* is the **object-id** of the Storage Template object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
maximum-size	Integer	Optional	Filter value to limit returned object to those that have a size property that is less than or equal to maximum-size .
minimum-size	Integer	Optional	Filter value to limit returned object to those that have a size property that is greater than or equal to minimum-size .

Name	Type	Rqd/Opt	Description
usage	String Enum	Optional	Filter string to limit returned objects to those that have a matching usage property. Value must be a valid storage template volume usage property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-template-volumes	Array of storage-template-volume-info objects	Array of storage-template-volume-info nested objects, described in the next table. The returned array may be empty.

Each nested storage-template-volume-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	Canonical URI path (element-uri) of the Storage Template Volume element object.
name	String	The name property of the storage template volume element.
size	Integer	The size property of the storage template volume element.
usage	String Enum	The usage property of the storage template volume element.

Description

This operation lists the storage template volumes that are owned by the identified storage template. The element URI, name, size and usage are provided for each.

If the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

If the **name** query parameter is specified, the returned list is limited to those storage template volumes that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **usage** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage template volume **usage** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage template volumes that have a **usage** property matching the specified value. If the **usage** parameter is omitted, this filtering is not done.

If the **minimum-size** query parameter is specified, the returned list is limited to those storage template volumes that have a **size** parameter that is greater than or equal to the specified value. If the **maximum-size** query parameter is specified, the returned list is limited to those storage template volumes that have a **size** parameter that is less than or equal to the specified value. When specified together, the **minimum-size** and **maximum-size** query parameters define a size range on which the volume is filtered. If either of these query parameters are omitted, the size filter is not bounded on one end. If both of these query parameters are omitted, no filtering on the **size** property is done.

If no storage template volumes are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 556.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 261. List Storage Template Volumes of a Storage Template: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage template with the object ID <i>{storage-template-id}</i> does not exist on the HMC or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes
HTTP/1.1
x-api-session: 4po625adh4w8byp0zc2uaqwjph44bkjh28afv14zf7f8qwan
```

Figure 308. List Storage Template Volumes of a Storage Template: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:40:13 GMT
content-type: application/json; charset=UTF-8
content-length: 410
{
  "storage-template-volumes": [
    {
      "element-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
      "name": "0.88 GiB Boot",
      "size": 0.88,
      "usage": "boot"
    },
    {
      "element-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b",
      "name": "2.64 GiB Data",
      "size": 2.64,
      "usage": "data"
    }
  ]
}

```

Figure 309. List Storage Template Volumes of a Storage Template: Response

Get Storage Template Volume Properties

The Get Storage Template Volume Properties operation retrieves the properties of a single Storage Template Volume element object.

HTTP method and URI

```
GET /api/storage-templates/{storage-template-id}/storage-template-volumes/{storage-template-volume-id}
```

In this request, the URI variable *{storage-template-id}* is the object ID of the Storage Template object and the URI variable *{storage-template-volume-id}* is the element ID of the Storage Template Volume element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Template Volume object as defined in the “Data model” on page 534. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage template volume object as defined in the “Storage Template Volume element object” on page 537.

A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission or if the element ID *{storage-template-volume-id}* does not identify a storage template volume in the storage template.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the [“Response body contents”](#) on page 558.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 262. Get Storage Template Volume Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	A storage template with object-id <i>{storage-template-id}</i> does not exist on the HMC or the API user does not have object-access permission to it.
	5	A storage template volume with element-id <i>{storage-template-volume-id}</i> does not exist in the storage template on the HMC.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/
9cc736a6-e210-11e8-82d0-fa163e9c462b HTTP/1.1
x-api-session: 35xby5sl8igqd0pzqxdfjmlad6vlsplfdi2svoeb5sv16qvhco
```

Figure 310. Get Storage Template Volume Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:41:23 GMT
content-type: application/json;charset=UTF-8
content-length: 440
{
  "class": "storage-template-volume",
  "cylinders": 1113,
  "description": "A Model 1 FICON boot volume",
  "device-number": null,
  "element-id": "9cc736a6-e210-11e8-82d0-fa163e9c462b",
  "element-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/
storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
  "model": "1",
  "name": "0.88 GiB Boot",
  "parent": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
  "size": 0.88,
  "usage": "boot"
}
```

Figure 311. Get Storage Template Volume Properties: Response

Inventory service data

Information about the Storage Templates managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Storage Template objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of **class "storage-template"** are to be included. Information for a particular storage template is included only if the API user has object-access permission to that object.

For each storage template to be included, the inventory response array includes the following:

- An array entry for the storage template object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Template Properties”](#) on page 548. That is, the data provided is the same as would be provided if a `Get Storage Template Properties` operation were requested targeting this object.
- An array entry for each storage template volume element associated with the storage template. For each such storage volume, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for [“Get Storage Template Volume Properties”](#) on page 558.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage template. These objects would appear as multiple array entries in the response array:

```
{
  "class": "storage-template",
  "connectivity": 4,
  "cpc-uri": "/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
  "creation-timestamp": 1541542161088,
  "description": "A sample FICON storage template",
  "modification-timestamp": 1541543867601,
  "name": "FICON Template",
  "object-id": "9cae8872-e210-11e8-82d0-fa163e9c462b",
  "object-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
  "parent": "/api/console",
  "shared": true,
  "storage-template-volume-uris": [
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
    "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b"
  ],
  "type": "fc"
},
{
  "class": "storage-template-volume",
  "cylinders": 1113,
  "description": "A Model 1 FICON boot volume",
  "device-number": null,
  "element-id": "9cc736a6-e210-11e8-82d0-fa163e9c462b",
  "element-uri": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
  "model": "1",
  "name": "0.88 GiB Boot",
  "parent": "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
  "size": 0.88,
  "usage": "boot"
}
```

Figure 312. Storage Template object: Sample inventory data - Response

Chapter 11. Core IBM Z resources

These APIs provide access to and control of the following Z and LinuxONE HMC/SE objects:

- Console
- Group
- CPC
- Logical Partition

In addition, these APIs provide access to the following CPC-related data items:

- Reset Activation Profiles
- Image Activation Profiles
- Load Activation Profiles
- Group Profiles
- Capacity Records

Operations Summary

Following are the operations summaries for each of the core Z and LinuxONE objects.

Console operations summary

The following table provides an overview of the operations provided for Console objects.

Table 263. Core Z resources - Console: operations summary	
Operation name	HTTP method and URI path
“Get Console Properties” on page 582	GET /api/console
“Restart Console” on page 588	POST /api/console/operations/restart
“Shutdown Console” on page 589	POST /api/console/operations/shutdown
“Reorder User Patterns” on page 591	POST /api/console/operations/reorder-user-patterns
“Get Console Audit Log” on page 593	GET /api/console/operations/get-audit-log
“Get Console Security Log” on page 598	GET /api/console/operations/get-security-log
“List Console Hardware Messages” on page 601	GET /api/console/hardware-messages
“Get Console Hardware Message Properties” on page 603	GET /api/console/hardware-messages/{hardware-message-id}

Table 263. Core Z resources - Console: operations summary (continued)

Operation name	HTTP method and URI path
“Delete Console Hardware Message” on page 605	DELETE /api/console/hardware-messages/{hardware-message-id}
“Request Console Service” on page 606	POST /api/console/hardware-messages/{hardware-message-id}/operations/request-service
“Get Console Service Request Information” on page 608	GET /api/console/hardware-messages/{hardware-message-id}/operations/get-service-information
“Decline Console Service” on page 610	POST /api/console/hardware-messages/{hardware-message-id}/operations/decline-service
“List Unmanaged CPCs” on page 611	GET /api/console/operations/list-unmanaged-cpcs
“Get Mobile App Preferences” on page 613	GET /api/console/operations/get-mobile-app-preferences
“Set Mobile App Preferences” on page 615	POST /api/console/operations/set-mobile-app-preferences
“Get CPC Notification Preferences for Device” on page 618	POST /api/console/operations/get-device-cpc-notification-preferences
“Update CPC Notification Preferences for Device” on page 622	POST /api/console/operations/update-device-cpc-notification-preferences

Table 264. Core Z resources - Console: URI variables

URI variable	Description
{hardware-message-id}	Element ID of the hardware message object

User operations summary

The following table provides an overview of the operations provided for User objects.

Table 265. Core Z resources - User: operations summary

Operation name	HTTP method and URI path
“List Users” on page 637	GET /api/console/users
“Get User Properties” on page 639	GET /api/users/{user-id}
“Update User Properties” on page 641	POST /api/users/{user-id}

Table 265. Core Z resources - User: operations summary (continued)	
Operation name	HTTP method and URI path
“Add User Role to User” on page 644	POST /api/users/{user-id}/operations/add-user-role
“Remove User Role from User” on page 646	POST /api/users/{user-id}/operations/remove-user-role
“Create User” on page 648	POST /api/console/users
“Delete User” on page 652	DELETE /api/users/{user-id}

Table 266. Core Z resources - User: URI variables	
URI variable	Description
{user-id}	Object ID of the User object

User Role operations summary

The following table provides an overview of the operations provided for User Role objects.

Table 267. Core Z resources - User Role: operations summary	
Operation name	HTTP method and URI path
“List User Roles” on page 659	GET /api/console/user-roles
“Get User Role Properties” on page 661	GET /api/user-roles/{user-role-id}
“Update User Role Properties” on page 663	POST /api/user-roles/{user-role-id}
“Add Permission to User Role” on page 665	POST /api/user-roles/{user-role-id}/operations/add-permission
“Remove Permission from User Role” on page 668	POST /api/user-roles/{user-role-id}/operations/remove-permission
“Create User Role” on page 671	POST /api/console/user-roles
“Delete User Role” on page 673	DELETE /api/user-roles/{user-role-id}

Table 268. Core Z resources - User Role: URI variables	
URI variable	Description
{user-role-id}	Object ID of the User Role object

Task operations summary

The following table provides an overview of the operations provided for Task objects.

Table 269. Core Z resources - Task: operations summary	
Operation name	HTTP method and URI path
“List Tasks” on page 676	GET /api/console/tasks
“Get Task Properties” on page 678	GET /api/console/tasks/{task-id}

Table 270. Core Z resources - Task: URI variables	
URI variable	Description
{task-id}	Element ID of the Task object

User Pattern operations summary

The following table provides an overview of the operations provided for User Pattern objects.

Table 271. Core Z resources - User Pattern: operations summary	
Operation name	HTTP method and URI path
“List User Patterns” on page 682	GET /api/console/user-patterns
“Get User Pattern Properties” on page 684	GET /api/console/user-patterns/{user-pattern-id}
“Update User Pattern Properties” on page 685	POST /api/console/user-patterns/{user-pattern-id}
“Create User Pattern” on page 687	POST /api/console/user-patterns
“Delete User Pattern” on page 690	DELETE /api/console/user-patterns/{user-pattern-id}

Table 272. Core Z resources - User Pattern: URI variables	
URI variable	Description
{user-pattern-id}	Element ID of the User Pattern object

Password Rule operations summary

The following table provides an overview of the operations provided for Password Rule objects.

Table 273. Core Z resources - Password Rule: operations summary	
Operation name	HTTP method and URI path
“List Password Rules” on page 695	GET /api/console/password-rules

Table 273. Core Z resources - Password Rule: operations summary (continued)	
Operation name	HTTP method and URI path
“Get Password Rule Properties” on page 697	GET /api/console/password-rules/{password-rule-id}
“Update Password Rule Properties” on page 699	POST /api/console/password-rules/{password-rule-id}
“Create Password Rule” on page 701	POST /api/console/password-rules
“Delete Password Rule” on page 703	DELETE /api/console/password-rules/{password-rule-id}

Table 274. Core Z resources - Password Rule: URI variables	
URI variable	Description
{password-rule-id}	Element ID of the Password Rule object

LDAP Server Definition operations summary

The following table provides an overview of the operations provided for LDAP Server Definition objects.

Table 275. Core Z resources - LDAP Server Definition: operations summary	
Operation name	HTTP method and URI path
“List LDAP Server Definitions” on page 708	GET /api/console/ldap-server-definitions
“Get LDAP Server Definition Properties” on page 710	GET /api/console/ldap-server-definitions/{ldap-server-definition-id}
“Update LDAP Server Definition Properties” on page 712	POST /api/console/ldap-server-definitions/{ldap-server-definition-id}
“Create LDAP Server Definition” on page 714	POST /api/console/ldap-server-definitions
“Delete LDAP Server Definition” on page 716	DELETE /api/console/ldap-server-definitions/{ldap-server-definition-id}

Table 276. Core Z resources - LDAP Server Definition: URI variables	
URI variable	Description
{ldap-server-definition-id}	Element ID of the LDAP Server Definition object

MFA Server Definition operations summary

The following table provides an overview of the operations provided for MFA Server Definition objects.

Table 277. Core Z resources - MFA Server Definition: operations summary

Operation name	HTTP method and URI path
“List MFA Server Definitions” on page 719	GET /api/console/mfa-server-definitions
“Get MFA Server Definition Properties” on page 721	GET /api/console/mfa-server-definitions/{mfa-server-definition-id}
“Update MFA Server Definition Properties” on page 723	POST /api/console/mfa-server-definitions/{mfa-server-definition-id}
“Create MFA Server Definition” on page 724	POST /api/console/mfa-server-definitions
“Delete MFA Server Definition” on page 726	DELETE /api/console/mfa-server-definitions/{mfa-server-definition-id}

Table 278. Core Z resources - MFA Server Definition: URI variables

URI variable	Description
{mfa-server-definition-id}	Element ID of the MFA Server Definition object

Group operations summary

The following table provides an overview of the operations provided for Group objects.

Table 279. Core Z resources - Group: operations summary

Operation name	HTTP method and URI path
“List Custom Groups” on page 730	GET /api/groups
“Get Custom Group Properties” on page 731	GET /api/groups/{group-id}
“Create Custom Group” on page 733	POST /api/groups
“Delete Custom Group” on page 735	DELETE /api/groups/{group-id}
“List Custom Group Members” on page 739	GET /api/groups/{group-id}/members
“Add Member to Custom Group” on page 736	POST /api/groups/{group-id}/operations/add-member
“Remove Member from Custom Group” on page 738	POST /api/groups/{group-id}/operations/remove-member

Table 280. Core Z resources - Groups: URI variables	
URI variable	Description
{group-id}	Object ID of a Group object

CPC operations summary

The following tables provide an overview of the operations provided for CPC objects.

Table 281. Core Z resources - CPC: operations summary	
Operation name	HTTP method and URI path
“List CPC Objects” on page 759	GET /api/cpcs
“Get CPC Properties” on page 761	GET /api/cpcs/{cpc-id}
“Update CPC Properties” on page 769	POST /api/cpcs/{cpc-id}
“Start CPC” on page 771	POST /api/cpcs/{cpc-id}/operations/start
“Stop CPC” on page 773	POST /api/cpcs/{cpc-id}/operations/stop
“Activate CPC” on page 775	POST /api/cpcs/{cpc-id}/operations/activate
“Deactivate CPC” on page 778	POST /api/cpcs/{cpc-id}/operations/deactivate
“Import Profiles” on page 780	POST /api/cpcs/{cpc-id}/operations/import-profiles
“Export Profiles” on page 781	POST /api/cpcs/{cpc-id}/operations/export-profiles
“Set Auto-Start List” on page 783	POST /api/cpcs/{cpc-id}/operations/set-auto-start-list
“Add Temporary Capacity” on page 785	POST /api/cpcs/{cpc-id}/operations/add-temp-capacity
“Remove Temporary Capacity” on page 788	POST /api/cpcs/{cpc-id}/operations/remove-temp-capacity
“Swap Current Time Server” on page 789	POST /api/cpcs/{cpc-id}/operations/swap-cts
“Set STP Configuration” on page 791	POST /api/cpcs/{cpc-id}/operations/set-stp-config
“Change STP-only Coordinated Timing Network” on page 793	POST /api/cpcs/{cpc-id}/operations/change-stponly-ctn

Table 281. Core Z resources - CPC: operations summary (continued)

Operation name	HTTP method and URI path
“Join STP-only Coordinated Timing Network” on page 794	POST /api/cpcs/{cpc-id}/operations/join-stponly-ctn
“Leave STP-only Coordinated Timing Network” on page 796	POST /api/cpcs/{cpc-id}/operations/leave-stponly-ctn
“Get CPC Audit Log” on page 797	GET /api/cpcs/{cpc-id}/operations/get-audit-log
“Get CPC Security Log” on page 799	GET /api/cpcs/{cpc-id}/operations/get-security-log
“List CPC Hardware Messages” on page 802	GET /api/cpcs/{cpc-id}/hardware-messages
“Get CPC Hardware Message Properties” on page 805	GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
“Delete CPC Hardware Message” on page 807	DELETE /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
“Request CPC Service” on page 809	POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/request-service
“Get CPC Service Request Information” on page 811	GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/get-service-information
“Decline CPC Service” on page 813	POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/decline-service
“Export WWPN List” on page 814	POST /api/cpcs/{cpc-id}/operations/export-port-names-list
“Import DPM Configuration” on page 817	POST /api/cpcs/{cpc-id}/operations/import-dpm-config

Table 282. Core Z resources - CPC: URI variables

URI variable	Description
{cpc-id}	Object ID of a CPC object
{hardware-message-id}	Element ID of the hardware message object

Logical partition operations summary

The following tables provide an overview of the operations provided for Logical Partition objects.

Table 283. Core Z resources - Logical partitions: operations summary

Operation name	HTTP method and URI path
“List Logical Partitions of CPC” on page 850	GET /api/cpcs/{cpc-id}/logical-partitions
“List Permitted Logical Partitions” on page 852	GET /api/console/operations/list-permitted-logical-partitions
“Get Logical Partition Properties” on page 855	GET /api/logical-partitions/{logical-partition-id}
“Update Logical Partition Properties” on page 860	POST /api/logical-partitions/{logical-partition-id}
“Activate Logical Partition” on page 861	POST /api/logical-partitions/{logical-partition-id}/operations/activate
“Deactivate Logical Partition” on page 863	POST /api/logical-partitions/{logical-partition-id}/operations/deactivate
“Reset Normal” on page 865	POST /api/logical-partitions/{logical-partition-id}/operations/reset-normal
“Reset Clear” on page 867	POST /api/logical-partitions/{logical-partition-id}/operations/reset-clear
“Load Logical Partition” on page 869	POST /api/logical-partitions/{logical-partition-id}/operations/load
“PSW Restart” on page 871	POST /api/logical-partitions/{logical-partition-id}/operations/psw-restart
“Start Logical Partition” on page 873	POST /api/logical-partitions/{logical-partition-id}/operations/start
“Stop Logical Partition” on page 874	POST /api/logical-partitions/{logical-partition-id}/operations/stop
“Send OS Command” on page 876	POST /api/logical-partitions/{logical-partition-id}/operations/send-os-cmd
“Open OS Message Channel” on page 878	POST /api/logical-partitions/{logical-partition-id}/operations/open-os-message-channel
“List OS Messages of a Logical Partition” on page 880	GET /api/logical-partitions/{logical-partition-id}/operations/list-os-messages
“Delete Logical Partition OS Message” on page 883	POST /api/logical-partitions/{logical-partition-id}/operations/delete-os-message
“SCSI Load” on page 885	POST /api/logical-partitions/{logical-partition-id}/operations/scsi-load
“SCSI Dump” on page 887	POST /api/logical-partitions/{logical-partition-id}/operations/scsi-dump

Table 283. Core Z resources - Logical partitions: operations summary (continued)	
Operation name	HTTP method and URI path
“List Managed Virtual Machines of a Logical Partition” on page 890	GET /api/logical-partitions/{ <i>logical-partition-id</i> }/managed-virtual-machines

Table 284. Core Z resources - Logical partitions: URI variables	
URI variable	Description
{ <i>cpc-id</i> }	Object ID of a CPC object
{ <i>logical-partition-id</i> }	Object ID of a Logical Partition object

Activation profile operations summary

The following tables provide an overview of the operations provided for the various types of Activation Profile objects.

Table 285. Core Z resources - Reset activation profile: operations summary	
Operation name	HTTP method and URI path
“List Reset Activation Profiles” on page 894	GET /api/cpcs/{ <i>cpc-id</i> }/reset-activation-profiles
“Get Reset Activation Profile Properties” on page 896	GET /api/cpcs/{ <i>cpc-id</i> }/reset-activation-profiles/{ <i>reset-activation-profile-name</i> }
“Update Reset Activation Profile Properties” on page 898	POST /api/cpcs/{ <i>cpc-id</i> }/reset-activation-profiles/{ <i>reset-activation-profile-name</i> }

Table 286. Core Z resources - Image activation profile: operations summary	
Operation name	HTTP method and URI path
“List Image Activation Profiles” on page 915	GET /api/cpcs/{ <i>cpc-id</i> }/image-activation-profiles
“Get Image Activation Profile Properties” on page 917	GET /api/cpcs/{ <i>cpc-id</i> }/image-activation-profiles/{ <i>image-activation-profile-name</i> }
“Update Image Activation Profile Properties” on page 920	POST /api/cpcs/{ <i>cpc-id</i> }/image-activation-profiles/{ <i>image-activation-profile-name</i> }

Table 287. Core Z resources - Load activation profile: operations summary	
Operation name	HTTP method and URI path
“List Load Activation Profiles” on page 925	GET /api/cpcs/{ <i>cpc-id</i> }/load-activation-profiles
“Get Load Activation Profile Properties” on page 927	GET /api/cpcs/{ <i>cpc-id</i> }/load-activation-profiles/{ <i>load-activation-profile-name</i> }

Table 287. Core Z resources - Load activation profile: operations summary (continued)	
Operation name	HTTP method and URI path
“Update Load Activation Profile Properties” on page 930	POST /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}

Table 288. Core Z resources - Group profile: operations summary	
Operation name	HTTP method and URI path
“List Group Profiles” on page 933	GET /api/cpcs/{cpc-id}/group-profiles
“Get Group Profile Properties” on page 935	GET /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}
“Update Group Profile Properties” on page 937	POST /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}

Table 289. Core Z resources - Activation profile: URI variables	
URI variable	Description
{cpc-id}	Object ID of a CPC object
{group-profile-name}	Group profile name
{image-activation-profile-name}	Image activation profile name
{load-activation-profile-name}	Load activation profile name
{reset-activation-profile-name}	Reset activation profile name

Capacity record operations summary

The following tables provide an overview of the operations provided for Capacity Record objects.

Table 290. Core Z resources - Capacity record: operations summary	
Operation name	HTTP method and URI path
“List Capacity Records” on page 942	GET /api/cpcs/{cpc-id}/capacity-records
“Get Capacity Record Properties” on page 943	GET /api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}

Table 291. Core Z resources - Capacity record: URI variables	
URI variable	Description
{cpc-id}	Object ID of a CPC object
{capacity-record-id}	Capacity record identifier

Shared nested objects

Some of the Core API objects share common nested objects and are documented here for ease of reference.

Table 292. ec-mcl-description object		
Field name	Type	Description
actions	Array of action objects	An optional array of pending action objects. This field is not provided when the EC MCL information pertains to the HMC. When the EC MCL information pertains to an SE, this field is only provided when the HMC is communicating with the CPC's SE, and this field will be null if the information is currently unavailable or there are no pending actions.
ec	Array of ec objects	An optional array of EC objects. When the EC MCL information pertains to an SE, this field is only provided when the HMC is communicating with the CPC's SE, and this field will be null if the information is currently unavailable.

Table 293. action object		
Field name	Type	Description
type	String Enum	One of: • "channel-config" - channels pending a config on/off • "coupling-facility-reactivation" - at least one coupling facility pending reactivation • "power-on-reset-tracking" - there is a need for a power-on-reset
activation	String Enum	One of: • "current" - the action is for the current activation • "next" - the action is for the next install and activation.
pending	Boolean	Is the action pending (true) or not pending (false)

Table 294. ec object		
Field name	Type	Description
number	String (1-6)	Engineering Change stream identifier.
part-number	String (1-8)	Engineering Change stream part number.
type	String (1-32)	Engineering Change stream name.
description	String (1-65)	Engineering Change stream descriptive text.
mcl	Array of mcl objects	The list of MicroCode Levels for this Engineering Change.

Table 295. mcl object

Field name	Type	Description
type	String Enum	One of: • "retrieved" - a retrieved or staged level • "activated" - an activated or applied level • "accepted" - a committed level • "installable-concurrent" - a non-disruptive apply-able level • "removable-concurrent" - a non-disruptive reject-able level.
level	String (1-3)	Microcode level.
last-update	Timestamp	Time stamp of the last update, in the number of milliseconds since midnight January 1, 1970 UTC. A null object is returned if no updates have occurred.

Table 296. stp-config object

Field name	Type	Description
stp-id	String (0-8)	If in STP-only or Mixed CTN, the STP identifier. Otherwise, an empty string. Valid characters are 0-9, a-z, A-Z, underscore(_) and dash(-).
etr-id	Integer (0-31)	ETR Identifier, if in ETR mode. If not in ETR mode, a null object is returned.
preferred-time-server	stp-node-object	Describes the Preferred Timer Server. This property is optional, only returned on a Get request when the information is set.
backup-time-server	stp-node-object	Describes the Backup Timer Server. This property is optional, only returned on a Get request when the information is set.
arbiter	stp-node-object	Describes the arbiter of the CTN. This property is optional, only returned on a Get request when the information is set.
current-time-server	String Enum	Describes the CPC's role in the CTN. One of: • "preferred" - CPC is the Preferred Time Server • "backup" - CPC is the Backup Time Server.

This object is used to identify a CPC to the STP services. When used as input on the Set STP Configuration operation, if the **object-uri** field is not provided, all other fields are required. If the **object-uri** field is provided, all other fields are optional. If all fields are provided, the **object-uri** field is ignored.

Table 297. stp-node object

Field name	Type	Description
object-uri	String URI	If the CPC is known to the HMC, contains the CPC's object-uri. Otherwise, contains a null object
type	String (0-6)	The CPC machine type, right justified and left padded with zeros or a empty string
model	String (0-3)	The CPC machine model or a empty string

Table 297. stp-node object (continued)

Field name	Type	Description
manuf	String (0-3)	The CPC manufacturer or a empty string
po-manuf	String (0-2)	The CPC plant of manufacturer or a empty string
seq-num	String (0-12)	The CPC sequence number or a empty string

Table 298. psw-description object

Field name	Type	Description
psw	String	Program Status Word (PSW) information for a single processor.
cpid	String (2)	The hexadecimal processor identifier, right justified and left padded with zeros.

Table 299. zaware-network object

Field name	Type	Description
chpid	String (2)	The required network adapter channel path identifier, in hexadecimal characters 0-9,a-f,A-F. The format is two hexadecimal digits (00-FF).
ipaddr-type	String Enum	Indicates how this network adapter's IP address is obtained. One of the following values: • "dhcp" - obtains an IP address via DHCP • "link-local" - obtains a link-local IP address • "static" - uses the specified IP address information.
vlan-id	Integer (0-4094)	If this network adapter is attached to a Virtual LAN, this field contains the VLAN identifier. Otherwise, a null object indicating that this network adapter is not attached to a Virtual LAN.
static-ip-info	network-ip-info object	When ipaddr-type is "static" , contains the static IP information. Otherwise, contains a null object indicating that a static IP address is not used.

Table 300. ssc-network object

Field name	Type	Description
chpid	String (2)	The required network adapter channel path identifier, in hexadecimal characters 0-9,a-f,A-F. The format is two hexadecimal digits (00-FF).
port	Integer	The number of the port on the network adapter identified by chpid . When the associated Support Element is at Version 2.13.1 or earlier, this property is not returned on a Get operation, and it cannot be specified on an Update operation.

Table 300. ssc-network object (continued)

Field name	Type	Description
ipaddr-type	String Enum	Indicates how this network adapter's IP address is obtained. One of the following values: • "dhcp" - obtains an IP address via DHCP • "link-local" - obtains a link-local IP address • "static" - uses the IP address information specified in static-ip-info.
vlan-id	Integer (0-4094)	If this network adapter is attached to a Virtual LAN, this field contains the VLAN identifier. Otherwise, a null object indicating that this network adapter is not attached to a Virtual LAN.
static-ip-info	network-ip-info object	When ipaddr-type is "static" , contains the static IP information. Otherwise, contains a null object indicating that a static IP address is not used.

Table 301. ip-info object

Field name	Type	Description
type	String Enum	The type of IP address being provided. One of the following values: • "ipv4" - an IPv4 address is provided • "ipv6" - an IPv6 address is provided.
ip-address	String/ IPV4 address or String/ IPV6 address	The IP address to be used. The format of the string (IPv4 or IPv6) must be as indicated by the type field.

Table 302. network-ip-info object

Field name	Type	Description
type	String Enum	The type of IP address being provided. One of the following values: • "ipv4" - an IPv4 address is provided • "ipv6" - an IPv6 address is provided.
ip-address	String/ IPV4 address or String/ IPV6 address	The IP address to be used. The format of the string (IPv4 or IPv6) must be as indicated by the type field.
prefix	Integer	The number of leading bits of ip-address that represent the network prefix. • When type is "ipv4" - valid values are 0-32 • When type is "ipv6" - valid values are 0-128.

Table 303. absolute-capping object		
Field name	Type	Description
type	String Enum	The type of absolute capping. One of the following values: • "none" - no absolute capping • "processors" - processor type absolute capping.
value	Float	When type is "none" , value is not specified. When type is "processors" , value is in the range .01...255.00 in increments of .01 and is the limit of usage independent of priority.

Console object

The Console object represents the single Z Hardware Management Console (HMC) application. The Console object offers a heterogeneous set of services and capabilities, from basic HMC control operations to general HMC information.

Object-access to the single Console object representing the local HMC is automatic for all authenticated users. A Console may (but need not) participate in a { Primary, Alternate } pairing with another Console. The Console object helps facilitate the management of a HMC's role in such a pairing.

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 70, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 68.

The following class-specific specializations apply to the other base managed object properties:

Table 304. Console object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/URI	The canonical URI path of the Console object, of the form /api/console
parent	—	String/URI	A Console object has no parent, so this property is always a null object.
class	—	String	The class of a Console object is "console" .
name	(ro)	String	The installation assigned name
description	(ro)	String	This property is not supported, and returned as null.

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 305. Console object: class specific additional properties			
Name	Qualifier	Type	Description
version	—	String (1-8)	The version number for the Console object.

Table 305. Console object: class specific additional properties (continued)

Name	Qualifier	Type	Description
ec-mcl-description	—	ec-mcl-description object	A nested object that describes the EC (Engineering Change) and MCL (Microcode Level) for the Console. Refer to the description of the ec-mcl-description object for details.
network-info	—	network-info object	A nested object describing the network information for this Hardware Management Console.
machine-info	—	machine-info object	A nested object describing the machine's BIOS characteristics.
hardware-messages	(c)(pc)	Array of hardware-message objects	The complete list of all Console hardware messages, each identified by its URI. This list corresponds to the list provided by the <code>List Console Hardware Messages</code> operation. If the console has no hardware messages, then an empty array is provided. The list of returned hardware messages can change as the result of the new messages being dynamically added or removed by the infrastructure or due to hardware messages being deleted via the <code>Delete Console Hardware Message</code> operation. Note: This property is not returned by the <code>Get Console Properties</code> operation, and only sessions associated with an HMC user with permission to the Hardware Messages task will receive a property-change notification for this property.
mobile-app-preferences	(p)(pc)	mobile-app-preferences object	The JSON nested object containing the mobile app preferences, as described in Table 317 on page 580. This property can be retrieved through the <code>Get Mobile App Preferences</code> operation, and modified through the <code>Set Mobile App Preferences</code> operation.

Table 306. network-info object properties

Name	Type	Description
this-hmc	Array of detailed-network-info objects	The collection of network information for the local Hardware Management Console. The number of objects returned is a function of the machine model and type on which the Hardware Management Console is executing. This information is available in the machine-info property.

Table 307. detailed-network-info properties

Name	Type	Description
hmc-name	String (1-16)	The Hardware Management Console name
interface-name	String	The network interface name
domain-name	String (1-255)	The domain name configured for this network interface

Table 307. detailed-network-info properties (continued)

Name	Type	Description
is-private	Boolean	Whether the interface is private (true) or public (false).
mac	String (1-12)	The MAC address of this network interface.
ipv4-address	Array of ipv4-info objects	A collection of nested objects which describe the IPv4 addresses for this network interface.
ipv6-address	Array of ipv6-info objects	A collection of nested objects which describe the IPv6 addresses for this network interface.

Table 308. ipv4-info properties

Name	Qualifier	Type	Description
subnet-mask	(pc)	String (1-15)	The IP mask value
ip-address	(pc)	String IPv4 address	The IPv4 address

Table 309. ipv6-info properties

Name	Qualifier	Type	Description
prefix-length	(pc)	Integer	The number of leading bits of the IPv6 address that represent the network prefix.
ip-address	(pc)	String IPv6 address	The IPv6 address

Table 310. machine-info properties

Name	Type	Description
machine-type	String (1-4)	The type of machine on which the Hardware Management Console is executing.
machine-model	String (1-3)	The model of machine on which the Hardware Management Console is executing.
machine-serial	String (1-10)	The serial number of machine on which the Hardware Management Console is executing.

Table 311. hardware-message object properties

Name	Type	Description
element-uri	String/ URI	The canonical URI path of the Console hardware message. The URI is in the following form: /api/console/hardware-messages/{hardware-message-id}, where {hardware-message-id} is the value of the element-id property of the hardware message.

Table 311. hardware-message object properties (continued)

Name	Type	Description
element-id	String (36)	The unique identifier for the hardware message. The element-id is in the form of a UUID.
parent	String/ URI	The parent of a console hardware message is the Console object. The parent value is the canonical URI path for the console.
class	String	The class of a hardware message object is " hardware-message ".
timestamp	Timestamp	The timestamp represents the date and time when the hardware message was created.
service-supported	Boolean	Indicates whether or not this hardware message represents a problem for which service may be requested. True is returned if the hardware message supports service, or false is returned if it does not.
text	String	The text of the hardware message.
details	Object	A hardware-message-details object if there are hardware message details, or null if there are no hardware message details.

Every hardware-message-details object contains the following base properties:

Table 312. hardware-message-details base properties

Name	Type	Description
type	String Enum	The type of detail available for this hardware message. Valid values: • "basic" - basic details in the form of one or more text messages. • "common-problem" - specific details for this problem-related hardware message.

A hardware-message-details object with a **type** value of "**basic**" provides general details. In addition to the base properties, it contains the following type-specific properties:

Table 313. hardware-message-details type-specific properties when the **type** value is "**basic**":

Name	Type	Description
messages	Array of String	The detail messages for the hardware message.

A hardware-message-details object with a **type** value of "**common-problem**" provides problem details. In addition to the base properties, it contains the following type-specific properties:

Table 314. hardware-message-details type-specific properties when the **type** value is "**common-problem**":

Name	Type	Description
created	Timestamp	The timestamp representing when the problem was created.
description	Array of String	The problem description, or null if there is no description.

*Table 314. hardware-message-details type-specific properties when the **type** value is "**common-problem**": (continued)*

Name	Type	Description
corrective-action	Array of String	The corrective action for the problem, or null if there is no corrective action.
repair-impact	Array of String	The repair impact for the problem, or null if there is no repair impact.
problem-data	Array of Object	An array of hardware-message-data-details objects for the problem data, or null if there is no problem data.
lir-node-data	Array of Object	An array of hardware-message-node-details objects for the problem Link Incident Record (LIR) node data, or null if there is no LIR node data. A Link Incident Record is reported by the Channel Subsystem for a hardware problem on an ESCON or FICON link.

A hardware-message-node-details object provides a title along with associated data detail:

Table 315. hardware-message-node-details:

Name	Type	Description
title	String	The node title.
details	Array of Object	An array of hardware-message-data-details objects.

A hardware-message-data-details object provides simple data detail:

Table 316. hardware-message-data-details:

Name	Type	Description
caption	String	The data caption.
value	String	The data value.

The mobile-app-preferences nested object contains the following fields:

Table 317. mobile-app-preferences object properties

Name	Type	Description
app-enabled	Boolean	True indicates that the mobile app is enabled for this console. False indicates that the mobile app is disabled and will not be allowed to access this console.
require-app-password-enabled	Boolean	True indicates that the mobile app must require its user to set an app password in order to access the app itself. False indicates the user is not required to secure the app with a password.
password-caching-enabled	Boolean	True indicates that the mobile app is permitted to securely store the user's password on the device for this console. False indicates this is not permitted, and the user must input the password on every logon to this console from the app.

Table 317. mobile-app-preferences object properties (continued)

Name	Type	Description
actions-enabled	Boolean	True indicates that the mobile app is permitted to perform actions against this console and the objects it manages, as permitted by the user's authority. False indicates that no actions are permitted from the mobile app to this console, and the app will be used only for monitoring.
action-settings-activate-partition	action-settings object	The mobile app settings information for the activate logical partition action.
action-settings-deactivate-partition	action-settings object	The mobile app settings information for the deactivate logical partition action.
action-settings-start-partition	action-settings object	The mobile app settings information for the start DPM partition action.
action-settings-stop-partition	action-settings object	The mobile app settings information for the stop DPM partition action.
action-settings-change-activation-profile	action-settings object	The mobile app settings information for the change activation profile of a logical partition action.
action-settings-load-os-into-partition	action-settings object	The mobile app settings information for the load OS into logical partition action.
action-settings-reset-partition	action-settings object	The mobile app settings information for the reset logical partition action.
action-settings-change-partition-weight	action-settings object	The mobile app settings information for the change DPM partition or logical partition weights and capping action.
action-settings-change-partition-processors	action-settings object	The mobile app settings information for the change DPM partition or logical partition processors action.
action-settings-delete-hardware-message	action-settings object	The mobile app settings information for the delete hardware message action.
action-settings-delete-os-message	action-settings object	The mobile app settings information for the delete OS message action.
action-settings-send-os-command	action-settings object	The mobile app settings information for the send OS command action.
notifications-enabled	Boolean	True indicates that notifications from this console to the mobile app are enabled and will flow to any registered devices. False indicates that notifications will not flow from this console to any device with the mobile app.

The action-settings nested object contains the following fields:

Table 318. action-settings object properties		
Field name	Type	Description
enablement	String Enum	The value indicating the action enablement setting for the mobile app. Valid values: • "disabled" - The action is not allowed to be performed from the mobile app for this HMC, even if the global actions-enabled setting is true. • "enabled" - The action is allowed to be performed from the mobile app for this HMC, unless the global actions-enabled setting is false. • "enabled-for-users" - The action is allowed to be performed from the mobile app for this HMC if and only if the global actions-enabled setting is true and the user or pattern-based user's template is included in the enabled-users-and-templates property.
enabled-users-and-templates	Array of String/ URI	The canonical URI paths of Users and User Templates for which the action is enabled. This property is only included when the enablement property has a value of "enabled-for-users". When included, its value must always have at least one URI. Note that this property cannot contain users of type pattern-based.

Get Console Properties

The Get Console Properties operation retrieves the properties of the Console object.

HTTP method and URI

GET /api/console

Response body contents

On successful completion, the response body contains an object that provides the current values of the properties for the Console object as defined in [“Data model” on page 576](#). Field names and data types in the object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the Console object.

On successful execution, HTTP status code 200 (OK) is returned and all of the current properties as defined by the data model for the Console object are provided in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described [“Response body contents” on page 582](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console HTTP/1.1
x-api-session: duqt0x27j17a9sn50e9w2sh5f0nxitb56zbiqnc5yjwxrsvvz
```

Figure 313. Get Console Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 22:04:21 GMT
content-type: application/json; charset=UTF-8
content-length: 4398
{
  "class": "console",
  "description": "",
  "ec-mcl-description": {
    "ec": [
      {
        "description": "Hardware Management Console Framework",
        "mcl": [
          {
            "last-update": null,
            "level": "-",
            "type": "retrieved"
          },
          {
            "last-update": null,
            "level": "-",
            "type": "activated"
          },
          {
            "last-update": null,
            "level": "-",
            "type": "accepted"
          },
          {
            "last-update": null,
            "level": "-",
            "type": "installable-concurrent"
          },
          {
            "last-update": null,
            "level": "-",
            "type": "removable-concurrent"
          }
        ]
      },
      {
        "number": "P46683",
        "part-number": "02WG827",
        "type": "SYSTEM"
      }
    ]
  }
}
```

Figure 314. Get Console Properties: Response (Part 1)

```

{
  "description": "Hardware Management Console Platform Firmware",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46658",
  "part-number": "02WF276",
  "type": "HMCBIOS"
},
{
  "description": "Licensed Internal Code Alerts",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46684",
  "part-number": "02WG828",
  "type": "MALERT"
},

```

Figure 315. Get Console Properties: Response (Part 2)

```

{
  "description": "Enablement of new features  ",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46685",
  "part-number": "02WG829",
  "type": "ENABLE1"
},
{
  "description": "Enablement of new features  ",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46686",
  "part-number": "02WG830",
  "type": "ENABLE2"
},

```

Figure 316. Get Console Properties: Response (Part 3)

```

{
  "description": "Firmware feature enablement ",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46687",
  "part-number": "02WG831",
  "type": "FFE"
},
{
  "description": "Open source components",
  "mcl": [
    {
      "last-update": null,
      "level": "-",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "-",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46688",
  "part-number": "02WG832",
  "type": "OPENSRC"
},

```

Figure 317. Get Console Properties: Response (Part 4)


```

    {
      "description": "Embedded Operating System",
      "mcl": [
        {
          "last-update": null,
          "level": "-",
          "type": "retrieved"
        },
        {
          "last-update": null,
          "level": "-",
          "type": "activated"
        },
        {
          "last-update": null,
          "level": "-",
          "type": "accepted"
        },
        {
          "last-update": null,
          "level": "-",
          "type": "installable-concurrent"
        },
        {
          "last-update": null,
          "level": "-",
          "type": "removable-concurrent"
        }
      ],
      "number": "P45842",
      "part-number": "02WG826",
      "type": "OS"
    }
  ],
  "is-locked": false,
  "machine-info": {
    "machine-model": "TW2",
    "machine-serial": "DK5C004",
    "machine-type": "2461"
  },
  "name": "HMCDAILY03",
  "network-info": {
    "this-hmc": [
      {
        "domain-name": "local",
        "hmc-name": "HMCDAILY03",
        "interface-name": "eth0",
        "ipv4-address": [
          {
            "ip-address": "9.60.14.37",
            "subnet-mask": "255.255.255.0"
          }
        ],
        "ipv6-address": [
          {
            "ip-address": "fe80:0:0:0:210:6fff:fe0d:9759",
            "prefix-length": 64
          }
        ]
      },
      {
        "is-private": false,
        "mac": "00106f0d9759"
      }
    ],
  },

```

Figure 318. Get Console Properties: Response (Part 5)

```

    {
      "domain-name": "local",
      "hmc-name": "HMCDAILY03",
      "interface-name": "eth1",
      "ipv4-address": [
        {
          "ip-address": "0.0.0.0",
          "subnet-mask": "255.255.255.255"
        }
      ],
      "ipv6-address": [],
      "is-private": false,
      "mac": "00106f0d975a"
    },
    {
      "domain-name": "local",
      "hmc-name": "HMCDAILY03",
      "interface-name": "eth2",
      "ipv4-address": [
        {
          "ip-address": "0.0.0.0",
          "subnet-mask": "255.255.255.0"
        }
      ],
      "ipv6-address": [],
      "is-private": false,
      "mac": "00106f0d975b"
    },
    {
      "domain-name": "local",
      "hmc-name": "HMCDAILY03",
      "interface-name": "eth3",
      "ipv4-address": [
        {
          "ip-address": "0.0.0.0",
          "subnet-mask": "255.255.255.0"
        }
      ],
      "ipv6-address": [],
      "is-private": false,
      "mac": "00106f0d975c"
    }
  ],
  "object-id": "106ffdd7-04fa-376c-a059-6c69486bc57f",
  "object-uri": "/api/console",
  "parent": null,
  "version": "2.15.0"
}

```

Figure 319. Get Console Properties: Response (Part 6)

Restart Console

The Restart Console operation restarts the Hardware Management Console.

HTTP method and URI

POST /api/console/operations/restart

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
force	Boolean	Optional	Whether the restart operation is processed when users are connected (true) or not (false). The default is false.

Description

The Hardware Management Console is restarted.

By default, the restart does not occur if one or more users are currently connected to the Hardware Management Console. This can be overridden by use of the **force** field in the request body.

On success, HTTP status code 202 (Accepted) is returned.

Authorization

To use Restart Console, you must have the following:

- Action/Task permission to the **Power Off or Restart** task
- Remote Restart must be enabled on the Hardware Management Console.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	267	The operation is rejected, due to the presence of HMC users. Either wait until all HMC users have logged off or retry the request with the force field set to "true" .
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	269	This operation is currently blocked. The error message will contain information on the blocking application.
	270	The remote restart operation is not enabled on the HMC.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Shutdown Console

Shutdown Console powers off the Hardware Management Console.

HTTP method and URI

```
POST /api/console/operations/shutdown
```

Request body contents

A request body must be specified. It has the following fields:

Field name	Type	Rqd/Opt	Description
force	Boolean	Optional	Whether the shutdown operation is processed when users are connected (true) or not (false). The default is false.

Description

The Hardware Management Console is powered off.

By default, the shutdown does not occur if one or more users are currently connected to the Hardware Management Console. This can be overridden by use of the force field in the request body.

The action to shutdown the Hardware Management Console occurs asynchronously. If the request is accepted, HTTP status code 202 (Accepted) is returned to indicate that the request has been initiated. However, because this action results in the targeted Hardware Management Console becoming inactive and powered off at completion, it is not possible to track the completion of this request. Thus no response body containing an asynchronous job URI is provided, nor is a job completion notification generated upon completion.

Authorization

To use Shutdown Console, you must have the following:

- Action/task permission to the **Power Off or Restart** task.
- Remote Shutdown must be enabled on the Hardware Management Console.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned but no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	267	The operation is rejected, due to the presence of HMC users. Either wait until all HMC users have logged off or retry the request with the force field set to true.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	270	The remote restart operation is not enabled on the HMC.
	304	This operation is currently blocked. The error message will contain information on the blocking application.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/operations/shutdown HTTP/1.1/
x-api-session: 5dul8zvlwa5s83eobcukaf1vug3s3kgidkyk9e5c5acsekabs1
content-type: application/json
content-length: 16
{
  "force": false
}
```

Figure 320. Shutdown Console: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 01 Mar 2013 19:38:25 GMT

<No response body>
```

Figure 321. Shutdown Console: Response

Reorder User Patterns

The `Reorder User Patterns` operation changes the search order of the console's User Patterns used when a user logs on to the console.

HTTP method and URI

```
POST /api/console/operations/reorder-user-patterns
```

Request body contents

The request body is a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
user-pattern-uris	Array of String/URI	Required	Ordered list of User Pattern object element-uri property values. The order of these URIs in the array defines the new order of the User Patterns.

Description

This operation reorders the console's User Patterns.

On successful execution of this operation the User Patterns are reordered to match the order of their **element-uri** properties in the **user-pattern-uris** array in the request body.

The request body is validated against the schema described in “Request body contents” on page 591. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If a URI in the request body does not designate an existing User Pattern object, status code 404 (Not Found) is returned. The array in the request body must include each of the console's currently defined User Patterns and no others. In addition, the API user must have action/task permission to the Manage User Patterns task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirement

This operation has the following authorization requirements:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 319. Reorder User Patterns: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The array in the request body is missing an existing User Pattern, or it contains an entry that designates a User Pattern that is not one of the console's current User Patterns.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	2	A URI in the request body does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/operations/reorder-user-patterns HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 301
{
  "user-pattern-uris": [
    "/api/console/user-patterns/497bf4ec-1dbf-11e4-8ceb-1c6f65065a91",
    "/api/console/user-patterns/6d897292-3ceb-11e4-9e36-1c6f65065a91",
    "/api/console/user-patterns/e40b9ba6-48e0-11e4-82a1-1c6f65065a91",
    "/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91"
  ]
}
```

Figure 322. Reorder User Patterns: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 323. Reorder User Patterns: Response

Get Console Audit Log

The Get Console Audit Log operation returns the console audit log, filtered according to the query parameters, if specified.

HTTP method and URI

GET /api/console/operations/get-audit-log

Query Parameters

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
end-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON array of JSON objects returned using HTTP chunked transfer encoding. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See [Table 320 on page 593](#) for more information.

A log-entry-info object contains information about a single log entry and the event which caused the entry. Each log-entry-info object contains the following fields:

Table 320. log-entry-info object properties		
Name	Type	Description
event-time	Timestamp	The time when the event occurred.
event-id	String	The ID number for the event.
event-name	String (0-12)	The name for the event, or null or an empty string if none.
userid	String	The user ID of the HMC/SE user associated with the event, or null if there is no user associated with the event.
user-uri	String/ URI	The canonical URI path of the HMC/SE user associated with the event, or null if there is no user associated with the event.
event-message	String	The complete, formatted message for the event.
event-data-items	Array of objects	An array of event-data-item-info objects, one for each item of event data associated with the event-message for the event. If there are no event data items, an empty array is provided. The order of items in this array is semantically significant and matches the documentation for this event-id .
event-details	Array of objects	An array of event-details-info objects, one for each event detail associated with the event. If there are no event details, an empty array is provided.

An event-details-info object contains information about a single event detail. Each event-details-info object contains the following fields:

Table 321. event-details-info object properties		
Name	Type	Description
event-details-message	String	The complete, formatted details message.
event-details-data-items	Array of objects	An array of event-data-item-info objects, one for each item of event details data associated with the event-details-message for the event. If there are no event details data items, an empty array is provided.

An event-data-item-info object contains information about a single item of event data. Each event-data-item-info object contains the following fields:

Table 322. event-data-item-info object properties		
Name	Type	Description
data-item-number	Integer	The number for this data item. This is the 0-based index of this event-data-item-info object in the event-data-items or event-details-data-items array in which it is contained. This number identifies the substitution variable to which this data item corresponds in the documentation for the event identified by event-id .
data-item-type	String Enum	Identifies the data type of the data item in the event-data-item field. Possible values are: • "long" • "float" • "string"
data-item-value	Varies. See data-item-type description	The data item.

Description

This operation returns the console's audit log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the begin-time query parameter is specified, then any entries earlier than that time are omitted. If the end-time query parameter is specified, then any entries later than that time are omitted.

The API user must have action/task permission to the **Audit and Log Management** task; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the audit log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirements:

- Action/task permission to the **Audit and Log Management** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described [“Response body contents” on page 593](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/get-audit-log HTTP/1.1
x-api-session: 3ws0pztvnx61quwkvk4kcpjodybuqyo8q840j2adzw8y08fg1h
```

Figure 324. Get Console Audit Log: Request

```

200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 22:26:52 GMT
content-type: application/json; charset=ISO-8859-1
[
  {
    "event-data-items": [
      {
        "data-item-number": 0,
        "data-item-type": "string",
        "data-item-value": "9.60.92.181"
      },
      {
        "data-item-number": 1,
        "data-item-type": "string",
        "data-item-value": "403"
      },
      {
        "data-item-number": 2,
        "data-item-type": "string",
        "data-item-value": "Logon"
      },
      {
        "data-item-number": 3,
        "data-item-type": "string",
        "data-item-value": "SYSTEM"
      },
      {
        "data-item-number": 4,
        "data-item-type": "string",
        "data-item-value": "Login failure, bad credentials for userid 8675309"
      },
      {
        "data-item-number": 5,
        "data-item-type": "string",
        "data-item-value": "/api/sessions"
      }
    ],
    "event-details": [],
    "event-id": "6055",
    "event-message": "A web services client on 9.60.92.181 attempted an unauthorized (403)
      action \"Logon\" as SYSTEM: Login failure, bad credentials for userid 8675309
      (URI:/api/sessions)",
    "event-name": "WSAPI",
    "event-time": 1487964584530,
    "user-uri": null,
    "userid": null
  },
  {
    "event-data-items": [
      {
        "data-item-number": 0,
        "data-item-type": "string",
        "data-item-value": "9.60.92.175"
      },
      {
        "data-item-number": 1,
        "data-item-type": "string",
        "data-item-value": "403"
      },
      {
        "data-item-number": 2,
        "data-item-type": "string",
        "data-item-value": "Set Mobile App Preferences"
      },
      {
        "data-item-number": 3,
        "data-item-type": "string",
        "data-item-value": "pedebug"
      }
    ],

```

Figure 325. Get Console Audit Log: Response (Part 1)

```

    {
      "data-item-number":4,
      "data-item-type":"string",
      "data-item-value":"Console"
    },
    {
      "data-item-number":5,
      "data-item-type":"string",
      "data-item-value":"HMCDAILY03"
    },
    {
      "data-item-number":6,
      "data-item-type":"string",
      "data-item-value":"Task"
    },
    {
      "data-item-number":7,
      "data-item-type":"string",
      "data-item-value":"Mobile App Preferences"
    },
    {
      "data-item-number":8,
      "data-item-type":"string",
      "data-item-value":"/api/console/operations/set-mobile-app-preferences"
    }
  ],
  "event-details":[],
  "event-id":"6053",
  "event-message":"A web services client on 9.60.92.175 attempted an unauthorized (403)
action \"Set Mobile App Preferences\" as pedebug against the Console object named
\\\"HMCDAILY03\\\". User does not have permission to the Task named \\\"Mobile App Preferences\\\"
(URI:/api/console/operations/set-mobile-app-preferences)",
  "event-name":"WSAPI",
  "event-time":1488213109380,
  "user-uri":"/api/users/0540e45c-686a-11e6-852d-00106f0d5d80",
  "userid":"PEDEBUG"
},
{
  "event-data-items":[
    {
      "data-item-number":0,
      "data-item-type":"string",
      "data-item-value":"9.60.74.177"
    },
    {
      "data-item-number":1,
      "data-item-type":"string",
      "data-item-value":"403"
    },
    {
      "data-item-number":2,
      "data-item-type":"string",
      "data-item-value":"GET"
    },
    {
      "data-item-number":3,
      "data-item-type":"string",
      "data-item-value":"n/a"
    },
    {
      "data-item-number":4,
      "data-item-type":"string",
      "data-item-value":"Asserted cradentials did not map to a session"
    },
    {
      "data-item-number":5,
      "data-item-type":"string",
      "data-item-value":"/api/console/operations/get-mobile-app-preferences"
    }
  ]
},

```

Figure 326. Get Console Audit Log: Response (Part 2)

```

    "event-details":[],
    "event-id":"6055",
    "event-message":"A web services client on 9.60.74.177 attempted an unauthorized (403)
action \"GET\" as n/a: Asserted cradentials did not map to a session (URI:/api/console/
operations/get-mobile-app-preferences)",
    "event-name":"WSAPI",
    "event-time":1567620717940,
    "user-uri":null,
    "userid":null
  }
]

```

Figure 327. Get Console Audit Log: Response (Part 3)

Get Console Security Log

The Get Console Security Log operation returns the console security log, filtered according to the query parameters, if specified.

HTTP method and URI

GET /api/console/operations/get-security-log

Query Parameters

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
end-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON array of JSON objects returned using HTTP chunked transfer encoding. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See [Table 320 on page 593](#) for more information.

Description

This operation returns the console's security log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the begin-time query parameter is specified, then any entries earlier than that time are omitted. If the end-time query parameter is specified, then any entries later than that time are omitted.

The API user must have action/task permission to the **View Security Logs** task; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event

message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirement:

- Action/task permission to the **View Security Logs** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described [“Response body contents” on page 598](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/get-security-log HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 328. Get Console Security Log: Request

```

200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT
content-type: application/json;charset=ISO-8859-1
[
  {
    "event-data-items": [
      {
        "data-item-number": 0,
        "data-item-type": "string",
        "data-item-value": "acsadmin"
      },
      {
        "data-item-number": 1,
        "data-item-type": "string",
        "data-item-value": "Sx149"
      },
      {
        "data-item-number": 2,
        "data-item-type": "string",
        "data-item-value": "admin.my.company.com [1.2.3.4]"
      }
    ],
    "event-details": [],
    "event-id": "1941",
    "event-message": "User acsadmin has logged on to Web Services API session Sx149 from location admin.my.company.com [1.2.3.4]",
    "event-name": "WSA Logon",
    "event-time": "1412285249660",
    "user-uri": "/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
    "userid": "ACSADMIN"
  },
  {
    "event-data-items": [
      {
        "data-item-number": 0,
        "data-item-type": "string",
        "data-item-value": "Dept Admin"
      }
    ],
    "event-details": [
      {
        "event-details-data-items": [],
        "event-details-message": "Task Role: Dept Admin --Based on role is null. Permitted tasks:"
      }
    ],
    "event-id": "1272",
    "event-message": "The task role Dept Admin has been created.",
    "event-name": "LOGTROLEADD",
    "event-time": "1412285252280",
    "user-uri": "/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
    "userid": "ACSADMIN"
  },

```

Figure 329. Get Console Security Log: Response (Part 1)

```

{
  "event-data-items":[
    {
      "data-item-number":0,
      "data-item-type":"string",
      "data-item-value":"Dept Admin"
    }
  ],
  "event-details":[
    {
      "event-details-data-items":[],
      "event-details-message":"Task Role: Dept Admin --Based on role is
        null. Permitted tasks: ClassId=XVirtualServer"
    }
  ],
  "event-id":"1273",
  "event-message":"The task role Dept Admin has been changed.",
  "event-name":"LOGTROLECHG",
  "event-time":1412285253000,
  "user-uri":"/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
  "userid":"ACSAADMIN"
}
]

```

Figure 330. Get Console Security Log: Response (Part 2)

List Console Hardware Messages

The List Console Hardware Messages operation lists the current set of hardware messages associated with the console.

HTTP method and URI

GET /api/console/hardware-messages

Query Parameters

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created earlier than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
end-time	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created later than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
hardware-messages	Array of hardware - message-info objects	Array of nested hardware-message-info objects as defined in the next table.

Each nested hardware-message-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	The canonical URI path of the hardware message. The URI is in the following form: <code>/api/console/hardware-messages/{hardware-message-id}</code>
timestamp	Timestamp	The date and time the hardware message was created
text	String	The text of the hardware message.

Description

This operation returns a set of console hardware messages in increasing timestamp order, filtered according to the query parameters, if specified.

If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

If there are no hardware messages associated with the console, or if no hardware messages are to be included in the results due to filtering, an empty array is returned and the operation completes successfully.

The API user must have Action/Task permission to the Hardware Messages task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 601](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 323. List Console Hardware Messages: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The begin-time value is greater than the end-time value.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task or the Hardware Messages task in view-only mode.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/console/hardware-messages HTTP/1.1
x-api-session: 4pw14919jtzcwohdfe8s9gw5zzv7v73yksomswrg50t7ni4q8r
```

Figure 331. List Console Hardware Messages: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Oct 2014 17:02:37 GMT
content-type: application/json; charset=UTF-8
content-length: 206
{
  "hardware-messages": [
    {
      "element-uri": "/api/console/hardware-messages/11c5f16a-4d58-11e4-ba8d-02215e673710",
      "text": "Licensed internal code has detected a problem. [Problem # 3]",
      "timestamp": 1412600137860
    }
  ]
}
```

Figure 332. List Console Hardware Messages: Response

Get Console Hardware Message Properties

The Get Console Hardware Message Properties operation retrieves the properties of a single console hardware message.

HTTP method and URI

```
GET /api/console/hardware-messages/{hardware-message-id}
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message to be retrieved.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the console hardware message object as defined in [“Data model”](#) on page 576. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation retrieves the properties of a single console hardware message specified by *{hardware-message-id}*.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 603](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task or the Hardware Messages task in view-only mode.
404 (Not Found)	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7 HTTP/1.1
x-api-session: 38v3uicf04oxah52q8spe8evm4esyrr83andorav2rv2xe6flq9
```

Figure 333. Get Console Hardware Message Properties: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:38:29 GMT
content-type: application/json; charset=UTF-8
content-length: 821
{
  "class": "hardware-message",
  "details": {
    "corrective-action": [
      "Service is required."
    ],
    "created": 1583173801902,
    "description": [
      "The Hardware Management Console licensed internal code has detected a problem."
    ],
    "lir-node-data": null,
    "problem-data": [
      {
        "caption": "System name",
        "value": "Local"
      },
      {
        "caption": "Date",
        "value": "Mar 2, 2020"
      },
      {
        "caption": "Time",
        "value": "1:30:01 PM"
      }
    ],
    "repair-impact": [
      "The S/390 microprocessor cluster will continue operating, but some Hardware Management Console functions may not be available."
    ],
    "type": "common-problem"
  },
  "element-id": "ef3d8ce0-5cb3-11ea-b607-fa163e63cff7",
  "element-uri": "/api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7",
  "parent": "/api/console",
  "service-supported": true,
  "text": "Licensed internal code has detected a problem. [Problem # 3]",
  "timestamp": 1583173847091
}

```

Figure 334. Get Console Hardware Message Properties: Response

Delete Console Hardware Message

The Delete Console Hardware Message operation deletes a single console hardware message

HTTP method and URI

```
DELETE /api/console/hardware-messages/{hardware-message-id}
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message to be deleted.

Description

This operation deletes a specific console hardware message. The hardware message to be deleted is identified by the *{hardware-message-id}* variable in the URI.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/console/hardware-messages/6b2d61a4-a1ac-11e4-87ee-5ef3fcae8020 HTTP/1.1
x-api-session: c8un3odpy8yyp150o3poz1ud4gwyfodlwyq495327bpyn2p0z
```

Figure 335. Delete Console Hardware Message: Request

```
204 No Content
date: Mon, 09 Feb 2015 20:07:31 GMT
server: zSeries management console API web server / 2.0

<No response body>
```

Figure 336. Delete Console Hardware Message: Response

Request Console Service

The Request Console Service operation electronically transmits problem information to request service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

```
POST /api/console/hardware-messages/{hardware-message-id}/operations/request-service
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message for which to request service.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
customer-name	String	Optional	The name of the person that can be contacted about the problem.
customer-phone	String	Optional	The telephone number of the person that can be contacted about the problem.

Description

This operation electronically requests service for the problem reported by this hardware message. Customer contact information may optionally be provided in the request body. Upon successful completion, the service request is prepared and queued for transmission, and this hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. Remotely requesting service must also be enabled and configured on the HMC; otherwise, status code 409 (Conflict) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <i>{hardware-message-id}</i> designates a console hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing console hardware message.
409 (Conflict)	341	The HMC is not enabled and configured for remotely requesting service.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7/operations/
  request-service HTTP/1.1
x-api-session: fvnsnbxibi13lmzk7poonx8upx515zbq4qyz0ghf8z7yxiev4
content-type: application/json
content-length: 56
{
  "customer-name": "Jenny",
  "customer-phone": "867-5309"
}
```

Figure 337. Request Console Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:46:47 GMT

<No response body>
```

Figure 338. Request Console Service: Response

Get Console Service Request Information

The Get Console Service Request Information operation returns problem information and a telephone number to be used for requesting service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

```
GET /api/console/hardware-messages/{hardware-message-id}/operations/get-service-information
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message for which to request service.

Response body contents

On successful completion, the response body contains a JSON object with the following fields.

Name	Type	Description
service-phone	String	The telephone number to call for service.
machine-model	String (1-3)	The model of the machine where the problem occurred.
machine-type	String (1-4)	The type of the machine where the problem occurred.
machine-serial-number	String (1-12)	The serial number of the machine where the problem occurred.
problem-type	String	The type of the problem to be reported to service.
problem-number	Integer	The identifying number of the problem to be reported to service.
problem-data	String	Additional problem data to be reported to service.
reference-code	String	The problem reference code to be reported to service.

Description

This operation is used to manually request service for the problem reported by this hardware message. This may be used if remote service is not configured or not functioning to call a service representative directly and provide the problem details. Upon successful completion, problem details to be reported to service are returned, and this hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 608](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <i>{hardware-message-id}</i> designates a console hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/hardware-messages/7ed7e20e-5d7e-11ea-b607-fa163e63cff7/operations/  
get-service-information HTTP/1.1  
x-api-session: 35wpq9m1ab0yxi9pvx27n673khzeahtt9c2v6fz2vsrf9g3ex0
```

Figure 339. Get Console Service Request Information: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:50:25 GMT
content-type: application/json; charset=UTF-8
content-length: 210
{
  "machine-model": "T01",
  "machine-serial-number": "000020026EA8",
  "machine-type": "8561",
  "problem-data": "",
  "problem-number": 4,
  "problem-type": "2",
  "reference-code": "E5D43206-BD254E4E",
  "service-phone": "1-800-IBM-SERV"
}

```

Figure 340. Get Console Service Request Information: Response

Decline Console Service

The Decline Console Service operation declines service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

POST `/api/console/hardware-messages/{hardware-message-id}/operations/decline-service`

In this request, the URI variable `{hardware-message-id}` is the unique identifier of the hardware message for which to decline service.

Description

This operation is used to decline service for the problem reported by this hardware message. Upon successful completion, the hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <code>{hardware-message-id}</code> designates a console hardware message that does not support service.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/hardware-messages/54efcda6-5cb3-11ea-b607-fa163e63cff7/operations/
  decline-service HTTP/1.1
x-api-session: 4ya8pci1s5u4jt0g9dvjilt7bpwcpb29nhkgnt2c63uan87upp
content-type: application/json
```

Figure 341. Decline Console Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:53:05 GMT

<No response body>
```

Figure 342. Decline Console Service: Response

List Unmanaged CPCs

The List Unmanaged CPCs operation lists the CPCs that have been discovered by this HMC but are not configured to be managed by this HMC.

HTTP method and URI

```
GET /api/console/operations/list-unmanaged-cpcs
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
cpcs	Array of cpc-info objects	Array of nested cpc-info objects as described in the next table.

Each nested cpc-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The canonical URI path for an unmanaged CPC object is of the form <code>/api/cpcs/{cpc-id}</code> where <code>{cpc-id}</code> is the unique identifier for the unmanaged CPC instance.
name	String	The name of the unmanaged CPC.

Description

This operation lists the CPCs that have been discovered by this HMC but are not configured to be managed by this HMC. Some basic information is provided for each CPC that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those unmanaged CPCs whose names match the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

An unmanaged CPC is included in the list only if the API user has object-access permission to that object. If there is an unmanaged CPC to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no unmanaged CPCs known to the HMC or if no unmanaged CPCs are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the unmanaged CPC objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 611](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 324. List Unmanaged CPCs: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage Notes

The APIs provide only minimal support for unmanaged CPC objects. There is no data model, and most of the standard operations are not provided. The `List Unmanaged CPCs` operation can be used to list them and to provide their name and object URI. The provided support also allows an API client to manage access to them through the `Add Permission to User Role` and `Remove Permission from User Role` operations. The HMC's **Add Object Definition** task is used to configure an unmanaged CPC to be managed by the HMC.

Example HTTP interaction

```
GET /api/console/operations/list-unmanaged-cpcs HTTP/1.1
x-api-session: 606ay5h8erhjme80hl7j4rglorqebifiirqnrd4m1ga9xyjv6
```

Figure 343. List Unmanaged CPCs: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 14:13:05 GMT
content-type: application/json; charset=UTF-8
content-length: 252
{
  "cpcs": [
    {
      "name": "P0LXSM20",
      "object-uri": "/api/cpcs/343a56e1-96ed-3191-b092-99a784311e43"
    },
    {
      "name": "D20BUSE",
      "object-uri": "/api/cpcs/07cc8420-78e6-3520-840a-ea6ff0074bbc"
    },
    {
      "name": "S202B",
      "object-uri": "/api/cpcs/3275e681-fe9b-3c54-ade8-6a08b802f781"
    }
  ]
}
```

Figure 344. List Unmanaged CPCs: Response

Get Mobile App Preferences

The Get Mobile App Preferences operation provides the current preferences for the mobile app on this console.

HTTP method and URI

```
GET /api/console/operations/get-mobile-app-preferences
```

Response body contents

On successful completion, the response body contains a JSON object that provides the **mobile-app-preferences** property of the Console object as defined in [“Data model” on page 576](#).

Description

This operation returns the **mobile-app-preferences** property of the Console object.

On successful execution, the **mobile-app-preferences** property is provided in the response body, and HTTP status code 200 (OK) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 613](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/get-mobile-app-preferences HTTP/1.1
x-api-session: 3thw38fnty3jtl5i8ym05yadmg8nth2hbn2j00hvf8y3xeondv
```

Figure 345. Get Mobile App Preferences: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 22 May 2019 20:04:41 GMT
content-type: application/json; charset=UTF-8
content-length: 930
{
  "mobile-app-preferences": {
    "action-settings-activate-partition": {
      "enablement": "enabled"
    },
    "action-settings-change-activation-profile": {
      "enablement": "enabled"
    },
    "action-settings-change-partition-processors": {
      "enablement": "enabled"
    },
    "action-settings-change-partition-weight": {
      "enablement": "enabled"
    },
    "action-settings-deactivate-partition": {
      "enablement": "enabled"
    },
    "action-settings-delete-hardware-message": {
      "enablement": "enabled"
    },
    "action-settings-delete-os-message": {
      "enablement": "enabled"
    },
    "action-settings-load-os-into-partition": {
      "enablement": "enabled"
    },
    "action-settings-reset-partition": {
      "enablement": "enabled"
    },
    "action-settings-send-os-command": {
      "enablement": "enabled"
    },
    "action-settings-start-partition": {
      "enablement": "enabled"
    },
    "action-settings-stop-partition": {
      "enablement": "enabled"
    },
    "actions-enabled": true,
    "app-enabled": true,
    "notifications-enabled": true,
    "password-caching-enabled": true,
    "require-app-password-enabled": false
  }
}

```

Figure 346. Get Mobile App Preferences: Response

Set Mobile App Preferences

The Set Mobile App Preferences operation sets one or more of the mobile app preferences for this console.

HTTP method and URI

POST /api/console/operations/set-mobile-app-preferences

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
app-enabled	Boolean	Optional	True indicates that the mobile app is enabled for this console. False indicates that the mobile app is disabled and will not be allowed to access this console.
require-app-password-enabled	Boolean	Optional	True indicates that the mobile app must require its user to set an app password in order to access the app itself. False indicates the user is not required to secure the app with a password.
password-caching-enabled	Boolean	Optional	True indicates that the mobile app is permitted to securely store the user's password on the device for this console. False indicates this is not permitted, and the user must input the password on every login to this console from the app.
actions-enabled	Boolean	Optional	True indicates that the mobile app is permitted to perform actions against this console and the objects it manages, as permitted by the user's authority. False indicates that no actions are permitted from the mobile app to this console, and the app will be used only for monitoring.
action-settings-activate-partition	action-settings object	Optional	The mobile app settings information for the activate logical partition action.
action-settings-deactivate-partition	action-settings object	Optional	The mobile app settings information for the deactivate logical partition action.
action-settings-start-partition	action-settings object	Optional	The mobile app settings information for the start DPM partition action.
action-settings-stop-partition	action-settings object	Optional	The mobile app settings information for the stop DPM partition action.
action-settings-change-activation-profile	action-settings object	Optional	The mobile app settings information for the change activation profile of a logical partition action.
action-settings-load-os-into-partition	action-settings object	Optional	The mobile app settings information for the load OS into logical partition action.
action-settings-reset-partition	action-settings object	Optional	The mobile app settings information for the reset logical partition action.
action-settings-change-partition-weight	action-settings object	Optional	The mobile app settings information for the change DPM partition or logical partition weights and capping action.
action-settings-change-partition-processors	action-settings object	Optional	The mobile app settings information for the change DPM partition or logical partition processors action.

Name	Type	Rqd/Opt	Description
action-settings-delete-hardware-message	action-settings object	Optional	The mobile app settings information for the delete hardware message action.
action-settings-delete-os-message	action-settings object	Optional	The mobile app settings information for the delete OS message action.
action-settings-send-os-command	action-settings object	Optional	The mobile app settings information for the send OS command action.
notifications-enabled	Boolean	Optional	True indicates that notifications from this console to the mobile app are enabled and will flow to any registered devices. False indicates that notifications will not flow from this console to any device with the mobile app.

Description

This operation sets one or more mobile app preferences for this console.

On successful execution, the preferences passed in the request body have been set on this console, and HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **HMC Mobile Settings** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	337	One or more of the User URIs specified in an action-settings object in the request body is of type pattern-based .
403 (Forbidden)	1	The user does not have access to the HMC Mobile Settings task.
404 (Not Found)	2	One or more of the User or User Template URIs specified in the action-settings object in the request body is not known to this console, or the user does not have permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/operations/set-mobile-app-preferences HTTP/1.1
x-api-session: 5aekjhn8t6g2xlc3xery2flewwey1xwxlnsm0ixhn00p81i1da
content-type: application/json
content-length: 81
{
  "actions-enabled":true,
  "app-enabled":true,
  "password-caching-enabled":false
}
```

Figure 347. Set Mobile App Preferences: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 18 Nov 2016 20:40:53 GMT

<No response body>
```

Figure 348. Set Mobile App Preferences: Response

Get CPC Notification Preferences for Device

The Get CPC Notification Preferences for Device operation retrieves the configured mobile app notification preferences for a CPC to a mobile device.

HTTP method and URI

```
POST /api/console/operations/get-device-cpc-notification-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
device-id	String	Required	The unique identifier of the mobile device to which the preferences being retrieved belong.
cpc-uri	String/ URI	Required	The object-uri property of the CPC for which the notification preferences apply.

Response body contents

The response body is a JSON object with the following fields:

Name	Type	Description
new-hardware-message	Boolean	True indicates that this console is configured to notify the identified device of new hardware messages for the identified CPC. False indicates the device with not be notified. The default value is false until modified with the Update CPC Notification Preferences for Device operation.

Name	Type	Description
cpc-acceptable-status-change	Boolean	True indicates that this console is configured to notify the identified device when the has-unacceptable-status property changes for the identified CPC. False indicates the device will not be notified. The default value is false until modified with the Update CPC Notification Preferences for Device operation.
new-os-message	Array of String/ URI	Array of object-uri values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which new operating system message events should notify the identified device. Any new OS message on the identified partitions will cause the identified device to be notified. To limit which OS messages lead to notifications, refer to new-os-message-filtered. Note that this list of partitions is mutually exclusive with those identified in new-os-message-filtered. The default value is empty until modified with the Update CPC Notification Preferences for Device operation.
new-os-message-filtered	Array of partition-os-message-filter objects	Array of partition-os-message-filter objects defining the Partitions (for DPM) or Logical Partitions (for non-DPM) and the filter conditions for which new operating system events should notify the identified device. As opposed to the new-os-message property, which notifies the device of all OS messages on an identified partition, this allows notification for only OS messages that pass the filter conditions. Note that this list of partitions is mutually exclusive with those listed in new-os-message. The default value is empty until modified with the Update CPC Notification Preferences for Device operation.
partition-acceptable-status-change	Array of String/ URI	Array of object-uri values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which the has-unacceptable-status property changes should notify the identified device. The default value is empty until modified with the Update CPC Notification Preferences for Device operation.

The partition-os-message-filter nested object contains the following fields:

Name	Type	Description
partition-uri	String/ URI	The object-uri of the Partition (for DPM) or Logical Partition (for non-DPM) object for which new operating system message events should notify the identified device when they pass the defined filters.

Name	Type	Description
filters	Array of os-message-filter objects	Array of os-message-filter objects to be applied to new operating system messages on the identified partition. Each filter is considered individually against new OS messages, but all conditions specified within each os-message-filter must be met in order for the filter to be considered true. In other words, the logical operation for evaluating a single filter object's conditions is AND, whereas all filter objects are evaluated together with OR.

The os-message-filter nested object contains the following fields:

Name	Type	Description
priority	Boolean	Filter to select messages based on their priority. True indicates that a new operating system message passes this filter if its is-priority property is set to true. False indicates that it passes this filter if is-priority is false. This field is omitted if priority is not to be considered by this filter.
held	Boolean	Filter to select messages based on whether they require a response. True indicates that a new operating system message passes this filter if its is-held property is set to true. False indicates that it passes this filter if is-held is false. This field is omitted if a response requirement is not to be considered by this filter.
message-text	String	Filter to select messages based on their message text in order to be notified when an OS emits specific matching messages. It must be a valid Java™ regular expression. To pass this filter, the string regular expression must match the operating system message's message-text property. This field is omitted if message text is not to be considered by this filter.

Description

This operation retrieves the mobile app notification preferences of a CPC for a device from this console. This specifies what notifications are currently being delivered to the mobile app on the identified device for the CPC. Note that a **device-id** may be passed for which an Update CPC Notification Preferences for Device operation has not been performed. In this case, default values will be returned, as defined in the [“Response body contents” on page 618](#).

On successful execution, the preference properties are provided in the response body, and HTTP status code 200 (OK) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 618](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	2	The requested cpc-uri is not known to this console.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request body is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/operations/get-device-cpc-notification-preferences HTTP/1.1
x-api-session: 64rvm5qub8e2viadxuitl1xfustyh4w19j2pcebb1hbyfe0pm
content-type: application/json
content-length: 120
{
  "cpc-uri": "/api/cpcs/9724bf69-038b-3152-841a-3ceb8ee21515",
  "device-id": "dXgLFgPZGMB06nPRB03nFinn17SW:ugACKp8kZfDgR"
}
```

Figure 349. Get CPC Notification Preferences for Device: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 07 Dec 2017 15:21:52 GMT
content-type: application/json; charset=UTF-8
content-length: 467
{
  "cpc-acceptable-status-change": false,
  "new-hardware-message": false,
  "new-os-message": [
    "/api/logical-partitions/1b7d74fd-6de7-3642-93a6-6c3a1a7488b6"
  ],
  "new-os-message-filtered": [
    {
      "filters": [
        {
          "held": true
        },
        {
          "priority": true
        }
      ],
      "partition-uri": "/api/logical-partitions/26d3ade6-3cc9-3841-bcc0-f9f2d8b46f0a"
    }
  ],
  "partition-acceptable-status-change": [
    "/api/logical-partitions/1b7d74fd-6de7-3642-93a6-6c3a1a7488b6",
    "/api/logical-partitions/26d3ade6-3cc9-3841-bcc0-f9f2d8b46f0a"
  ]
}
```

Figure 350. Get CPC Notification Preferences for Device: Response

Update CPC Notification Preferences for Device

The Update CPC Notification Preferences for Device operation updates the configured mobile app notification preferences for a CPC to a mobile device.

HTTP method and URI

POST /api/console/operations/update-device-cpc-notification-preferences

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
device-id	String	Required	The unique identifier of the mobile device to which the preferences being updated belong.
cpc-uri	String/ URI	Required	The object-uri property of the CPC for which the notification preferences apply.
new-hardware-message	Boolean	Optional	True indicates that this console will now be configured to notify the identified device of new hardware messages for the identified CPC. False indicates the device will not be notified. Note that passing true requires object-access permission to the identified CPC.
cpc-acceptable-status-change	Boolean	Optional	True indicates that this console will now be configured to notify the identified device of has-unacceptable-status property changes for the identified CPC. False indicates the device will not be notified. Not that passing true requires object-access permission to the identified CPC.
new-os-message	Array of String/ URI	Optional	Array of object-uri values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which new operating system message events will notify the identified device. Any new OS messages on the identified partitions will cause the identified device to be notified. To limit which OS messages lead to notifications, refer to new-os-message-filtered . Note that this list of partitions is mutually exclusive with those identified in new-os-message-filtered . Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed partition object must have the parent identified by the cpc-uri .

Name	Type	Rqd/Opt	Description
new-os-message-filtered	Array of partition-os-message-filter objects	Optional	Array of partition-os-message-filter objects defining the Partitions (for DPM) or Logical Partitions (for non-DPM) and the filter conditions for which new operating system events will notify the identified device. As opposed to the new-os-message property which notifies the device of all OS messages on an identified partition, this allows notification for only OS messages that pass the filter conditions. Note that this list of partitions is mutually exclusive with those identified in new-os-message. Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed Partition object must have the parent identified by the cpc-uri.
partition-acceptable-status-change	Array of String/ URI	Optional	Array of object-uri values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which has-unacceptable-status property changes will notify the identified device. Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed partition object must have the parent identified by the cpc-uri.

The partition-os-message-filter nested object contains the following fields:

Name	Type	Rqd/Opt	Description
partition-uri	String/ URI	Required	The object-uri of the Partition (for DPM) or Logical Partition (for non-DPM) object for which new operating system message events should notify the identified device when they pass the defined filters.
filters	Array of os-message-filter objects	Required	Array of os-message-filter objects to be applied to new operating system messages on the identified partition. Each filter is considered individually against new OS messages, but all conditions specified within each os-message-filter must be met in order for the filter to be considered true. In other words, the logical operation for evaluating a single filter object's conditions is AND, whereas all filter objects are evaluated together with OR. Note that the array must be non-empty.

The os-message-filter nested object contains the following fields:

Name	Type	Rqd/Opt	Description
priority	Boolean	Optional	Filter to select messages based on their priority. True indicates that a new operating system message passes this filter if its is-priority property is set to true. False indicates that it passes this filter if is-priority is false. Omit this field if priority is not to be considered by this filter. Note that at least one filter field is required.
held	Boolean	Optional	Filter to select messages based on whether they require a response. True indicates that a new operating system message passes this filter if its is-held property is set to true. False indicates that it passes this filter if is-held is false. Omit this field if a response requirement is not to be considered by this filter. Note that at least one filter field is required.
message-text	String	Optional	Filter to select messages based on their message text in order to be notified when an OS emits specific matching messages. It must be a valid Java regular expression. To pass this filter, the string regular expression must match the operating system message's message-text property. Omit this field if message text is not to be considered by this filter. If specified, the string must be non-empty. Note that at least one filter field is required.

Description

This operation updates the mobile app notification preferences of a CPC for a device from this console. This specifies what notifications are to be delivered to the mobile app on the identified device for the CPC.

Note that object-access permission to the CPC is required only to turn on the **new-hardware-message** or **cpc-acceptable-status-change** notification properties. For any **object-uri** included in the **new-os-message** array or **partition-acceptable-status-change** array, only object-access permission to that partition or logical partition is required.

On successful execution, the notification preference properties are updated for the identified device and CPC from this console, and HTTP status code 204 (No Content) is returned.

Authorization requirements

Object-access permission is required to the CPC to enable notifications for its hardware messages and acceptable status changes. Object-access permission is required to a logical partition or partition to enable notifications for its operating system messages or acceptable status changes.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	2	The requested cpc-uri is not known to this console, or one of the CPC-specific notification preferences was asked to be enabled but the user does not have object-access permission to the CPC.
	335	One or more of the requested logical partition or partition URIs is not known to the identified CPC and/or to this console, or the user does not have object-access permission to it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request body is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/operations/update-device-cpc-notification-preferences HTTP/1.1
x-api-session: 2u80csmk000d1mjpgvfvavmhodr71eb2ilocsvbv6weng0ma8
content-type: application/json
content-length: 318
{
  "cpc-acceptable-status-change":false,
  "cpc-uri":"/api/cpcs/ded1343f-c248-3fc0-afb4-8e54a2ea9647",
  "device-id":"f99e434d-6a38-3487-b331",
  "new-hardware-message":true,
  "new-os-message":[
    "/api/logical-partitions/5709de82-63ee-370f-9928-3a1331332acc",
    "/api/logical-partitions/5989e5ec-b820-37ae-b5f7-81c476a96885"
  ]
}
```

Figure 351. Update CPC Notification Preferences for Device: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 18 Nov 2016 21:12:06 GMT

<No response body>
```

Figure 352. Update CPC Notification Preferences for Device: Response

Inventory service data

Information about the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Console objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included.

For each Console object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Console](#)

Properties” on page 582. That is, the data provided is the same as would be provided if a Get Console Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a single console. This object would appear as one array entry in the response array:

```
{
  "class": "console",
  "description": "Endicott Test HMC",
  "ec-mcl-description": {
    "ec": [
      {
        "description": "Hardware Management Console Framework",
        "mcl": [
          {
            "last-update": 1422026097000,
            "level": "39",
            "type": "retrieved"
          },
          {
            "last-update": 1423074018000,
            "level": "39",
            "type": "activated"
          },
          {
            "last-update": null,
            "level": "000",
            "type": "accepted"
          },
          {
            "last-update": null,
            "level": "39",
            "type": "installable-concurrent"
          },
          {
            "last-update": null,
            "level": "1",
            "type": "removable-concurrent"
          }
        ]
      },
      {
        "number": "N98841",
        "part-number": "00LY737",
        "type": "SYSTEM"
      }
    ]
  },
}
```

Figure 353. Console object: Sample inventory data (Part 1)


```

    {
      "description": "Enablement of new features  ",
      "mcl": [
        {
          "last-update": null,
          "level": "000",
          "type": "retrieved"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "activated"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "accepted"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "installable-concurrent"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "removable-concurrent"
        }
      ],
      "number": "N98844",
      "part-number": "00LY740",
      "type": "ENABLE1"
    }
  ],
}

```

Figure 354. Console object: Sample inventory data (Part 2)

```

"ip-swapping-available": true,
"is-auto-switch-enabled": true,
"is-locked": false,
"machine-info": {
  "machine-model": "PBC",
  "machine-serial": "KQ0N5RF",
  "machine-type": "7382"
},
"name": "ZFXHMC2",
"network-info": {
  "paired-hmc": {
    "hmc-name": "ZFXHMC1",
    "ipv4-address": [
      "9.60.15.110",
      "9.60.14.110"
    ],
    "ipv6-address": [
      "2002:93c:ffb:1:5ef3:fcff:feaf:deb1",
      "fdd8:673b:d89b:1:5ef3:fcff:feaf:deb1",
      "fe80:0:0:0:5ef3:fcff:feaf:deb1"
    ]
  },
  "this-hmc": [
    {
      "domain-name": "",
      "hmc-name": "ZFXHMC2",
      "interface-name": "eth0",
      "ipv4-address": [
        {
          "ip-address": "9.60.15.111",
          "subnet-mask": "255.255.255.0"
        }
      ],
      "ipv6-address": [
        {
          "ip-address": "2002:93c:ffb:1:5ef3:fcff:feae:8019",
          "prefix-length": 64
        },
        {
          "ip-address": "fdd8:673b:d89b:1:5ef3:fcff:feae:8019",
          "prefix-length": 64
        },
        {
          "ip-address": "fe80:0:0:0:5ef3:fcff:feae:8019",
          "prefix-length": 64
        }
      ],
      "is-private": false,
      "mac": "5CF3FCAE8019"
    }
  ],
}

```

Figure 355. Console object: Sample inventory data (Part 3)

```

    {
      "domain-name": "",
      "hmc-name": "ZFXHMC2",
      "interface-name": "eth1",
      "ipv4-address": [
        {
          "ip-address": "9.60.14.111",
          "subnet-mask": "255.255.255.0"
        }
      ],
      "ipv6-address": [
        {
          "ip-address": "fe80:0:0:0:5ef3:fcff:feae:801a",
          "prefix-length": 64
        }
      ],
      "is-private": false,
      "mac": "5CF3FCAE801A"
    }
  ],
  "object-id": "ec982d6c-bcc1-3ae8-b39c-a2efd14734b4",
  "object-uri": "/api/console",
  "paired-role": "primary",
  "parent": null,
  "version": "2.13.0"
}

```

Figure 356. Console object: Sample inventory data (Part 4)

User-related-access permission

An HMC user has access to certain information about their own user account. This information is contained in their User object and related objects. An API user's access permission to their own User object and specific related objects is known as user-related-access permission. Through such permission, those objects will be included in a List <class> operation, unless otherwise filtered out, and the API user is permitted to issue a Get <class> Properties operation on them, unless specifically prohibited.

The object types included in user-related-access permission are:

- User
- User Role
- User Pattern
- Password Rule
- LDAP Server Definition
- MFA Server Definition

User-related-access permission includes the following:

- Permission for an object of the above types to be included in the response body of a List <class> operation.
- Permission to view properties of objects of the following types through the Get <class> Properties and Get Inventory operations:
 - User
 - User Role
 - User Pattern
 - Password Rule
- Permission to update certain properties of the User object. See the [“Data model” on page 630](#) or the Update User Properties operation for details.

User object

A User object represents a single Hardware Management Console user. There are different types of console users. A typical customer-defined user is known as a standard user. A user template defines certain attributes of a group of users whose user IDs match the expression in a User Pattern; these definitions are known as template users. When a user logs on with a user ID that matches the expression in a User Pattern, a pattern-based user is created. There are certain user definitions supplied by the system; they are known as system-defined users.

All API users are permitted to see their own User object in a `List Users` response, issue `Get User Properties` for their own User object and, with the exception of pattern-based users, issue `Update User Properties` to alter certain properties of their own User object. An API user with action/task permission to the `Manage Users` task is permitted to view and change any standard or system-defined User object. An API user with action/task permission to the `Manage User Templates` task is permitted to view and change any template User object.

User objects may be replicated to this HMC via its Data Replication facility. If that is the case, the `Update User Properties`, `Add User Role to User`, `Remove User Role from User` and `Delete User` operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

System-defined users

In most respects, system-defined users are indistinguishable from standard users. They can be modified or even deleted. Most properties of system-defined users may be changed, but certain others are immutable; the immutable properties are denoted as such in the data model section that follows. While system-defined users can be deleted and their name reused for a standard user definition, that practice is discouraged due to the likely confusion such a situation would cause. The typical system-defined users include the following:

- ACSADMIN
- ADVANCED
- OPERATOR
- SERVICE
- SYSPROG
- STORAGEADMIN

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 70, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 68.

The following class-specific specializations apply to the other base managed object properties:

Table 325. User object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the User object is of the form <code>/api/users/{user-id}</code> , where <code>{user-id}</code> is the value of the object-id property (not the name property) of the User object.
parent	—	String/ URI	The canonical URI path of the Console object.

Table 325. User object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
class	—	String	The class of a User object is "user" .
name	(ro)	String	If type is not "template": the name (console user ID) of the User object. It must be 4-320 characters in length and consist only of alphanumeric characters and the following special characters: "@<+:#='&*();-/\\,%_>.". This name must be unique among all users whose type is not "template" defined on the console. While preexisting names may contain a double quote (") character, it is not permitted in new user names. If type is "template": the name of the template. While preexisting template names are virtually unrestricted in terms of length and characters, new template names must conform to the length and character requirement of the name property described in the "Base managed object properties schema" on page 70. This name must be unique among all template definitions on the console. For the purpose of verifying uniqueness only, this name is treated in a case-insensitive fashion when used to create a new User object of any type.
description	(w)(pc)	String (0-1024)	The description of the User object. Default: an empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Note: Some properties are only valid for users of a specific type. Such properties are only included in the User object if the user is of that type, as indicated by its **type** property. For example, a user with a **type** of **"standard"** includes the **disabled** property but not the **user-pattern-uri** property.

Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **password-rule-uri** is **null** when the **authentication-type** value is **"ldap"**.

Table 326. User object: class specific additional properties

Name	Qualifier	Type	Description
type	—	String Enum	The type of user. Supported values are: • "standard" - a standard, normal user. • "template" - a user template. • "pattern-based" - a user created dynamically from a User Pattern and its associated template. • "system-defined" - a user supplied by the system. Certain properties of system-defined users are immutable.
user-pattern-uri	—	String/ URI	The canonical URI path of the User Pattern object upon which this user is based. Prerequisite: type is "pattern-based"

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
user-template-uri	—	String/ URI	The canonical URI path of the User Template object upon which this user is based. Prerequisite: type is "pattern-based"
disabled	(w)(pc)	Boolean	Indicates whether the user is currently disabled. When disabled, the user is prevented from logging on to the console via either the UI or the Web Services APIs. Prerequisite: type is not "template" . Default: false
authentication-type	(w)(pc)	String Enum	The type of user ID and password authentication used for this user, which must be one of the following: • "local" - the console performs the authentication. • "ldap" - authentication is delegated to the LDAP server identified in the ldap-server-definition-uri property. If type is "template" , this must be "ldap" . Note: The value of this property is a prerequisite for certain other properties. Changing this value requires certain properties to be included in the same request; see the Update User Properties operation for details.
password-rule-uri	(w)(pc)	String/ URI	The canonical URI path of the Password Rule for this user. Prerequisite: authentication-type is "local" .
password	(wo)(pc)	String	The console logon password for this user. The specific length, character and other requirements on this password are controlled by the authentication type and Password Rule assigned to this user. Note the (wo) qualifier; this field may be altered via an API, but it is not included in the response when this object's properties are retrieved via an API.
password-expires	—	Integer	The time interval, in days, until the user's current password expires. A value of 0 indicates that the password will expire within the next 24 hours. A value of -1 indicates that the HMC does not enforce password expiration for this user; however, if this user is authenticated with an external authentication mechanism (e.g. LDAP) such expiration might be enforced by that mechanism.
force-password-change	(w)(pc)	Boolean	Indicates whether the user should be forced to change their console logon password the next time they log in. Prerequisite: authentication-type is "local" Default: true

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
ldap-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the configuration object for the LDAP server used for authentication of this user. Prerequisite: authentication-type is "ldap".
userid-on-ldap-server	(w)(pc)	String (0-32)	The user ID for this user on the LDAP server identified in ldap-server-definition-uri , or null if the user's console user ID (value of the name property) should be used. See the LDAP Server Definition object for more information on how this property is used. Prerequisite: authentication-type is "ldap" and type is not "template". Default: an empty string
session-timeout	(w)(pc)	Integer (0-525600)	The session timeout in minutes for this user. This is the interval over which a user's UI session can run before being prompted for identity verification. 0 indicates no timeout. Default: 0
verify-timeout	(w)(pc)	Integer (0-525600)	The verification timeout in minutes for this user. This is the amount of time allowed for the user to re-enter their password after being prompted due to a session timeout (see the session-timeout property). 0 indicates no timeout. Default: 15
idle-timeout	(w)(pc)	Integer (0-525600)	The idle timeout in minutes for this user. This is the amount of time the user's UI session can be idle before it is disconnected. 0 indicates no timeout. Default: 0
min-pw-change-time	(w)(pc)	Integer (0-525600)	The minimum password change time in minutes for this user. This is the minimum amount of time that must elapse between changes to this user's password. 0 indicates no minimum; that is, the password can be changed immediately after it has just been changed. Prerequisite: authentication-type is "local". Default: 0
max-failed-logins	(w)(pc)	Integer (0-525600)	The maximum number of failed login attempts for this user. This is maximum number of consecutive failed login attempts before the user is temporarily disabled for the amount of time specified in the disable-delay property. 0 indicates that the user is never disabled due to failed login attempts. Default: 3

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
disable-delay	(w)(pc)	Integer (0-525600)	The time in minutes that the user is disabled after exceeding the maximum number of failed login attempts specified in the max-failed-logins property. 0 indicates that the user is not disabled for any period of time after reaching the maximum number of invalid login attempts. Default: 1
inactivity-timeout	(w)(pc)	Integer (0-525600)	The inactivity timeout in days for this user. This is the maximum number of days of inactivity (consecutive days with no login) before the user is disabled. 0 indicates no timeout. Default: 0
disruptive-pw-required	(w)(pc)	Boolean	Indicates whether the user's password is required to perform disruptive actions via the UI. Default: true
disruptive-text-required	(w)(pc)	Boolean	Indicates whether text input is required to perform disruptive actions via the UI. Default: false
allow-remote-access	(w)(pc)	Boolean	Indicates whether the user is allowed to access the HMC via its remote web server interface Default: false
allow-management-interfaces	(w)(pc)	Boolean	Indicates whether the user is allowed access to management interfaces. This includes access to the Web Services APIs. Default: false
max-web-services-api-sessions	(w)(pc)	Integer (0-9999)	The maximum number of simultaneous Web Services API sessions the user is permitted to have. Default: 100
web-services-api-session-idle-timeout	(w)(pc)	Integer (1-360)	The idle timeout in minutes for Web Services API sessions created by this user. This is the amount of time a Web Services API session can be idle before it is terminated. Default: 360
user-roles	(c)(pc)	Array of String/ URI	The list of user roles defined for this user. Each element in this array is a canonical URI path for a User Role object. The roles provided in this list can change as a result of the Add User Role to User and Remove User Role from User operations. This property is immutable if type is " system-defined ".

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
default-group-uri	(w)(pc)	String/ URI	The canonical URI path of the user's default group or null if the user has no default group. Managed objects created by this user automatically become members of this group. The user must have object-access permission to this group. This must be a user-defined group to which the user has object-access permission. API users are permitted to change their own default group designation via the Update User Properties operation. Default: null
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.
multi-factor-authentication-required	(w)(pc)	Boolean	Indicates whether the user is required to use the HMC's built-in MFA support. If true, the user is required to enter their current TOTP multi-factor authentication code (time-based one-time password) in addition to their logon password during UI and API logons. Setting this to true will cause mfa-types to be set to a one-element array containing "hmc-totp". Setting this to false will cause mfa-types to be set to null if "hmc-totp" is present. Default: false
force-shared-secret-key-change	(w)(pc)	Boolean	Indicates whether the user is required to establish a new shared secret key during the next logon. The shared secret key is used to calculate the user's current multi-factor authentication code, which is required during logon. Prerequisite: multi-factor-authentication-required is true Default: false
email-address	(w)(pc)	String (0-254)	The user's email address or null if the user has no email address. This email address must roughly adhere to Internet Engineering Task Force (IETF) RFC 822.

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
mfa-types	(w)(pc)	Array of String Enum	Identifies the types of multi-factor authentication (MFA) the user is required to use when logging onto the HMC, or null if MFA is not required. When setting this property, the API client program is responsible for keeping it and multi-factor-authentication-required consistent. Each element of this array must be unique and must be one of the following: • "hmc-totp" - Time-based one-time password validated by the HMC. This is equivalent to setting multi-factor-authentication-required to true. If present in the array, it must be the only element of the array. • "mfa-server" - Additional factors validated by an MFA server. Default: null
primary-mfa-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the MFA Server Definition object for the primary MFA server used to authenticate the user. Prerequisite: mfa-types contains "mfa-server" Default: null
backup-mfa-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the MFA Server Definition object for the backup MFA server used to authenticate the user, or null if there is no backup server. Must specify a different MFA server than the primary MFA server. Prerequisite: mfa-types contains "mfa-server" Default: null
mfa-policy	(w)(pc)	String (1-64)	The name of the MFA policy, such as a RACF® Policy, that applies to the user when an MFA server authenticates the user. It must identify a policy whose only MFA factor is the RSA SecurID factor. Prerequisite: mfa-types contains "mfa-server" Default: null
mfa-userid	(w)(pc)	String (1-64)	The MFA user ID. This is a user ID, such as a RACF user ID, that identifies this user to the MFA server that authenticates this user. For User objects with a type of "pattern-based", this property's default value may be overridden by the LDAP attribute identified by mfa-userid-override. Prerequisite: type is not "template", and mfa-types contains "mfa-server" Default: same value as name property

Table 326. User object: class specific additional properties (continued)

Name	Qualifier	Type	Description
mfa-userid-override	(w)(pc)	String (1-256)	The name of the LDAP attribute that contains the MFA user ID, such as a RACF user ID, that identifies the user to the MFA server that authenticates the user, or null if there is no such attribute. This can be used to override the value of the mfa-userid property during authentication. If the named LDAP attribute does not exist in a user's directory entry, or it exists but is empty, then the user's MFA user ID is not altered. Prerequisite: type is "template", and mfa-types contains "mfa-server" Default: null

List Users

The **List Users** operation lists standard, template and system-defined users defined to the console. With one very specific exception, pattern-based users are never included in the response to the **List Users** operation.

HTTP method and URI

GET /api/console/users

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid type property value, with the exception of " pattern-based ".

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
users	Array of objects	Array of nested user-info objects as described in the next table.

Each nested user-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The object-uri property of the User object.
name	String	The name property of the User object.
type	String Enum	The type property of the User object.

Description

The **List Users** operation lists users defined to the console. Some basic properties are provided for each user.

If the **name** query parameter is specified, the returned list is limited to those users that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid user **type** property value for this operation. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those users that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A user is included in the list only if the API user has sufficient access permission to that object. The access permission requirements for User objects vary depending on the type of User object. All API users have user-related-access permission to their own User object. Action/task permission to the **Manage Users** task includes access permission to all non-template Users, and action/task permission to the **Manage User Templates** task includes access permission to all template users. If there is a User object to which the API user does not have permission, that object is omitted from the list, but no error status code results.

A pattern-based user is only included in the response if it meets the filtering criteria and is the User object for the API user.

If there are no users defined to the console or if no users are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User object included in the response body, or, depending on the type of User object, action/task permission to the **Manage Users** task or the **Manage User Templates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 637](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 327. List Users: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/users?type=system-defined&name=.*ADMIN HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 357. List Users: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:30 GMT
content-type: application/json;charset=UTF-8
content-length: 229
{
  "users": [
    {
      "name": "ACSDADMIN",
      "object-uri": "/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
      "type": "system-defined"
    },
    {
      "name": "STORAGEADMIN",
      "object-uri": "/api/users/ae6bf048-3dc0-11e4-8dd1-1c6f65065a91",
      "type": "system-defined"
    }
  ]
}
```

Figure 358. List Users: Response

Get User Properties

The `Get User Properties` operation retrieves the properties of a single User object that is designated by its object ID. With one very specific exception, this operation does not support pattern-based users.

HTTP method and URI

```
GET /api/users/{user-id}
```

In this request, the URI variable `{user-id}` is either the object ID of the User object whose properties are to be returned or the keyword value **"this-user"** which designates the API user that issued the request.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User object as defined in the data model section. Field names and data types in the JSON object are the same as the property names and data types defined in the [“Data model”](#) on page 630.

Description

This operation returns the current properties of a single User object that is designated by `{user-id}`.

On successful execution, all of the current properties as defined in the data model for the User object, except those designated as write-only properties, are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User object and the API user must have access permission to it. All API users have user-related-access permission to their own User object. Action/task permission to the **Manage Users** task includes access permission to all non-template Users, and action/task permission to the **Manage User Templates** task includes access permission to all template users. If the URI path does

not designate an existing User object or the API user does not have access permission to it, status code 404 (Not Found) is returned.

This operation does not support pattern-based users, unless the target of the operation is the User object for the API user that issued the request.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User object specified in the request URI, or, depending on the type of User object specified in the request URI, action/task permission to the **Manage Users** task or the **Manage User Templates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 639](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 359. Get User Properties: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 21:27:30 GMT
content-type: application/json;charset=UTF-8
content-length: 1387
{
  "allow-management-interfaces":false,
  "allow-remote-access":false,
  "authentication-type":"local",
  "backup-mfa-server-definition-uri":null,
  "class":"user",
  "default-group-uri":null,
  "description":"Gabby McRosie - company president",
  "disable-delay":1,
  "disabled":false,
  "disruptive-pw-required":true,
  "disruptive-text-required":false,
  "email-address":"finn@example.com",
  "force-password-change":true,
  "force-shared-secret-key-change":false,
  "idle-timeout":0,
  "inactivity-timeout":0,
  "is-locked":false,
  "ldap-server-definition-uri":null,
  "max-failed-logins":3,
  "max-web-services-api-sessions":100,
  "mfa-policy":null,
  "mfa-types":[
    "hmc-totp"
  ],
  "mfa-userid":null,
  "mfa-userid-override":null,
  "min-pw-change-time":0,
  "multi-factor-authentication-required":true,
  "name":"Gabby",
  "object-id":"e9e8d20a-4a7a-11e4-91ee-1c6f65065a91",
  "object-uri":"/api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91",
  "parent":"/api/console",
  "password-expires":-1,
  "password-rule-uri":"/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
  "primary-mfa-server-definition-uri":null,
  "replication-overwrite-possible":false,
  "session-timeout":0,
  "type":"standard",
  "user-roles":[
    "/api/user-roles/ea6f9b14-4a7a-11e4-affa-1c6f65065a91",
    "/api/user-roles/ea41a664-4a7a-11e4-91ee-1c6f65065a91",
    "/api/user-roles/ea094df0-4a7a-11e4-8777-1c6f65065a91"
  ],
  "userid-on-ldap-server":null,
  "verify-timeout":15,
  "web-services-api-session-idle-timeout":360
}

```

Figure 360. Get User Properties: Response

Update User Properties

The Update User Properties operation updates the properties of a single User object that is designated by its object ID. This operation is not valid for pattern-based users.

HTTP method and URI

POST /api/users/{user-id}

In this request, the URI variable {user-id} is the object ID of the User object whose properties are to be updated or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the User object specified by *{user-id}*.

The URI path must designate an existing User object and the API user must have permission to update it. All API users have user-related-access permission to their own User object, which includes permission to update certain properties. Action/task permission to the **Manage Users** task includes update permission to all non-template users, and action/task permission to the **Manage User Templates** task includes update permission to all template users. If the URI path does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user does not have action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, an attempt to update any field other than the user's own **password** or **default-group-uri** field results in status code 403 (Forbidden), and specifying a group to which the API user does not have object-access permission in **default-group-uri** results in status code 404 (Not Found).

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) or 409 (Conflict) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., attempting to set **password-rule-uri** when the **authentication-type** value is **"ldap"**). An attempt to update a pattern-based user is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **authentication-type** is changed from **"ldap"** to **"local"**, and **force-password-change** is not defined in the request body, **force-password-change** will be defaulted to false). Note however that certain fields must be included in the request body if the request alters the **authentication-type** property, because they are required in order to perform the newly specified type of logon authentication. Specifically, changing **authentication-type** to **"local"** requires that the following properties also be specified in the same request: **password**, **password-rule-uri**. Changing **authentication-type** to **"ldap"** requires that the following property also be specified: **ldap-server-definition-uri**. Also, certain fields must be included in the request body if the request alters, directly or indirectly, the **mfa-types** property, because they are required in order to perform the newly specified type of multi-factor authentication. Specifically, adding **"mfa-server"** to **mfa-types** requires that the following properties also be specified in the same request: **primary-mfa-server-definition-uri**, **mfa-policy**.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- For a user to update their own **password** or **default-group-uri** property, user-related-access permission to the User object specified in the request URI or action/task permission to the **Manage Users** task is required.
- An API user with action/task permission to the **Manage Users** task or the **Manage User Templates** task, depending on the type of User object specified in the request URI, may update any writable property of that User object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 328. Update User Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	311	The new password does not conform to the requirements of the password policy in effect for this user.
	314	This operation is not supported for an object of this type. Pattern-based users may not be updated.
	330	The operation is not valid because the API user cannot disable their own user ID.
	337	The primary-mfa-server-definition-uri in the request body and the backup-mfa-server-definition-uri in the request body designate the same MFA Server Definition object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.
	323	The password-rule-uri field in the request body does not designate an existing Password Rule object.
	324	The ldap-server-definition-uri field in the request body does not designate an existing LDAP Server Definition object.
	325	The default-group-uri field in the request body does not designate an existing resource of the expected type, or designates a resource for which the user identified by the request URI does not have object-access permission.
	338	The primary-mfa-server-definition-uri in the request body does not designate an existing MFA Server Definition object.
	339	The backup-mfa-server-definiton-uri in the request body does not designate an existing MFA Server Definition object.
409 (Conflict)	8	The request body contains a field whose presence or value is inconsistent with the current state of the User object.
	321	The user's authentication-type property cannot be changed at this time. The specified user is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.
	337	The operation could not be completed because it would result in the primary-mfa-server-definition-uri and the backup-mfa-server-definition-uri designating the same MFA Server Definition object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 145
{
  "allow-management-interfaces":true,
  "description":"A new and improved description of this User",
  "web-services-api-session-idle-timeout":240
}
```

Figure 361. Update User Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:30 GMT

<No response body>
```

Figure 362. Update User Properties: Response

Add User Role to User

The Add User Role to User operation adds a specified User Role to a specified user. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

POST /api/users/{user-id}/operations/add-user-role

In this request, the URI variable *{user-id}* is the object ID of the user to which a User Role is to be added or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
user-role-uri	String/ URI	Required	The canonical URI path of the User Role to be added.

Description

This operation adds a User Role to a user.

On successful execution of this operation the User Role specified in the request body has been added to the user identified in the request URI.

The request body is validated against the schema described in [“Request body contents”](#) on page 644. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-

access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. If the request body does not designate an existing User Role, status code 404 (Not Found) is returned. If the specified object is already in the collection of the user's User Roles, status code 409 (Conflict) is returned. An attempt to update the User Role collection of a system-defined or pattern-based user is not valid and fails with status code 400 (Bad Request).

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to modify a standard user or the **Manage User Templates** task to modify a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 329. Add User Role to User: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. The User Role collection of system-defined and pattern-based users may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	315	The object designated by the URI in the request body is already in the user's collection of User Roles.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91/operations/add-user-role HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 73
{
  "user-role-uri": "/api/user-roles/eaecdf34-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 363. Add User Role to User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 364. Add User Role to User: Response

Remove User Role from User

The Remove User Role from User operation removes a specified User Role from a specified user. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

```
POST /api/users/{user-id}/operations/remove-user-role
```

In this request, the URI variable `{user-id}` is the object ID of the user from which a User Role is to be removed or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
user-role-uri	String/ URI	Required	The canonical URI path of the User Role to be removed.

Description

This operation removes a User Role from a user.

On successful execution of this operation the User Role specified in the request body has been removed from the user identified in the request URI.

The request body is validated against the schema described in “Request body contents” on page 646. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. If the specified object is not in the collection of the user's User Roles, status code 409 (Conflict) is returned. An

attempt to update the User Role collection of a system-defined or pattern-based user is not valid and fails with status code 400 (Bad Request).

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to modify a standard user or the **Manage User Templates** task to modify a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 330. Remove User Role from User: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. The User Role collection of system-defined and pattern-based users may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	316	The object designated by the URI in the request body is not in the user's collection of User Roles.
	321	The User Role cannot be removed at this time. The user is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.
	328	The User Role cannot be removed at this time, because doing so would leave the user without object-access permission to their default group.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91/operations/remove-user-role HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 73
{
  "user-role-uri": "/api/user-roles/eaecdf34-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 365. Remove User Role from User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 366. Remove User Role from User: Response

Create User

The Create User operation creates a standard or template User object with the given properties. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

```
POST /api/console/users
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the user's name property. Note that the length and character requirements that apply to this field are dependent on the value of the type field.
description	String	Optional	The value to be set as the user's description property.
type	String Enum	Required	The value to be set as the user's type property. Must be "standard" or "template" .
disabled	Boolean	Optional	The value to be set as the user's disabled property.
authentication-type	String Enum	Required	The value to be set as the user's authentication-type property.
password-rule-uri	String/URI	Required if authentication-type is "local"	The value to be set as the user's password-rule-uri property.
password	String	Required if authentication-type is "local"	The value to be set as the user's password property.

Field name	Type	Rqd/Opt	Description
force-password-change	Boolean	Optional	The value to be set as the user's force-password-change property.
ldap-server-definition-uri	String/ URI	Required if authentication-type is "ldap"	The value to be set as the user's ldap-server-definition-uri property.
userid-on-ldap-server	String	Optional	The value to be set as the user's userid-on-ldap-server property.
session-timeout	Integer	Optional	The value to be set as the user's session-timeout property.
verify-timeout	Integer	Optional	The value to be set as the user's verify-timeout property.
idle-timeout	Integer	Optional	The value to be set as the user's idle-timeout property.
min-pw-change-time	Integer	Optional	The value to be set as the user's min-pw-change-time property.
max-failed-logins	Integer	Optional	The value to be set as the user's max-failed-logins property.
disable-delay	Integer	Optional	The value to be set as the user's disable-delay property.
inactivity-timeout	Integer	Optional	The value to be set as the user's inactivity-timeout property.
disruptive-pw-required	Boolean	Optional	The value to be set as the users disruptive-pw-required property.
disruptive-text-required	Boolean	Optional	The value to be set as the user's disruptive-text-required property.
allow-remote-access	Boolean	Optional	The value to be set as the user's allow-remote-access property.
allow-management-interfaces	Boolean	Optional	The value to be set as the user's allow-management-interfaces property.
max-web-services-api-sessions	Integer	Optional	The value to be set as the user's max-web-services-api-sessions property.
web-services-api-session-idle-timeout	Integer	Optional	The value to be set as the user's web-services-api-session-idle-timeout property.
multi-factor-authentication-required	Boolean	Optional	The value to be set as the user's multi-factor-authentication-required property.
mfa-types	Array of String Enum	Optional	The value to be set as the user's mfa-types property.

Field name	Type	Rqd/Opt	Description
primary-mfa-server-definition-uri	String/URI	Required if mfa-types contains "mfa-server"	The value to be set as the user's primary-mfa-server-definition-uri property.
backup-mfa-server-definition-uri	String/URI	Optional	The value to be set as the user's backup-mfa-server-definition-uri property.
mfa-policy	String	Required if mfa-types contains "mfa-server"	The value to be set as the user's mfa-policy property.
mfa-userid	String	Optional	The value to be set as the user's mfa-userid property.
mfa-userid-override	String	Optional	The value to be set as the user's mfa-userid-override property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
object-uri	String/URI	Canonical URI path of the new User object.

Description

This operation creates a new console user.

On successful execution of this operation the user is created using the inputs as specified by the request body. The URI of the new user is provided in the response body and in a **Location** response header as well. An Inventory Change notification is emitted asynchronously.

The request body is validated against the schema described in the “Request body contents” on page 648. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **password-rule-uri** when the **authentication-type** value is "ldap") or the specified name is not unique. If a URI in the request body does not designate an existing resource of the appropriate type, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Users** task to create a standard user or the **Manage User Templates** task to create a template user; otherwise, status code 403 (Forbidden) is returned.

Certain user names are used internally by the Hardware Management Console and are therefore not available for use when creating a new user. An attempt to create a user with one of these names results in status code 400 (Bad Request) indicating that there is already a user with that name. The list of such names is case-insensitive and includes the following:

- SOOACADMIN
- SOOADVANCED
- SOOENSADMIN
- SOOENSOPERATOR
- SOOOPERATOR
- SOOSERVICE
- SOOSTORAGEADMIN

- SOOSYSPROG
- PEDEBUG

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to create a standard user or the **Manage User Templates** task to create a template user.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 650, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 331. Create User: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A user with the name specified in the request body already exists.
	311	The password does not conform to the requirements of the password policy in effect for this user.
	337	The primary-mfa-server-definition-uri in the request body and the backup-mfa-server-definition-uri in the request body designate the same MFA Server Definition object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	323	The password-rule-uri field in the request body does not designate an existing Password Rule object.
	324	The ldap-server-definition-uri field in the request body does not designate an existing LDAP Server Definition object.
	338	The primary-mfa-server-definition-uri in the request body does not designate an existing MFA Server Definition object.
	339	The backup-mfa-server-definiton-uri in the request body does not designate an existing MFA Server Definition object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/users HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 234
{
  "authentication-type":"local",
  "description":"Gabby McRosie - company president",
  "name":"Gabby",
  "password":"abc123pw",
  "password-rule-uri":"/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
  "type":"standard"
}
```

Figure 367. Create User: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:29 GMT
content-type: application/json; charset=UTF-8
content-length: 64
{
  "object-uri":"/api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91"
}
```

Figure 368. Create User: Response

Delete User

The Delete User operation deletes a User object designated by its object ID. This operation is not valid for pattern-based users.

HTTP method and URI

```
DELETE /api/users/{user-id}
```

In this request, the URI variable *{user-id}* is the object ID of the User object to be deleted.

Description

This operation removes a specified user from the console. The user is identified by the *{user-id}* variable in the URI.

Upon successfully removing the user, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously.

The URI path must designate an existing User object; otherwise, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. It is an error for an API user to attempt to delete his own User object; any attempt to do so results in status code 400 (Bad Request). If the request URI identifies a template user, and a user or a User Pattern refers to that template user, the request fails and status code 409 (Conflict) is returned. An attempt to delete a pattern-based user is not valid and fails with status code 400 (Bad Request).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to delete a non-template user, or the **Manage User Templates** task to delete a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 332. Delete User: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	312	This operation is not supported for an object of this type. Pattern-based users may not be deleted.
	313	The request URI designates the API user's own User object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users or User Patterns refer to this template user.
	320	The object cannot be deleted at this time. It is currently identified as the Automatic Logon ID for the Hardware Management Console.
	321	The object cannot be deleted at this time. It is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 369. Delete User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 370. Delete User: Response

Inventory service data

Information about the users managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"user"** are to be included. An entry for a particular user is included only if the API user has access permission to that object as described in the Get User Properties operation.

For each User object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the Get User Properties operation. That is, the data provided is the same as would be provided if a Get User Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a single user. This object would appear as one array entry in the response array:

```

{
  "allow-management-interfaces": false,
  "allow-remote-access": false,
  "authentication-type": "ldap",
  "class": "user",
  "default-group-uri": "/api/groups/fc400fc6-54e3-335d-854f-a0a6d10f000b",
  "description": "System administrator",
  "disable-delay": 1,
  "disabled": false,
  "disruptive-pw-required": true,
  "disruptive-text-required": false,
  "email-address": "finn@example.com",
  "force-password-change": null,
  "force-shared-secret-key-change": false,
  "idle-timeout": 0,
  "inactivity-timeout": 0,
  "is-locked": false,
  "ldap-server-definition-uri": "/api/console/ldap-server-definitions/4927787e-34c4-11e4-a1ea-5ef3fcae8020",
  "max-failed-logins": 3,
  "max-web-services-api-sessions": 100,
  "min-pw-change-time": null,
  "multi-factor-authentication-required": true,
  "name": "sysadmin",
  "object-id": "9069f7a6-34c5-11e4-af4d-5ef3fcae8020",
  "object-uri": "/api/users/9069f7a6-34c5-11e4-af4d-5ef3fcae8020",
  "parent": "/api/console",
  "password-expires": -1,
  "password-rule-uri": null,
  "replication-overwrite-possible": false,
  "session-timeout": 0,
  "type": "standard",
  "user-roles": [
    "/api/user-roles/b39afb87-d915-4070-a22f-91b158c6c01e"
  ],
  "userid-on-ldap-server": "sysadmin",
  "verify-timeout": 15,
  "web-services-api-session-idle-timeout": 15
}

```

Figure 371. User object: Sample inventory data

User Role object

A User Role object represents an authority role which can be assigned to one or more console users. A role may allow access to specific managed objects, classes of managed objects, groups and/or tasks. There are two types of User Roles: user-defined and system-defined. User-defined User Roles are created by a console user, whereas the system-defined User Roles are pre-defined, standard User Roles supplied with the console.

User role objects may be replicated to this HMC via its Data Replication facility. If that is the case, the Update User Role Properties, Add Permission to User Role, Remove Permission from User Role and Delete User Role operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

Through user-related-access permission described in [“User-related-access permission”](#) on page 629, API users are permitted to see certain User Role objects in a List User Roles response and issue Get User Role Properties for those User Role objects. An API user with action/task permission to the Manage User Roles task is permitted to view any User Role object and change any user-defined User Role object.

System-defined user roles

There are many system-defined User Roles supplied with the console. System-defined roles may not be modified or deleted. The names assigned to the system-defined roles are enumerated in [Appendix B, “Enum values for the User Role object,”](#) on page 975.

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 70, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 68.

The following class-specific specializations apply to the other base managed object properties:

Table 333. User Role object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the User Role object is of the form <code>/api/user-roles/{user-role-id}</code> , where <code>{user-role-id}</code> is the value of the object-id property of the User Role object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a User Role object is "user-role" .
name	(ro)	String	The name of the User Role object. This name must be unique among all of the console's User Roles with the same type value. While pre-existing User Role names are virtually unrestricted in terms of length and characters, new User Role names must conform to the length and character requirements of the name property described in the “Base managed object properties schema” on page 70. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new User Role object. The names of system-defined User Roles are listed in Appendix B, “Enum values for the User Role object,” on page 975.
description	(w)(pc)	String (0-1024)	The description of the User Role object. Default: an empty string

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 334. User Role object: class specific additional properties			
Name	Qualifier	Type	Description
type	—	String Enum	The type of User Role. Supported values are: • "user-defined" • "system-defined"

Table 334. User Role object: class specific additional properties (continued)

Name	Qualifier	Type	Description
associated-system-defined-user-role-uri	(w)(pc)	String/ URI	Canonical URI path of the system-defined User Role associated with this User Role. The User Roles which are valid associated User Roles are identified in Appendix B, “Enum values for the User Role object,” on page 975. If this is a system-defined User Role, this property is null, because system-defined User Roles do not have an associated User Role. Default: the URI path of the Operator Tasks system-defined User Role.
permissions	(c)(pc)	Array of objects	The list of permissions included in this User Role. These permissions are identified by permission-info objects, as defined in the next table. The members provided in this list can change as a result of the Add Permission to User Role and Remove Permission from User Role operations. If there are no permissions in this User Role, an empty array is provided. Default: an empty array
is-inheritance-enabled	(w)(pc)	Boolean	Indicates whether User Role inheritance is enabled. When true , if this User Role permits access to a parent managed object, then all managed objects that are hosted by the parent managed object are also permitted by this role. When false , no such inherited permissions exist. Default: false .
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

Each nested permission-info object contains the following fields:

Table 335. permission-info object properties

Name	Type	Description
permitted-object	String/ URI or String Enum	Canonical URI path or String Enum which identifies the object(s) or task to which permission has been granted. Granting permission to a Task object gives the user action/task permission to the console task associated with that Task object, and the user is permitted to target any of their authorized objects with that task. If this identifies a class of managed objects, a specific managed object or a Group object, then permission is granted to view the details of the object(s) and target the object(s) with any authorized task for which it is an appropriate target. This field is one of the following: • The identifier for a class of managed objects. This identifier is the String Enum value for the class from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI <code>"/api/system-manual-definition"</code>, which denotes a specific managed object known on the HMC UI as "System Manual Definition"¹ The type of object(s) is indicated by the permitted-object-type property.
permitted-object-type	String Enum	Identifies the type of object(s) identified by the permitted-object property. • "object" - permitted-object contains a URI that identifies one of the following: – A specific managed object – A Group object – A Task object. • "object-class" - permitted-object contains a String Enum value from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973.
include-members	Boolean	Indicates whether the members of a group are included in the User Role, or if only the group itself is included. True if members are included; false otherwise. Prerequisite: the permitted-object field identifies a Group object that does not use pattern-matching.
view-only-mode	Boolean	Indicates whether it is a task's view-only version that is in this User Role. Only certain tasks support a view-only mode. This field is only provided if the permitted-object field identifies such a Task object. A User Role cannot have both the view-only version and the non-view-only version in its set of permissions. Prerequisite: the permitted-object field identifies a Task object that supports a view-only mode.

Table 335. permission-info object properties (continued)

Name	Type	Description
¹ The object identified by this special URI does not have full API support. It is only valid as a permitted object identifier in a User Role object. Thus it may be included in the Get User Role Properties response body, and it is valid in the request body of the Add Permission to User Role and Remove Permission from User Role operations.		

List User Roles

The List User Roles operation lists User Roles defined to the console.

HTTP method and URI

GET /api/console/user-roles

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid type property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
user-roles	Array of objects	Array of nested user-role-info objects as described in the next table.

Each nested user-role-info object contains the following fields:

Field name	Type	Description
object-uri	String/URI	The object-uri property of the User Role object.
name	String	The name property of the User Role object.
type	String Enum	The type property of the User Role object.

Description

The List User Roles operation lists User Roles defined to the console. Some basic properties are provided for each user role.

If the **name** query parameter is specified, the returned list is limited to those User Roles that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid User Role **type** property value. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those User Roles that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A User Role is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage User Roles** task. If there is a User Role to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no User Roles defined to the console or if no User Roles are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Role objects included in the response body or action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 659](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage Notes

While it is intended that system-defined User Roles have a name that begins with "hmc-", there is no such guarantee. API clients are cautioned to use the **type** property rather than **name** to reliably distinguish between system-defined and user-defined User Roles.

Example HTTP interaction

```
GET /api/console/user-roles?name=.*admin.*&type=system-defined HTTP/1.1
x-api-session: 40hwdiopt6avosjzkk4v20sac85xtd842klcjh9hox3wfh8f89
```

Figure 372. List User Roles: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 21:12:08 GMT
content-type: application/json; charset=UTF-8
content-length: 556
{
  "user-roles": [
    {
      "name": "hmc-access-administrator-tasks",
      "object-uri": "/api/user-roles/11182f6f-f24a-4aea-b18b-a2b2be46bdf1",
      "type": "system-defined"
    },
    {
      "name": "hmc-energy-administrator-tasks",
      "object-uri": "/api/user-roles/c4888502-72c7-4507-9c1b-9df9a24128ae",
      "type": "system-defined"
    },
    {
      "name": "hmc-storage-administrator-objects",
      "object-uri": "/api/user-roles/6ef9ab39-4195-4035-a5e0-00e91944352a",
      "type": "system-defined"
    },
    {
      "name": "hmc-storage-administrator-tasks",
      "object-uri": "/api/user-roles/9f197c15-6eb4-4b1c-94d9-b38ffa5861ab",
      "type": "system-defined"
    }
  ]
}

```

Figure 373. List User Roles: Response

Get User Role Properties

The Get User Role Properties operation retrieves the properties of a single User Role object that is designated by its object ID.

HTTP method and URI

GET /api/user-roles/{user-role-id}

In this request, the URI variable {user-role-id} is the object ID of the User Role object whose properties are to be retrieved.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User Role object as defined in the data model section. Field names and data types in the JSON object are the same as the property names and data types defined in the [“Data model”](#) on page 656.

Description

This operation returns the current properties of a single User Role object that is designated by {user-role-id}.

On successful execution, all of the current properties as defined in the data model for the User Role object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User Role object and the API user must have user-related-access permission to it or action/task permission to the **Manage User Roles** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Role object specified in the request URI, or action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 661](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage Notes

While it is intended that system-defined User Roles have a name that begins with "hmc-", there is no such guarantee. API clients are cautioned to use the **type** property rather than **name** to reliably distinguish between system-defined and user-defined User Roles.

Example HTTP interaction

```
GET /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 374. Get User Role Properties: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT
content-type: application/json; charset=UTF-8
content-length: 847
{
  "associated-se-user-role-uri": null,
  "class": "user-role",
  "description": "Role for managing department business",
  "is-inheritance-enabled": false,
  "is-locked": false,
  "name": "Dept Admin",
  "object-id": "eb53f840-4a7a-11e4-affa-1c6f65065a91",
  "object-uri": "/api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91",
  "parent": "/api/console",
  "permissions": [
    {
      "permitted-object": "/api/console/tasks/4d5a39f0-c1df-4a2e-9a46-5bf6f6a759f3",
      "permitted-object-type": "object",
      "view-only-mode": false
    },
    {
      "include-members": true,
      "permitted-object": "/api/groups/cafb3a9b-6a34-4475-938f-98c5d60868a5",
      "permitted-object-type": "object"
    },
    {
      "permitted-object": "lpar-image",
      "permitted-object-type": "object-class"
    },
    {
      "permitted-object": "/api/console/tasks/45042443-7202-40b3-8630-b7a563a21d8d",
      "permitted-object-type": "object"
    }
  ],
  "replication-overwrite-possible": false,
  "type": "user-defined"
}

```

Figure 375. Get User Role Properties: Response

Update User Role Properties

The Update User Role Properties operation updates the properties of a single user-defined User Role object that is designated by its object ID. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

POST /api/user-roles/{user-role-id}

In this request, the URI variable {user-role-id} is the object ID of the User Role object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the User Role object specified by {user-role-id}.

The URI path must designate an existing User Role object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. An attempt to update a system-defined User Role is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 336. Update User Role Properties: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be updated.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 99
{
  "description": "A new and improved description of this User Role",
  "is-inheritance-enabled": true
}
```

Figure 376. Update User Role Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 377. Update User Role Properties: Response

Add Permission to User Role

The Add Permission to User Role operation adds a specified permission to a specified user-defined User Role thereby granting that permission to all users that have that User Role. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/user-roles/{user-role-id}/operations/add-permission
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role to which a permission is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
permitted-object	String/ URI or String Enum	Required	Canonical URI path or String Enum which identifies the object(s) or task to which permission is to be granted. See the data model for more information about permissions. This field is one of the following: • The identifier for a class of managed objects. This identifier is the String Enum value for the class from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI <code>/api/system-manual-definition</code>, which denotes a specific managed object known on the HMC UI as "System Manual Definition" The type of object(s) is indicated by the permitted-object-type field.
permitted-object-type	String Enum	Required	Identifies the type of object(s) identified by the permitted-object field. Supported values are: • "object" - permitted-object contains a URI that identifies one of the following: – A specific managed object – A Group object – A Task object. • "object-class" - permitted-object contains a String Enum value from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973.
include-members	Boolean	Optional	Indicates whether the members of the group are included in the User Role, or if only the group itself is included. True if members are included; false otherwise. Prerequisite: the permitted-object field identifies a Group object that does not use pattern-matching. Default: false
view-only-mode	Boolean	Optional	Indicates whether it is the task's view-only version that is being added to this User Role. Only certain tasks support a view-only mode. This field is only allowed if the permitted-object field identifies such a Task object. Prerequisite: the permitted-object field identifies a Task object that supports a view-only mode. Default: true

Description

This operation adds a permission to a User Role

On successful execution of this operation the permission specified in the request body has been added to the User Role identified in the request URI.

The request body is validated against the schema described in the [“Request body contents”](#) on page 665. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User Role object, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. If the URI in the request body does not designate an existing resource, status code 404 (Not Found) is returned. An attempt to alter a system-defined User Role is not valid and fails with status code 400 (Bad Request).

If the specified permission is already in the User Role, or an attempt is made to have both the view-only and non-view-only versions of a Task in the User Role, status code 409 (Conflict) is returned.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 337. Add Permission to User Role: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	315	The permission identified in the request body is already in the User Role.
	327	The view-only and non-view-only versions of a Task cannot be in the same User Role. An attempt was made to add one version when the other was already in the User Role's set of permitted objects.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91/operations/  
  add-permission HTTP/1.1  
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c  
content-type: application/json  
content-length: 114  
{  
  "permitted-object":"/api/console/tasks/5d8c9f60-a2b3-4327-9fd4-791df8a60dcc",  
  "permitted-object-type":"object"  
}
```

Figure 378. Add Permission to User Role: Request

```
204 No Content  
server: zSeries management console API web server / 2.0  
cache-control: no-cache  
date: Thu, 02 Oct 2014 21:27:32 GMT  
  
<No response body>
```

Figure 379. Add Permission to User Role: Response

Remove Permission from User Role

The Remove Permission from User Role operation removes a specified permission from a specified user-defined User Role. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/user-roles/{user-role-id}/operations/remove-permission
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role from which a permission is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
permitted-object	String/ URI or String Enum	Required	Canonical URI path or String Enum which identifies the object(s) or task to which permission is to be removed. See the data model for more information about permissions. This field is one of the following: • The identifier for a class of managed object. This identifier is the String Enum value for the class from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI <code>"/api/system-manual-definition"</code>, which denotes a specific managed object known on the HMC UI as "System Manual Definition" The type of object(s) is indicated by the permitted-object-type field.
permitted-object-type	String Enum	Required	Identifies the type of object(s) identified by the permitted-object field. Supported values are: • "object" - permitted-object contains a URI that identifies one of the following: – A specific managed object. – A Group object. – A Task object. • "object-class" - permitted-object contains a String Enum value from Appendix A, “Enum values for a type of managed objects within User Roles,” on page 973.
include-members	Boolean	Optional	Indicates whether the members of the group are included in this operation, or if only the group itself is included. True if members are included; false otherwise. Prerequisite: the permitted-object field identifies a Group object. Default: false
view-only-mode	Boolean	Optional	Indicates whether it is permission to a task's view-only version that is to be removed from this User Role. Only certain tasks support a view-only mode. This field is only allowed if the permitted-object field identifies such a Task object. Prerequisite: the permitted-object field identifies a Task object that supports a view-only mode. Default: true

Description

This operation removes a permission from a User Role

On successful execution of this operation the permission specified in the request body has been removed from the User Role identified in the request URI.

The request body is validated against the schema described in [“Request body contents”](#) on page 668. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User Role object, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. If the URI in the request body does not designate an existing resource, status code 404 (Not Found) is returned. An attempt to alter a system-defined User Role is not valid and fails with status code 400 (Bad Request).

If the specified permission is not in the User Role, or if removing it would leave one or more users without object-access permission to their default group, status code 409 (Conflict) is returned.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 338. Remove Permission from User Role: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be altered.
	316	The permission identified in the request body is not in the User Role.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	316	The permission identified in the request body is not in the User Role.
	328	The permission cannot be removed at this time, because doing so would leave one or more users without object-access permission to their default group.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91/operations/  
  remove-permission HTTP/1.1  
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c  
content-type: application/json  
content-length: 114  
{  
  "permitted-object":"/api/console/tasks/5d8c9f60-a2b3-4327-9fd4-791df8a60dcc",  
  "permitted-object-type":"object"  
}
```

Figure 380. Remove Permission from User Role: Request

```
204 No Content  
server: zSeries management console API web server / 2.0  
cache-control: no-cache  
date: Thu, 02 Oct 2014 21:27:32 GMT  
  
<No response body>
```

Figure 381. Remove Permission from User Role: Response

Create User Role

The Create User Role operation creates a user-defined User Role object with the given properties on the console. This operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/console/user-roles
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the User Role's name property.
description	String	Optional	The value to be set as the User Role's description property.
associated-system-defined-user-role-uri	String/ URI	Optional	The value to be set as the User Role's associated-system-defined-user-role-uri property.
is-inheritance-enabled	Boolean	Optional	The value to be set as the User Role's is-inheritance-enabled property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the new User Role object.

Description

This operation creates a new, empty, user-defined User Role.

On successful execution of this operation the User Role is created using the inputs as specified by the request body. The URI of the new User Role is provided in the response body and in a **Location** response header as well. An Inventory Change notification is emitted asynchronously. The Add Permission to User Role operation can then be used to add permissions to the new User Role.

The request body is validated against the schema described in “Request body contents” on page 671. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the specified **name** is not unique, status code 400 (Bad Request) is returned. In addition, the API user must have action/task permission to the **Manage User Roles** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 671, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 339. Create User Role: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A User Role with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43 and the Location response header contains the URI of the newly created object.

Example HTTP interaction

```
POST /api/console/user-roles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 78
{
  "description": "Role for managing department business",
  "name": "Dept Admin"
}
```

Figure 382. Create User Role: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT
content-type: application/json; charset=UTF-8
content-length: 69
{
  "object-uri": "/api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91"
}
```

Figure 383. Create User Role: Response

Delete User Role

The Delete User Role operation deletes a user-defined User Role object designated by its object ID. This operation is not valid for system-defined User Roles.

HTTP method and URI

```
DELETE /api/user-roles/{user-role-id}
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role object to be deleted.

Description

This operation removes a specified User Role from the console. The User Role is identified by the *{user-role-id}* variable in the URI.

Upon successfully removing the User Role, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously.

The URI path must designate an existing User Role object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. An attempt to delete a system-defined User Role is not valid and fails with status code 400 (Bad Request). If any user has the specified User Role, the request fails and status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 340. Delete User Role: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	312	This operation is not supported for an object of this type. System-defined User Roles may not be deleted.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this User Role.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 384. Delete User Role: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 385. Delete User Role: Response

Inventory service data

Information about the User Roles managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User Role objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"user-role"** are to be included. An entry for a particular User Role is included only if the API user has access permission to that object as described in the `Get User Role Properties` operation.

For each User Role object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get User Role Properties` operation. That is, the data provided is the same as would be provided if a `Get User Role Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single User Role. This object would appear as one array entry in the response array:

```
{
  "associated-system-defined-user-role-uri": "/api/user-roles/b39afb87-d915-4070-a22f-91b158c6c01e",
  "class": "user-role",
  "description": "Role that allows management of users",
  "is-inheritance-enabled": false,
  "is-locked": false,
  "name": "user_roles_9",
  "object-id": "db7f9448-3737-11e4-a5fc-5ef3fcae8020",
  "object-uri": "/api/user-roles/db7f9448-3737-11e4-a5fc-5ef3fcae8020",
  "parent": "/api/console",
  "permissions": [
    {
      "permitted-object": "/api/console/tasks/f8d653f4-eab2-4547-97c0-a26f762218ba",
      "permitted-object-type": "object"
    },
    {
      "permitted-object": "/api/console/tasks/36e32fb4-7b60-4677-b462-e786f337ea0f",
      "permitted-object-type": "object"
    },
    {
      "permitted-object": "/api/console/tasks/8ef2b7ca-c2d2-4a5b-9d52-b4d1a28ccb15",
      "permitted-object-type": "object"
    }
  ],
  "replication-overwrite-possible": false,
  "type": "user-defined"
}
```

Figure 386. User Role object: Sample inventory data

Task object

A Task object is an element of the console object and represents an action that a console user with appropriate authority can perform. These actions could be available via the console's graphical user interface, the Web Services APIs or both. Tasks are predefined by the console and cannot be created, modified or deleted.

All API users have object-access permission to all Tasks and thus are permitted to issue `List Tasks` and `Get Task Properties` for any Task object. Note that this object-access permission to a Task object does not give an API user action/task permission to the task represented by the Task object.

Data model

The Task object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics”](#) on page 68.

Table 341. Task object: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Task object is of the form <code>/api/console/tasks/{task-id}</code> , where <code>{task-id}</code> is the value of the element-id property of the Task object.

Table 341. Task object: properties (continued)

Name	Qualifier	Type	Description
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/URI	The canonical URI path of the console object.
class	—	String	The class of a Task object is "task" .
name	(ro)	String Enum	The name of the Task object. The task names are documented in Appendix C, "Enum values for the Task object," on page 977.
description	—	String (0-1024)	The description of the Task object. Default: an empty string
view-only-mode-supported	—	Boolean	Indicates whether this task supports a view-only mode.

List Tasks

The List Tasks operation lists Tasks defined to the console.

HTTP method and URI

GET /api/console/tasks

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
tasks	Array of objects	Array of nested task-info objects as described in the next table.

Each nested task-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	The element-uri property of the Task object.
name	String	The name property of the Task object.

Description

This operation lists tasks defined to the console. Some basic properties are provided for each task.

If the **name** query parameter is specified the returned list is limited to those tasks that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If no tasks are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 676](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/tasks?name=manage-user.* HTTP/1.1
x-api-session: 3hrke2gqugoxejhxenpcdd022kff8iwx10opa33yiv3vse17pd
```

Figure 387. List Tasks: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 09 Sep 2019 18:27:18 GMT
content-type: application/json; charset=UTF-8
content-length: 417
{
  "tasks": [
    {
      "element-uri": "/api/console/tasks/4d3beaea-8971-4ea2-a96d-caff8e60cb10",
      "name": "manage-user-roles"
    },
    {
      "element-uri": "/api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7",
      "name": "manage-users"
    },
    {
      "element-uri": "/api/console/tasks/89c2501e-469a-4b95-81d7-188198e8528f",
      "name": "manage-user-templates"
    },
    {
      "element-uri": "/api/console/tasks/e5b286dd-8089-448b-a01e-d6261e84a283",
      "name": "manage-user-patterns"
    }
  ]
}
```

Figure 388. List Tasks: Response

Get Task Properties

The Get Task Properties operation retrieves the properties of a single Task object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/tasks/{task-id}
```

In this request, the URI variable *{task-id}* is the element ID of the Task object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the Task object as defined in the data model section. Field names and data types in the JSON object are the same as the property names and data types defined in the [“Data model” on page 675](#).

Description

This operation returns the current properties of a single Task object specified by *{task-id}*.

On successful execution, all of the current properties as defined in the data model for the Task object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Task object; otherwise, status code 404 (Not Found) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 678](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7 HTTP/1.1
x-api-session: 69prd5pgk20edka08dmxjaqp0iyqpu0ixiq2028tduivi242yh
```

Figure 389. Get Task Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 09 Sep 2019 18:28:56 GMT
content-type: application/json; charset=UTF-8
content-length: 268
{
  "class": "task",
  "description": "Create, modify, and delete users",
  "element-id": "8a522e34-ca08-475a-9d71-90100f2316a7",
  "element-uri": "/api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7",
  "name": "manage-users",
  "parent": "/api/console",
  "view-only-mode-supported": false
}
```

Figure 390. Get Task Properties: Response

Inventory service data

Information about the tasks managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Task objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included.

For each Task object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the Get Task Properties operation. That is, the data provided is the same as would be provided if a Get Task Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a single Task. This object would appear as one array entry in the response array:

```

{
  "class": "task",
  "description": "Customize the Application Programming Interface for the console",
  "element-id": "ee76dca1-9aee-4530-bf66-667ae728ed10",
  "element-uri": "/api/console/tasks/ee76dca1-9aee-4530-bf66-667ae728ed10",
  "name": "customize-api-settings",
  "parent": "/api/console",
  "view-only-mode-supported": false
}

```

Figure 391. Task object: Sample inventory data

User Pattern object

User Patterns and user templates allow a system administrator to define a group of console users at once whose user IDs all match a certain pattern (for example, a regular expression) and who have a certain set of attributes. A User Pattern object is an element of the console object and defines a pattern for user IDs that are not defined to the console but can be verified by an LDAP server for user authentication. Each pattern identifies a template User object which defines many characteristics of such users. A successful logon with a user ID that matches a User Pattern results in the creation of a pattern-based user, with many of its attributes coming from the associated template. User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object operation [“Reorder User Patterns”](#) on page 591.

Through user-related-access permission described in [“User-related-access permission”](#) on page 629, API users are permitted to see certain User Pattern objects in a `List User Patterns` response and issue `Get User Pattern Properties` for those User Pattern objects. An API user with action/task permission to the `Manage User Patterns` task is permitted to view and change any User Pattern object and change the pattern search order.

User Pattern objects may be replicated to this HMC via its Data Replication facility. If that is the case, the `Update User Pattern Properties` and `Delete User Pattern` operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

Data model

This object contains the following properties. Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **template-name-override** is **null** when the **ldap-server-definition-uri** value is **null**.

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics”](#) on page 68.

Table 342. User Pattern object: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the User Pattern object is of the form <code>/api/console/user-patterns/{user-pattern-id}</code> , where <code>{user-pattern-id}</code> is the value of the element-id property of the User Pattern object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a User Pattern object is "user-pattern" .

Table 342. User Pattern object: properties (continued)

Name	Qualifier	Type	Description
name	(w)	String	The name of the User Pattern object. This name must be unique among all User Patterns on the console. The length and character requirements on this property are the same as those of the name property described in the “Base managed object properties schema” on page 70. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new User Pattern object.
description	(w)	String (0-1024)	The description of the User Pattern object. Default: an empty string
pattern	(w)	String	The actual User Pattern expression. The combination of pattern and type must be unique on the console. Must not be an empty string.
type	(w)	String Enum	The style in which this pattern is expressed, which is one of the following values: • "glob-like" - The pattern may include limited special characters: an asterisk in the pattern matches zero or more characters in the user ID, and a question mark in the pattern matches any single character in the user ID. • "regular-expression" - The pattern is expressed as a UNIX regular expression.
search-order-index	—	Integer	A zero-based index position of this User Pattern in the pattern search order used during logon processing.
retention-time	(w)	Integer (0-21474 83647)	The time in days that the user data for pattern-based users created based on this pattern will be retained. A value of 0 indicates not to retain the settings.
user-template-uri	(w)	String/ URI	The canonical URI path of the User object containing the template definition that applies when a successful match occurs using this User Pattern.
ldap-server-definition-uri	(w)	String/ URI	The canonical URI path of the LDAP Server Definition object which identifies the LDAP server to be used to fetch the template name and/or domain names mentioned in the template-name-override and domain-name-restrictions properties when processing a logon for a user ID that matches this pattern, or null if the LDAP server in the template is used.
template-name-override	(w)	String	The name of the LDAP attribute that contains the name of the user template definition for the user, or null if there is no such attribute. When not null, this property overrides the user template identified in the user-template-uri field, and it must not be an empty string. Prerequisite: ldap-server-definition-uri is not null Default: null

Table 342. User Pattern object: properties (continued)

Name	Qualifier	Type	Description
domain-name-restrictions	(w)	String	The name of the LDAP attribute that contains the information about which consoles the user is allowed to log onto, or null if there is no such attribute. Must not be an empty string. Prerequisite: ldap-server-definition-uri is not null Default: null
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

List User Patterns

The `List User Patterns` operation lists User Patterns defined to the console.

HTTP method and URI

`GET /api/console/user-patterns`

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid type property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
user-patterns	Array of objects	Array of nested user-pattern-info objects as described in the next table.

Each nested user-pattern-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	The element-uri property of the User Pattern object.
name	String	The name property of the User Pattern object.
type	String Enum	The type property of the User Pattern object.

Description

This operation lists User Patterns defined to the console. Some basic properties are provided for each User Pattern.

A User Pattern is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage User Patterns** task. If there is a User Pattern to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no User Patterns defined to the console or if no User Patterns are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Pattern objects included in the response body or action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 682](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage Notes

User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object operation [“Reorder User Patterns” on page 591](#). The `List User Patterns` operation does not specify the order in which the User Pattern URIs appear in the response body, and there is no guarantee that the order in the response will not change in subsequent invocations. Use the **search-order-index** property to determine a pattern's position in the search order.

Example HTTP interaction

```
GET /api/console/user-patterns?type=regular-expression HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzi4c
```

Figure 392. List User Patterns: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json; charset=UTF-8
content-length: 314
{
  "user-patterns": [
    {
      "element-uri": "/api/console/user-patterns/497bf4ec-1dbf-11e4-8ceb-1c6f65065a91",
      "name": "IBM Intranet User Pattern",
      "type": "regular-expression"
    },
    {
      "element-uri": "/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91",
      "name": "Company email pattern",
      "type": "regular-expression"
    }
  ]
}
```

Figure 393. List User Patterns: Response

Get User Pattern Properties

The Get User Pattern Properties operation retrieves the properties of a single User Pattern object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable *{user-pattern-id}* is the element ID of the User Pattern object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User Pattern object as defined in the [“Data model” on page 680](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single User Pattern object specified by *{user-pattern-id}*.

On successful execution, all of the current properties as defined in the data model for the User Pattern object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User Pattern object and the API user must have user-related-access permission to it or action/task permission to the **Manage User Patterns** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Pattern object specified in the request URI or action/task permission to the **Manage User Patterns** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 684](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 394. Get User Pattern Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json; charset=UTF-8
content-length: 593
{
  "class": "user-pattern",
  "description": "User Pattern based on company email addresses",
  "domain-name-restrictions": null,
  "element-id": "ec5b012a-4a7a-11e4-8777-1c6f65065a91",
  "element-uri": "/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91",
  "ldap-server-definition-uri": null,
  "name": "Company email pattern",
  "parent": "/api/console",
  "pattern": ".*@our\\.company\\.com",
  "replication-overwrite-possible": false,
  "retention-time": 8,
  "search-order-index": 3,
  "template-name-override": null,
  "type": "regular-expression",
  "user-template-uri": "/api/users/ec473e56-4a7a-11e4-91ee-1c6f65065a91"
}
```

Figure 395. Get User Pattern Properties: Response

Update User Pattern Properties

The Update User Pattern Properties operation updates the properties of a single User Pattern object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable `{user-pattern-id}` is the element ID of the User Pattern object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the User Pattern object specified by *{user-pattern-id}*.

The URI path must designate an existing User Pattern object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Pattern object or action/task permission to the **Manage User Patterns** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Pattern object but not action/task permission to the **Manage User Patterns** task, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., attempting to set **template-name-override** when the **ldap-server-definition-uri** value is **null**).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **ldap-server-definition-uri** is changed from non-null to **null**, and **template-name-override** is not defined in the request body, **template-name-override** will be defaulted to **null**).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 343. Update User Pattern Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	324	The ldap-server-definition-uri field in the request body does not designate an existing LDAP Server Definition object.
	326	The user-template-uri field in the request body does not designate an existing template type User object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 95
{
  "description": "A new and improved description of this User Pattern",
  "retention-time": 30
}
```

Figure 396. Update User Pattern Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 397. Update User Pattern Properties: Response

Create User Pattern

The Create User Pattern operation creates a User Pattern object with the given properties on the console.

HTTP method and URI

```
POST /api/console/user-patterns
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the User Pattern's name property.
description	String	Optional	The value to be set as the User Pattern's description property.
pattern	String	Required	The value to be set as the User Pattern's pattern property.
type	String Enum	Required	The value to be set as the User Pattern's type property.
retention-time	Integer	Required	The value to be set as the User Pattern's retention-time property.
user-template-uri	String/ URI	Required	The value to be set as the User Pattern's user-template-uri property.

Field name	Type	Rqd/Opt	Description
ldap-server-definition-uri	String/ URI	Optional	The value to be set as the User Pattern's ldap-server-definition-uri property.
template-name-override	String	Optional	The value to be set as the User Pattern's template-name-override property.
domain-name-restrictions	String	Optional	The value to be set as the User Pattern's domain-name-restrictions property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the new User Pattern object.

Description

This operation creates a new User Pattern.

On successful execution of this operation the User Pattern is created using the inputs as specified by the request body. The new User Pattern is placed at the end of the pattern search order used during logon processing. The URI of the new User Pattern is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the [“Request body contents”](#) on page 687. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **template-name-override** when the **ldap-server-definition-uri** value is **null**) or the specified pattern is not unique. If a URI in the request body does not designate an existing resource of the appropriate type, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage User Patterns** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents”](#) on page 688, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 344. Create User Pattern: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A User Pattern with the pattern specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	324	The ldap-server-definition-uri field in the request body does not designate an existing LDAP Server Definition object.
	326	The user-template-uri field in the request body does not designate an existing template type User object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage Notes

User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object's Reorder User Patterns operation.

Example HTTP interaction

```
POST /api/console/user-patterns HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 260
{
  "description": "User Pattern based on company email addresses",
  "name": "Company email pattern",
  "pattern": ".*@our\\.company\\.com",
  "retention-time": 8,
  "type": "regular-expression",
  "user-template-uri": "/api/users/ec473e56-4a7a-11e4-91ee-1c6f65065a91"
}
```

Figure 398. Create User Pattern: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 83
{
  "element-uri": "/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 399. Create User Pattern: Response

Delete User Pattern

The Delete User Pattern operation deletes a User Pattern object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable *{user-pattern-id}* is the element ID of the User Pattern object to be deleted.

Description

This operation removes a specified User Pattern from the console. The User Pattern is identified by the *{user-pattern-id}* variable in the URI.

Upon successfully removing the User Pattern, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing User Pattern object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Pattern object or action/task permission to the **Manage User Patterns** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Pattern object but not action/task permission to the **Manage User Patterns** task, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Usage Note

It is permitted to delete a User Pattern even if there is a pattern-based user based on that pattern logged onto the HMC at the time of the deletion. Note that this will cause certain operations issued with the value of the **user-pattern-uri** property in that user's User object to fail, most likely with a 404 (Not Found) status code.

Example HTTP interaction

```
DELETE /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 400. Delete User Pattern: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 401. Delete User Pattern: Response

Inventory service data

Information about the User Patterns managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User Pattern objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular User Pattern is included only if the API user has access permission to that object as described in the Get User Pattern Properties operation.

For each User Pattern object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the Get User Pattern Properties operation. That is, the data provided is the same as would be provided if a Get User Pattern Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the Get Inventory response to describe a single User Pattern. This object would appear as one array entry in the response array:

```
{
  "class": "user-pattern",
  "description": "",
  "domain-name-restrictions": null,
  "element-id": "8c1ae17a-34c4-11e4-a30c-5ef3fcae8020",
  "element-uri": "/api/console/user-patterns/8c1ae17a-34c4-11e4-a30c-5ef3fcae8020",
  "ldap-server-definition-uri": "/api/console/ldap-server-definitions/4927787e-34c4-11e4-a1ea-5ef3fcae8020",
  "name": "testpattern_inventory",
  "parent": "/api/console",
  "pattern": "user-*",
  "replication-overwrite-possible": false,
  "retention-time": 0,
  "search-order-index": 0,
  "template-name-override": null,
  "type": "glob-like",
  "user-template-uri": "/api/users/641c7d96-34c4-11e4-b2c2-5ef3fcae8020"
}
```

Figure 402. User Pattern object: Sample inventory data

Password Rule object

A Password Rule object is an element of the console object and represents a rule which a console user(s) must follow when creating a console logon password. Each console user using local authentication is assigned a password rule. There are certain system-defined password rules available for use.

Through user-related-access permission described in [“User-related-access permission”](#) on page 629, API users are permitted to see certain Password Rule objects in a `List Password Rules` response and issue `Get Password Rule Properties` for those Password Rule objects. An API user with action/task permission to the `Manage Password Rules` task is permitted to view any Password Rule object and change any user-defined Password Rule object.

Password rule objects may be replicated to this HMC via its Data Replication facility. If that is the case, the `Update Password Rule Properties` and `Delete Password Rule` operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

System-defined password rules

Unlike user-defined password rules, the system-defined password rules may not be modified. While system-defined password rules can be deleted and their name reused for a user-defined password rule, that practice is discouraged due to the likely confusion such a situation would cause. The names of the typical system-defined password rules include:

- Basic
- Standard
- Strict

Password rule parts

Password rule parts are optional requirements to be applied to individual parts of a password. These requirements are applied, in order, to the password, from left to right. Each of these requirements must be met by some part of the password in order for the password to meet all of the requirements of the Password Rule.

For example, to require a password to consist of 1-3 letters followed by a 4 or 5 digit number, two rule parts are defined. The first rule part requires from 1 to 3 characters, each of which must be alphabetic; the second rule part requires from 4 to 5 characters, each of which must be numeric. Passwords such as "pa1600" and "Hey90210" meet the requirements of both of those rule parts.

Data model

The Password Rule object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics”](#) on page 68.

Table 345. Password Rule object: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for a Password Rule object is of the form <code>/api/console/password-rules/{password-rule-id}</code> , where <code>{password-rule-id}</code> is the value of the element-id property of the Password Rule object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.

Table 345. Password Rule object: properties (continued)

Name	Qualifier	Type	Description
class	—	String	The class of a Password Rule object is "password-rule" .
name	(ro)	String	The name of the Password Rule object. This name must be unique among all password rules on the console. While preexisting Password Rule names are virtually unrestricted in terms of length and characters, new Password Rule names must conform to the length and character requirements of the name property described in the “Base managed object properties schema” on page 70. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new Password Rule object.
description	(w)	String (0-1024)	The description of the Password Rule object. Default: an empty string
type	—	String Enum	Identifies the type of password rule. It must be one of the following values: • "system-defined" - A password rule defined by the system. System-defined rules may not be modified. • "user-defined" - A password rule defined by a user.
expiration	(w)	Integer	The total number of days a password is valid before it expires. A value of 0 indicates that the password never expires. Default: 0
min-length	(w)	Integer (1-256)	The minimum required length of the password. Cannot be greater than max-length. Default: 8
max-length	(w)	Integer (1-256)	The maximum allowed length of the password. Cannot be less than min-length. Default: 256
consecutive-characters	(w)	Integer	The maximum number of characters that are allowed to be repeated in a row. A value of 0 indicates that there is no such limit. Default: 0
similarity-count	(w)	Integer	The maximum number of consecutive characters in the current password that can match consecutive characters in the previous password. A value of 0 indicates that there is no such limit. Default: 0
history-count	(w)	Integer	The number of previous passwords to which a new password is compared for uniqueness. A value of 0 indicates that there is no such comparison. Default: 0

Table 345. Password Rule object: properties (continued)

Name	Qualifier	Type	Description
case-sensitive	(w)	Boolean	Indicates whether the password is case sensitive. Default: false
character-rules	(w)	Array of objects	Optional rules to be applied to individual parts of the password. These rules are applied, in order, to the password, from left to right. Each of these rules must be met by some part of the password in order for the password to meet the requirements of this Password Rule. This property is an array of nested character-rule objects as described in the next table. If there are no rule parts, an empty array is provided. Default: <empty array>
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

Each nested password-rule-part object contains the following fields:

Table 346. character-rule object properties

Name	Type	Description
min-characters	Integer	The minimum number of characters required by this password rule part. Must be at least 1, and cannot be greater than max-characters .
max-characters	Integer	The maximum number of characters allowed by this password rule part. Must be at least 1, and cannot be less than min-characters .
alphabetic	String Enum	This field determines the inclusion of alphabetic characters within this part of the password. It must be one of the following values: • "allowed" - There can be alphabetic characters. • "not-allowed" - There cannot be alphabetic characters. • "required" - There must be alphabetic characters.
numeric	String Enum	This field determines the inclusion of numeric characters within this part of the password. It must be one of the following values: • "allowed" - There can be numeric characters. • "not-allowed" - There cannot be numeric characters. • "required" - There must be numeric characters.

Table 346. character-rule object properties (continued)

Name	Type	Description
special	String Enum	This field determines the inclusion of special characters within this part of the password. It must be one of the following values: • "allowed" - There can be special characters. • "not-allowed" - There cannot be special characters. • "required" - There must be special characters. Special characters include: greater than (>), less than (<), tilde (~), exclamation mark (!), at sign (@), number sign (#), question mark (?), dollar sign (\$), vertical bar (), percent sign (%), caret (^), ampersand (&), asterisk (*), left and right parentheses (), underscore (_), plus sign (+), hyphen (-), equals sign (=), left and right curly braces ({ }), left and right square brackets ([]), back slash (\), forward slash (/), period (.), comma (,), colon (:), accent (`), quotation mark ("), semicolon (;), and apostrophe (').
custom-character-sets	Array of objects	Optional specific character requirements for this password part, as specified in an array of nested custom-character-set objects defined in the next table. This allows the specification of custom character sets and their inclusion requirement. There can be up to 2 custom character sets for a rule part. If none are defined, an empty array is provided.

Each nested specific-property object contains the following fields:

Table 347. custom-character-set object properties

Name	Type	Description
character-set	String	A string consisting of the characters that comprise this custom character set.
inclusion	String Enum	This field determines the inclusion of characters in this character set within part of the password. It must be one of the following values: • "allowed" - Characters can be included. • "not-allowed" - Characters cannot be included. • "required" - At least one character must be included.

List Password Rules

The `List Password Rules` operation lists Password Rules defined to the console.

HTTP method and URI

```
GET /api/console/password-rules
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Name	Type	Rqd/Opt	Description
type	String Enum	Optional	Filter string to limit returned objects to those that have a matching type property. Value must be a valid type property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
password-rules	Array of objects	Array of nested password-rule-info objects as described in the next table. If no Password Rules are to be returned, an empty array is provided.

Each nested password-rule-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	The element-uri property of the Password Rule object.
name	String	The name property of the Password Rule object.
type	String Enum	The type property of the Password Rule object.

Description

This operation lists Password Rules defined to the console. Some basic properties are provided for each Password Rule.

If the **name** query parameter is specified the returned list is limited to those Password Rules that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid Password Rule **type** property value. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those Password Rules that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A Password Rule is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage Password Rules** task. If there is a console Password Rule to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no Password Rules defined to the console or if no Password Rules are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- User-related-access permission to the Password Rules objects included in the response body or action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 696](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/password-rules?type=system-defined HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 403. List Password Rules: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 390
{
  "password-rules": [
    {
      "element-uri": "/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
      "name": "Basic",
      "type": "system-defined"
    },
    {
      "element-uri": "/api/console/password-rules/4a79360a-3dbf-11e4-980d-1c6f65065a91",
      "name": "Standard",
      "type": "system-defined"
    },
    {
      "element-uri": "/api/console/password-rules/4a792b24-3dbf-11e4-980d-1c6f65065a91",
      "name": "Strict",
      "type": "system-defined"
    }
  ]
}
```

Figure 404. List Password Rules: Response

Get Password Rule Properties

The Get Password Rule Properties operation retrieves the properties of a single Password Rule object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/password-rules/{password-rule-id}
```

In this request, the URI variable *{password-rule-id}* is the element ID of the Password Rule object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the Password Rule object as defined in the [“Data model”](#) on page 692. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single Password Rule object that is designated by *{password-rule-id}*.

On successful execution, all of the current properties as defined in the data model for the Password Rule object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Password Rule object and the API user must have user-related-access permission to it or action/task permission to the **Manage Password Rules** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the Password Rule object specified in the request URI or action/task permission to the **Manage Password Rules** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 698.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 405. Get Password Rule Properties: Request


```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json; charset=UTF-8
content-length: 656
{
  "case-sensitive":true,
  "character-rules":[
    {
      "alphabetic":"not-allowed",
      "custom-character-sets":[
        {
          "character-set":"*!^",
          "inclusion":"not-allowed"
        }
      ],
      "max-characters":256,
      "min-characters":8,
      "numeric":"not-allowed",
      "special":"required"
    }
  ],
  "class":"password-rule",
  "consecutive-characters":0,
  "description":"Password must be very special",
  "element-id":"ecb26fb4-4a7a-11e4-affa-1c6f65065a91",
  "element-uri":"/api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91",
  "expiration":0,
  "history-count":0,
  "max-length":256,
  "min-length":8,
  "name":"All specials",
  "parent":"/api/console",
  "replication-overwrite-possible":false,
  "similarity-count":0,
  "type":"user-defined"
}

```

Figure 406. Get Password Rule Properties: Response

Update Password Rule Properties

The Update Password Rule Properties operation updates the properties of a single user-defined Password Rule object that is designated by its element ID.

HTTP method and URI

POST /api/console/password-rules/{password-rule-id}

In this request, the URI variable {password-rule-id} is the element ID of the Password Rule object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the Password Rule object specified by {password-rule-id}.

The URI path must designate an existing Password Rule object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated Password Rule

object or action/task permission to the **Manage Password Rules** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated Password Rule object but not action/task permission to the **Manage Password Rules** task, status code 403 (Forbidden) is returned

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. An attempt to update a system-defined password rule is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 348. Update Password Rule Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined password rules may not be updated.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 109
{
  "description": "A new and improved description of this Password Rule",
  "expiration": 90,
  "history-count": 5
}
```

Figure 407. Update Password Rule Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 408. Update Password Rule Properties: Response

Create Password Rule

The Create Password Rule operation creates a user-defined Password Rule object with the given properties.

HTTP method and URI

POST /api/console/password-rules

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the Password Rule's name property.
description	String	Optional	The value to be set as the Password Rule's description property.
expiration	Integer	Optional	The value to be set as the Password Rule's expiration property.
min-length	Integer	Optional	The value to be set as the Password Rule's min-length property.
max-length	Integer	Optional	The value to be set as the Password Rule's max-length property.
consecutive-characters	Integer	Optional	The value to be set as the Password Rule's consecutive-characters property.
similarity-count	Integer	Optional	The value to be set as the Password Rule's similarity-count property.
history-count	Integer	Optional	The value to be set as the Password Rule's history-count property.
case-sensitive	Boolean	Optional	The value to be set as the Password Rule's case-sensitive property.
character-rules	Array of objects	Optional	The value to be set as the Password Rule's character-rules property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the new Password Rule object.

Description

This operation creates a new user-defined Password Rule.

On successful execution of this operation the Password Rule is created using the inputs as specified by the request body. The URI of the new Password Rule is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the [“Request body contents” on page 701](#). If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the specified name is not unique, status code 400 (Bad Request) is returned. In addition, the API user must have action/task permission to the Manage Password Rules task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 701](#) and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 349. Create Password Rule: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	A password rule with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/password-rules HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 298
{
  "character-rules":[
    {
      "alphabetic":"not-allowed",
      "custom-character-sets":[
        {
          "character-set":"*!^",
          "inclusion":"not-allowed"
        }
      ],
      "max-characters":256,
      "min-characters":8,
      "numeric":"not-allowed",
      "special":"required"
    }
  ],
  "description":"Password must be very special",
  "name":"All specials"
}
```

Figure 409. Create Password Rule: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json; charset=UTF-8
content-length: 82
{
  "element-uri":"/api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91"
}
```

Figure 410. Create Password Rule: Response

Delete Password Rule

The Delete Password Rule operation deletes a Password Rule object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/password-rules/{password-rule-id}
```

In this request, the URI variable *{password-rule-id}* is the element ID of the Password Rule object to be deleted.

Description

This operation removes a specified Password Rule from the console. The Password Rule is identified by the *{password-rule-id}* variable in the URI.

Upon successfully removing the Password Rule, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing Password Rule object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated Password Rule object or action/task permission to the **Manage Password Rules** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated Password Rule object but not

action/task permission to the **Manage Password Rules** task, status code 403 (Forbidden) is returned. If any user has the specified Password Rule, the request fails and status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this Password Rule.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzi4c
```

Figure 411. Delete Password Rule: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 412. Delete Password Rule: Response

Inventory service data

Information about the Password Rules managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Password Rule objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular Password Rule is

included only if the API user has access permission to that object as described in the `Get Password Rule Properties` operation.

For each Password Rule object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get Password Rule Properties` operation. That is, the data provided is the same as would be provided if a `Get Password Rule Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single Password Rule. This object would appear as one array entry in the response array:

```
{
  "case-sensitive": false,
  "character-rules": [
    {
      "alphabetic": "required",
      "custom-character-sets": [],
      "max-characters": 1,
      "min-characters": 1,
      "numeric": "not-allowed",
      "special": "not-allowed"
    },
    {
      "alphabetic": "allowed",
      "custom-character-sets": [],
      "max-characters": 6,
      "min-characters": 4,
      "numeric": "required",
      "special": "not-allowed"
    },
    {
      "alphabetic": "required",
      "custom-character-sets": [],
      "max-characters": 1,
      "min-characters": 1,
      "numeric": "not-allowed",
      "special": "not-allowed"
    }
  ],
  "class": "password-rule",
  "consecutive-characters": 2,
  "description": "",
  "element-id": "56d11882-eaff-11e2-9ec7-5cf3fcae8019",
  "element-uri": "/api/console/password-rules/56d11882-eaff-11e2-9ec7-5cf3fcae8019",
  "expiration": 180,
  "history-count": 0,
  "max-length": 8,
  "min-length": 6,
  "name": "Strict",
  "parent": "/api/console",
  "replication-overwrite-possible": false,
  "similarity-count": 0,
  "type": "system-defined"
}
```

Figure 413. Password Rule object: Sample inventory data

LDAP Server Definition object

An LDAP Server Definition object is an element of the console object and contains information about an LDAP server that may be used for console user authorization purposes. LDAP servers are sometimes referred to as Enterprise Directory Servers on the zManager user interface and publications.

All API users are permitted to issue List LDAP Server Definitions and retrieve very basic information for the LDAP Server Definition, if any, that applies to them. That is, they can use that operation to retrieve very basic information for the LDAP Server Definition identified in their User object. An API user with action/task permission to the **Manage LDAP Server Definitions** task is permitted to view and change any LDAP Server Definition object.

LDAP Server Definition objects may be replicated to this HMC via its Data Replication facility. If that is the case, the Update LDAP Server Definition Properties and Delete LDAP Server Definition operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

Data model

This object contains the following properties. Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **search-filter** is **null** when the **location-method** value is **"pattern"**.

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

Table 350. LDAP Server Definition object: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for an LDAP Server Definition object is of the form <code>/api/console/ldap-server-definitions/{ldap-server-definition-id}</code> , where <code>{ldap-server-definition-id}</code> is the value of the element-id property of the LDAP Server Definition object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of an LDAP Server Definition object is "ldap-server-definition" .
name	(ro)	String	The name of the LDAP Server Definition object. This name must be unique among all LDAP Server Definition objects defined to the console. While preexisting LDAP Server Definition names are virtually unrestricted in terms of length and characters, new LDAP Server Definition names must conform to the length and character requirements of the name property described in the “Base managed object properties schema” on page 70. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new LDAP Server Definition object.
description	(w)	String (0-1024)	The description of the LDAP Server Definition object. Default: an empty string
primary-hostname-ipaddr	(w)	String, String/ IPV4 address, or String/IPV6 address	The host name or IP address of the primary LDAP server. It must contain at least 1 non-whitespace character.

Table 350. LDAP Server Definition object: properties (continued)

Name	Qualifier	Type	Description
connection-port	(w)	Integer	The TCP port number, which must be greater than 0, on which the server accepts connections, or null if the server uses the standard LDAP port appropriate for the value of the use-ssl property. The standard LDAP port values are 636 if use-ssl is true and 389 if use-ssl is false . Default: null
backup-hostname-ipaddr	(w)	String, String/IPV4 address, or String/IPV6 address	The host name or IP address of the backup LDAP server (which must contain at least 1 non-whitespace character), or null if there is none. Default: null
use-ssl	(w)	Boolean	Indicates whether the server uses SSL for incoming connections. Default: false
tolerate-untrusted-certificates	(w)	Boolean	Indicates whether the server should tolerate self-signed or otherwise untrusted certificates. Prerequisite: use-ssl is true Default: false
bind-distinguished-name	(w)	String	Part of the bind information for the initial connection, if needed: the distinguished name, which must contain at least 1 non-whitespace character; otherwise, null . Default: null Prerequisite: Both bind-distinguished-name and bind-password must be null , or they both must be non-empty strings.
bind-password	(wo)	String	Part of the bind information for the initial connection, if needed: the password; otherwise, null . Default: null Prerequisite: Both bind-distinguished-name and bind-password must be null , or they both must be non-empty strings.
location-method	(w)	String Enum	The method this server uses to locate a user's directory entry. Must be one of the following: • "pattern" - Use a distinguished name pattern. • "subtree" - Use a distinguished name subtree.

Table 350. LDAP Server Definition object: properties (continued)

Name	Qualifier	Type	Description
search-distinguished-name	(w)	String	The distinguished name to use when searching for a user's directory entry. If location-method is "pattern", this is the distinguished name pattern to use when searching for a user's directory entry. It must include the string "{0}", which indicates where in the pattern the user ID is to be substituted. The user ID is the value of the userid-on-ldap-server property of the user's User object, unless it is null, in which case the name property of the User object is used. If location-method is "subtree", this is the distinguished name of the subtree to search for a user's directory entry.
search-scope	(w)	String Enum	Indicates how much of the subtree should be searched when searching for a user's directory entry in a subtree. The filter is specified in the search-filter property. Must be one of the following: • "all" - Search the entire subtree. • "one-level" - Search one level only. Prerequisite: location-method is "subtree" Default: "all"
search-filter	(w)	String	The LDAP search filter to use when searching for a user's directory entry in a subtree. It must include the string "{0}", which indicates where in the filter the user ID is to be substituted. The user ID is the value of the userid-on-ldap-server property of the user's User object, unless it is null, in which case the name property of the User object is used. The search-scope property specifies how this filter is used during the search. Prerequisite: location-method is "subtree"
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

List LDAP Server Definitions

The List LDAP Server Definitions operation lists LDAP Server Definitions defined to the console.

HTTP method and URI

GET /api/console/ldap-server-definitions

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
ldap-server-definitions	Array of objects	Array of nested ldap-server-definition-info objects as described in the next table.

Each nested ldap-server-definition-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	The element-uri property of the LDAP Server Definition object.
name	String	The name property of the LDAP Server Definition object.

Description

This operation lists LDAP Server Definitions defined to the console. Some basic properties are provided for each LDAP Server Definition.

If the **name** query parameter is specified the returned list is limited to those LDAP Server Definitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

An LDAP Server Definition is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage LDAP Server Definitions** task. If there is an LDAP Server Definition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no LDAP Server Definitions defined to the console or if no LDAP Server Definitions are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the LDAP Server Definition objects included in the response body or action/task permission to the **Manage LDAP Server Definitions** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 709](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/console/ldap-server-definitions?name=IBM.*&name=Company.* HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 414. List LDAP Server Definitions: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT
content-type: application/json;charset=UTF-8
content-length: 276
{
  "ldap-server-definitions":[
    {
      "element-uri":"/api/console/ldap-server-definitions/
        3ac6550e-1dbb-11e4-9aa4-1c6f65065a91",
      "name":"IBM LDAP server"
    },
    {
      "element-uri":"/api/console/ldap-server-definitions/
        ece481ca-4a7a-11e4-8777-1c6f65065a91",
      "name":"Company LDAP server"
    }
  ]
}
```

Figure 415. List LDAP Server Definitions: Response

Get LDAP Server Definition Properties

The Get LDAP Server Definition Properties operation retrieves the properties of a single LDAP Server Definition object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/ldap-server-definitions/{ldap-server-definition-id}
```

In this request, the URI variable *{ldap-server-definition-id}* is the element ID of the LDAP Server Definition object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the Password Rule object as defined in the [“Data model”](#) on page 706. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single LDAP Server Definition object that is designated by *{ldap-server-definition-id}*.

On successful execution, all of the current properties as defined in the data model for the LDAP Server Definition object, except those designated as write-only properties, are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 710](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 416. Get LDAP Server Definition Properties: Request

```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT
content-type: application/json;charset=UTF-8
content-length: 640
{
  "backup-hostname-ipaddr":null,
  "bind-distinguished-name":null,
  "class":"ldap-server-definition",
  "connection-port":null,
  "description":"Directory server for the company",
  "element-id":"ece481ca-4a7a-11e4-8777-1c6f65065a91",
  "element-uri":"/api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91",
  "location-method":"subtree",
  "name":"Company LDAP server",
  "parent":"/api/console",
  "primary-hostname-ipaddr":"ldap1.our.company.com",
  "replication-overwrite-possible":false,
  "search-distinguished-name":"o=our,ou=company.com",
  "search-filter":"email={0}",
  "search-scope":"all",
  "tolerate-untrusted-certificates":false,
  "use-ssl":true
}

```

Figure 417. Get LDAP Server Definition Properties: Response

Update LDAP Server Definition Properties

The Update LDAP Server Definition Properties operation updates the properties of a single LDAP Server Definition object that is designated by its element ID.

HTTP method and URI

POST /api/console/ldap-server-definitions/{*ldap-server-definition-id*}

In this request, the URI variable {*ldap-server-definition-id*} is the element ID of the LDAP Server Definition object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the LDAP Server Definition object specified by {*ldap-server-definition-id*}.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., attempting to set **search-filter** when the **location-method** value is **"pattern"**).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **location-method** is changed from **"pattern"** to **"subtree"**, and **search-scope** is not defined in the request body, **search-scope** will be defaulted to **all**).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 133
{
  "backup-hostname-ipaddr": "ldap2.my.company.com",
  "description": "A new and improved description of this LDAP Server Definition"
}
```

Figure 418. Update LDAP Server Definition Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT

<No response body>
```

Figure 419. Update LDAP Server Definition Properties: Response

Create LDAP Server Definition

The Create LDAP Server Definition operation creates an LDAP Server Definition object with the given properties.

HTTP method and URI

POST /api/console/ldap-server-definitions

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the LDAP Server Definition's name property.
description	String	Optional	The value to be set as the LDAP Server Definition's description property.
primary-hostname-ipaddr	String	Required	The value to be set as the LDAP Server Definition's primary-hostname-ipaddr property.
connection-port	Integer	Optional	The value to be set as the LDAP Server Definition's connection-port property.
backup-hostname-ipaddr	String	Optional	The value to be set as the LDAP Server Definition's backup-hostname-ipaddr property.
use-ssl	Boolean	Optional	The value to be set as the LDAP Server Definition's use-ssl property.
tolerate-untrusted-certificates	Boolean	Optional	The value to be set as the LDAP Server Definition's tolerate-untrusted-certificates property.
bind-distinguished-name	String	Optional	The value to be set as the LDAP Server Definition's bind-distinguished-name property.
bind-password	String	Optional	The value to be set as the LDAP Server Definition's bind-password property.
location-method	String Enum	Optional	The value to be set as the LDAP Server Definition's location-method property.
search-distinguished-name	String	Required	The value to be set as the LDAP Server Definition's search-distinguished-name property.
search-scope	String Enum	Optional	The value to be set as the LDAP Server Definition's search-scope property.
search-filter	String	Required if location-method is "subtree"	The value to be set as the LDAP Server Definition's search-filter property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the new LDAP Server Definition object.

Description

This operation creates a new LDAP Server Definition.

On successful execution of this operation the LDAP Server Definition is created using the inputs as specified by the request body. The URI of the new LDAP Server Definition is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the [“Request body contents” on page 714](#). If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **search-filter** when the **location-method** value is **"pattern"**) or the specified name is not unique. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 714](#), and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 351. Create LDAP Server Definition: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An LDAP Server Definition with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/console/ldap-server-definitions HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
content-type: application/json
content-length: 264
{
  "description": "Directory server for the company",
  "location-method": "subtree",
  "name": "Company LDAP server",
  "primary-hostname-ipaddr": "ldap1.our.company.com",
  "search-distinguished-name": "o=our,ou=company.com",
  "search-filter": "email={0}",
  "use-ssl": true
}
```

Figure 420. Create LDAP Server Definition: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json; charset=UTF-8
content-length: 94
{
  "element-uri": "/api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 421. Create LDAP Server Definition: Response

Delete LDAP Server Definition

The Delete LDAP Server Definition operation deletes an LDAP Server Definition object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/ldap-server-definitions/{ldap-server-definition-id}
```

In this request, the URI variable *{ldap-server-definition-id}* is the element ID of the LDAP Server Definition object to be deleted.

Description

This operation removes a specified LDAP Server Definition from the console. The LDAP Server Definition is identified by the *{ldap-server-definition-id}* variable in the URI.

Upon successfully removing the LDAP Server Definition, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned. If any user is defined to use the specified LDAP Server Definition, the request fails and HTTP status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this LDAP Server Definition.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91
HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqx18c4r066ge9kcyzr4c
```

Figure 422. Delete LDAP Server Definition: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT

<No response body>
```

Figure 423. Delete LDAP Server Definition: Response

Inventory service data

Information about the LDAP Server Definitions managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for LDAP Server Definition objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular LDAP Server Definition is included only if the API user has access permission to that object as described in the `Get LDAP Server Definition Properties` operation.

For each LDAP Server Definition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get LDAP Server Definition Properties` operation. That is, the data provided is the same as would be provided if a `Get LDAP Server Definition Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single LDAP Server Definition. This object would appear as one array entry in the response array:

```
{
  "backup-hostname-ipaddr": null,
  "bind-distinguished-name": null,
  "class": "ldap-server-definition",
  "connection-port": null,
  "description": "",
  "element-id": "ffbf71f4-370d-11e4-a5fc-5ef3fcae8020",
  "element-uri": "/api/console/ldap-server-definitions/ffbf71f4-370d-11e4-a5fc-5ef3fcae8020",
  "location-method": "pattern",
  "name": "Temp_LDAP_13_56ba2f43-98c0-4848-9af8-cdb45b56f082",
  "parent": "/api/console",
  "primary-hostname-ipaddr": "bluepages.ibm.com",
  "replication-overwrite-possible": false,
  "search-distinguished-name": "uid={0}744,c=in,ou=bluepages,o=ibm.com",
  "search-filter": null,
  "search-scope": null,
  "tolerate-untrusted-certificates": null,
  "use-ssl": false
}
```

Figure 424. LDAP Server Definition object: Sample inventory data

MFA Server Definition object

An MFA Server Definition object is an element of the Console object and contains information about an MFA server that may be used for console user authorization purposes.

All API users are permitted to issue `List MFA Server Definitions` and retrieve very basic information for the MFA Server Definitions, if any, that apply to them. That is, they can use that operation to retrieve very basic information for the MFA Server Definitions identified in their User object. An API user with action/task permission to the **Manage Multi-factor Authentication** task is permitted to view and change any MFA Server Definition object.

MFA Server Definition objects may be replicated to this HMC via its Data Replication facility. If that is the case, the `Update MFA Server Definition Properties` and `Delete MFA Server Definition` operations should be used with care as they will prevent further replication of the modified or deleted object to this HMC.

Data model

The MFA Server Definition object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

Table 352. MFA Server Definition object: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for an MFA Server Definition object is of the form <code>/api/console/mfa-server-definitions/{mfa-server-definition-id}</code> , where <code>{mfa-server-definition-id}</code> is the value of the element-id property of the MFA Server Definition object.
element-id	—	String (36)	The unique identifier for this object.

Table 352. MFA Server Definition object: properties (continued)

Name	Qualifier	Type	Description
parent	—	String/ URI	The canonical URI path of the Console object.
class	—	String (21)	The class of an MFA Server Definition object is "mfa-server-definition" .
name	(w)(pc)	String	The name of the MFA Server Definition object. This name must be unique among all of the console's MFA Server Definitions, and it must conform to the length and character requirements of the name property described in the "Base managed object properties schema" on page 70. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new MFA Server Definition object or update the name of an existing one.
description	(w)(pc)	String (0-1024)	The description of the MFA Server Definition object. Default: an empty string
hostname-ipaddr	(w)(pc)	String/ Hostname, String/IPV4 Address, or String/IPV6 Address	The hostname or IP address of the server. Must not be null.
port	(w)(pc)	Integer (1-65535)	The TCP port number on which the server accepts connections. Default: 6789
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

List MFA Server Definitions

The List MFA Server Definitions operation lists MFA Server Definitions defined to the console.

HTTP method and URI

GET /api/console/mfa-server-definitions

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
mfa-server-definitions	Array of mfa-server-definition-info objects	Array of nested mfa-server-definition-info objects as described in the next table.

Each nested mfa-server-definition-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	The element-uri property of the MFA Server Definition object.
name	String	The name property of the MFA Server Definition object.

Description

This operation lists MFA Server Definitions defined to the console. Some basic properties are provided for each MFA Server Definition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those MFA Server Definitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

An MFA Server Definition is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage Multi-factor Authentication** task. If there is an MFA Server Definition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no MFA Server Definitions defined to the console or if no MFA Server Definitions are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the MFA Server Definition objects included in the response body or action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 719](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	14	A query parameter contains an invalid value.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/mfa-server-definitions HTTP/1.1
x-api-session: 2q7icwfrl9deu0rxrupb0p6mobicvirs0w4y1bvtfocrz0ix5fl
```

Figure 425. List MFA Server Definitions: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:13:32 GMT
content-type: application/json
content-length: 254
{
  "mfa-server-definitions": [
    {
      "element-uri": "/api/console/mfa-server-definitions/14ef02da-c857-11e9-8189-fa163e8e8d46",
      "name": "MFA server 2"
    },
    {
      "element-uri": "/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46",
      "name": "MFA server 1"
    }
  ]
}
```

Figure 426. List MFA Server Definitions: Response

Get MFA Server Definition Properties

The Get MFA Server Definition Properties operation retrieves the properties of a single MFA Server Definition object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties of the MFA Server Definition object as defined in the “Data model” on page 718. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current property values of the MFA Server Definition object specified by *{mfa-server-definition-id}*.

On successful execution, the current values of the properties as defined in “Data model” on page 718 for the MFA Server Definition object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing MFA Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multifactor Authentication** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 721](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The element ID in the request URI <i>{mfa-server-definition-id}</i> does not designate an existing MFA Server Definition object or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 29t78zp0syhk4lhe5xyx6ib53qloudk7zthe6dsd5nnefcvkxa
```

Figure 427. Get MFA Server Definition Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:15:43 GMT
content-type: application/json
content-length: 364
{
  "class": "mfa-server-definition",
  "description": "Company primary IBM MFA server",
  "element-id": "3007226a-c856-11e9-9df4-fa163e8e8d46",
  "element-uri": "/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46",
  "hostname-ipaddr": "mfa1.internal.example.com",
  "name": "MFA server 1",
  "parent": "/api/console",
  "port": 6789,
  "replication-overwrite-possible": false
}
```

Figure 428. Get MFA Server Definition Properties: Response

Update MFA Server Definition Properties

The Update MFA Server Definition Properties operation updates the properties of a single MFA Server Definition object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the “Data model” on page 718 for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the MFA Server Definition object specified by *{mfa-server-definition-id}*.

On successful execution, the MFA Server Definition object has been updated with the supplied property values and status code 204 (No Content) is returned without supplying a response body.

The URI path must designate an existing MFA Server definition object; otherwise, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multifactor Authentication** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in “Request body contents” on page 723. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which Property Change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **MFA Server Definition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The element ID in the request URI <i>{mfa-server-definition-id}</i> does not designate an existing MFA Server Definition object or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 5oqk73rga8zzqrqne7bmpz1scpppu1onkno12jbgt2q095fjvw
content-type: application/json
content-length: 14
{
  "port":6790
}
```

Figure 429. Update MFA Server Definition Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:18:22 GMT
content-type: application/json

<No response body>
```

Figure 430. Update MFA Server Definition Properties: Response

Create MFA Server Definition

The Create MFA Server Definition operation creates an MFA Server Definition object with the given properties.

HTTP method and URI

```
POST /api/console/mfa-server-definitions
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The value to be set as the MFA Server Definition's name property.
description	String	Optional	The value to be set as the MFA Server Definition's description property.

Field name	Type	Rqd/Opt	Description
hostname-ipaddr	String/Hostname, String/IPV4 Address, or String/IPV6 Address	Required	The value to be set as the MFA Server Definition's hostname-ipaddr property.
port	Integer	Optional	The value to be set as the MFA Server Definition's port property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
element-uri	String/URI	Canonical URI path of the new MFA Server Definition object.

Description

This operation creates a new MFA Server Definition object.

On successful execution of this operation the MFA Server Definition is created using the values specified in the request body. The URI of the new MFA Server Definition is provided in the response body and in a **Location** response header.

The request body is validated against the schema described in the [“Request body contents” on page 724](#). If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because the specified name is not unique. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentication** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 725](#), and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 353. Create MFA Server Definition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	An MFA Server Definition with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/console/mfa-server-definitions HTTP/1.1
x-api-session: 1l0nki6fyqq4zuue0lxmlahx8tnchoihm2w27cin0blkwma516
content-type: application/json
content-length: 121
{
  "description": "Company primary IBM MFA server",
  "hostname-ipaddr": "mfa1.internal.example.com",
  "name": "MFA server 1"
}
```

Figure 431. Create MFA Server Definition: Request

```
201 Created
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:06:50 GMT
content-type: application/json
content-length: 90
{
  "element-uri": "/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46"
}
```

Figure 432. Create MFA Server Definition: Response

Delete MFA Server Definition

The Delete MFA Server Definition operation deletes an MFA Server Definition object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object to be deleted.

Description

This operation removes the MFA Server Definition specified by *{mfa-server-definition-id}* from the console.

On successful execution, the MFA Server Definition object has been removed and no response body is provided. An Inventory Change notification is emitted asynchronously to this operation.

The URI path must designate an existing MFA Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentication** task; otherwise, status code 403 (Forbidden) is returned.

If any user or user template is defined to use the specified MFA Server Definition, the request fails and HTTP status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The element ID in the request URI <i>{mfa-server-definition-id}</i> does not designate an existing MFA Server Definition object.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this MFA Server Definition.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
DELETE /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 5qlow7gr99vbc4gyidnc769ipivfsh6z3z2aj54ydw9z0sbfn
```

Figure 433. Delete MFA Server Definition: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:20:28 GMT

<No response body>
```

Figure 434. Delete MFA Server Definition: Response

Inventory service data

Information about the MFA Server Definitions managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for MFA Server Definition objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"console"** are to be included.

For each MFA Server Definition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get MFA Server Definition Properties` operation. That is, the data provided is the same as would be provided if a `Get MFA Server Definition Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single MFA Server Definition. This object would appear as one array entry in the response array:

```
{
  "class": "mfa-server-definition",
  "description": "Backup MFA server",
  "element-id": "14ef02da-c857-11e9-8189-fa163e8e8d46",
  "element-uri": "/api/console/mfa-server-definitions/14ef02da-c857-11e9-8189-fa163e8e8d46",
  "hostname-ipaddr": "mfa2.internal.example.com",
  "name": "MFA server 2",
  "parent": "/api/console",
  "port": 5402,
  "replication-overwrite-possible": false
}
```

Figure 435. MFA Server Definition object: Sample inventory data

Group Object

The Hardware Management Console provides a fixed set of system-defined groups to which managed objects of certain types automatically belong, as members. For example, defined CPCs are automatically members of the CPC group. By their nature, the members of the system-defined groups are obtainable via list operations of the appropriate API. For example, all the CPCs managed by a Hardware Management Console can be obtained via a `List CPCs` operation. Therefore, list operations for system-defined groups are unnecessary. By their nature, the existence of a system-defined group and its content (members) is implicit. Therefore, create/delete operations for system-defined groups are both unnecessary and inappropriate.

These system-defined groups are distinct from user-defined ("custom") groups. The latter are explicitly created by users for their own purposes: for example, it may be convenient for management purposes to take some proper subset of the members of the system-defined CPC group as a user-defined group of CPCs. User-defined groups may be homogeneous (all members of the same managed object type, as in this previous example), but need not be.

A Group object represents one or more managed objects which are called group members. Each member is of some object type: CPC, Logical Partition, etc. Note that groups may be heterogeneous (with member objects of differing types), and may even have other groups as members.

Users may define groups in one of two ways:

1. by use of a pattern-match expression to implicitly define membership (pattern-matching group)
2. by explicitly choosing members.

This API can be used to view/manage custom groups, and membership within these groups. The latter is subject to restrictions, based on which of the two fundamentally different means of definition the user employed:

- If pattern-matching was specified, then group membership is "implicit". In this case, operations to add/remove a member are unnecessary (simply create/delete the managed object, itself, using the appropriate API operation). Accordingly, member-management operations are not supported for groups using pattern-matching.
- If pattern-matching was not specified, then group membership is "explicit", and in this case operations to add/remove group members are both useful and appropriate. Accordingly, for custom groups not based on pattern-matching, member-management operations are supported. Note that such operations do not affect the member object itself, only its group membership status.
- When groups are defined using pattern-matching, the types of managed objects to which pattern-matching is applied must be explicitly specified. Regardless of the POSIX regular expression specified as the match pattern, managed objects whose names match the pattern but who are not of the specified object type(s) are not considered to be members of the group.
- Groups are not intrinsically ordered in any way, nor are members within a given group. List-oriented operations therefore do not return ordered results.

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 70, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 68

The following class-specific specializations apply to the other base managed object properties:

Table 354. Group object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Group object, of the form <code>/api/groups/{group-id}</code> where <code>{group-id}</code> is the value of the object-id property of the Group object.
parent	—	String	This property is always a null object.
class	—	String	The class of a Group object is "group" .
name	(ro)	String	The group name specified by the user when the group was created
description	(ro)	String	The description specified by the user when the group was created, or if none was provided, the IBM provided caption text.
replication-overwrite-possible	—	Boolean	Indicates whether this object is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties:

Table 355. Group object: class specific additional properties		
Name	Type	Description
match-info	match-info object	A nested object which pertains to pattern-matching groups only, as described in the next table. An empty value is returned for groups which do not use pattern-matching.

The match-info object contains the following fields:

Table 356. match-info object properties

Name	Type	Description
pattern	String	A regular expression used to define membership for pattern-matching groups. This field has no length limitations.
types	Array of String Enum	Specifies the type(s) of objects that are eligible for membership in pattern-matching groups. One or more of the following: • "custom-groups" - zManager API objects of class "group" • "defined-cpc" - zManager API objects of class "cpc" • "logical-partition" - zManager API objects of class "logical-partition"

List Custom Groups

The List Custom Groups operation lists the custom groups which are visible to the API user.

HTTP method and URI

GET /api/groups

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
groups	Array of group-info objects	Array of nested objects which identify groups that are visible to the API user.

Each nested group-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The value of the Group object's object-uri property.
name	String	The value of the Group object's name property.

Description

This operation lists the Group objects which are visible to the API user. Only groups to which the caller has authorization will be returned.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section. If no groups exist, or if no groups are visible to the API user, HTTP status code 200 (OK) is returned, along with an empty response body.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Group object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described [“Response body contents” on page 730](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/groups HTTP/1.1
x-api-session: 4ipkcgbjpy5kocelt652l3dvv85gi81iqy5bz8yrpt6vtrt8ks
```

Figure 436. List Custom Groups: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:02:42 GMT
content-type: application/json; charset=UTF-8
content-length: 283
{
  "groups": [
    {
      "name": "Finance department CPCs",
      "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341"
    },
    {
      "name": "Test Group",
      "object-uri": "/api/groups/febde5ab-a4a6-35bf-9e01-83aae59d7e52"
    }
  ]
}
```

Figure 437. List Custom Groups: Response

Get Custom Group Properties

The Get Custom Group Properties operation retrieves the properties of a single Group object that is designated by the `{group-id}`.

HTTP method and URI

```
GET /api/groups/{group-id}
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Response body contents

On success, HTTP status code 200 (OK) is returned and the response body contains an object that provides the current values of the properties for the Group object as defined in [“Data model” on page 729](#). Field names and data types in the object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the Group object designated by *{group-id}*.

The URI path *{group-id}* must designate an existing Group object.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Group object designated by *{group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 732](#).

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object-id in the URI (<i>{group-id}</i>) does not designate an existing group, or the API user does not have sufficient access (as described above).

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341 HTTP/1.1
x-api-session: 42e6t4chltpvd6l4l6lwi3111tf7fv2hes80hjqs3inv7cp
```

Figure 438. Get Custom Group Properties: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:45 GMT
content-type: application/json;charset=UTF-8
content-length: 250
{
  "class": "group",
  "description": "Spacely Sprockets Web Servers",
  "replication-overwrite-possible": false,
  "is-locked": false,
  "match-info": {},
  "name": "SS-Web-Servers",
  "object-id": "ee2782af-dd98-3ec0-bc2d-cfe2e9154341",
  "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341",
  "parent": null
}

```

Figure 439. Get Custom Group Properties: Response

Create Custom Group

Use the Create Custom Group operation to create a custom group.

HTTP method and URI

POST /api/groups

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String	Required	The name for the new custom Group object
description	String	Optional	The description for the new custom Group object
match-info	match-info object	Optional	A nested object describing the pattern match. If not provided, this is not a pattern-match custom group. Refer to “Class specific additional properties” on page 729 for details.

Response body contents

Field name	Type	Description
object-uri	String	The object URI of the new custom group.

Description

Group objects are programmatically identified by object-id and not by name. To avoid the confusion which might result from allowing redundant names, the **name** property is required for this operation, and the (case-sensitive) value supplied for the name property must be distinct from that of all currently-existing Group objects. In keeping with restrictions imposed by the Hardware Management Console's Graphical User Interface (GUI), the following set of names is also not allowed:

- the current name of the Hardware Management Console
- the GUI View names {"Groups", "Exceptions", "Active Tasks", "Console Actions", "Task List", "Books", "Help", "Ensemble"}

On success, a custom group managed object is created reflecting the Request Body contents and HTTP status code 201 (Created) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Grouping** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in [“Response body contents” on page 733](#). In addition, the **Location** response header contains the URI of the newly created object.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	261	One of the following errors was detected: • The pattern string specified in match-info is not valid. This must be a non-empty string which is a valid regular expression. • One or more of the types specified in match-info is invalid. At least one type must be specified, and all must be values as documented for the match-info types property.
	290	The requested name is either reserved or already in use.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/groups HTTP/1.1
x-api-session: 42r6t4chltpvd6l4l61wi3111tf7fv2hes80hjqs3inv7cp
content-type: application/json
content-length: 74
{
  "description": "Spacely Sprockets Web Servers",
  "name": "SS-Web-Servers"
}
```

Figure 440. Create Custom Group: Request

```
201 Created
server: zSeries management console API web server / 1.0
location: /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:44 GMT
content-type: application/json; charset=UTF-8
content-length: 65
{
  "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341"
}
```

Figure 441. Create Custom Group: Response

Delete Custom Group

Use the Delete Custom Group operation to delete a custom group.

HTTP method and URI

```
DELETE /api/groups/{group-id}
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Description

If successful, the custom group managed object designated by *{group-id}* is deleted.

If *{group-id}* does not identify an existing custom group, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the custom group designated by *{group-id}*
- Action/task permission for the **Grouping** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The URI's <i>{group-id}</i> does not designate an existing custom Group object, or the API user does not have object-access to the group.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341 HTTP/1.1
x-api-session: 42r6t4chltpvd6l4l6lwi3111tf7fv2hes80hjqs3inv7cp
```

Figure 442. Delete Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:45 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 443. Delete Custom Group: Response

Add Member to Custom Group

Use the Add Member to Custom Group operation to add a member to a custom group.

HTTP method and URI

```
POST /api/groups/{group-id}/operations/add-member
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
object-uri	String	Required	The object URI of the object to be added to the group

Description

If successful, the managed object designated in the request body attains membership in the custom group identified by *{group-id}*.

The operation is subject to the following restrictions:

- The designated managed object must exist and must not already be a member of the group identified by *{group-id}*
- The group identified by *{group-id}* must be a custom group defined without a pattern-matching specification.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the custom Group object designated by *{group-id}*
- Object-access permission to the object designated by the request body
- Action/task permission for the **Grouping** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	291	The designated managed object is already a member of the custom group identified by <i>{group-id}</i> .
	294	The group identified by <i>{group-id}</i> was defined using pattern-matching.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	293	The addition of the member to the custom group designated by the URI (<i>{group-id}</i>) would introduce a circular reference, which is not permitted.
404 (Not Found)	1	The URI's <i>{group-id}</i> does not designate an existing custom Group object, or the API user does not have object-access to the group.
	2	The request body does not designate an existing managed object, or the API user does not have sufficient access to the managed object.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/operations/add-member HTTP/1.1
x-api-session: 42r6t4chltpvd6l4l6lwi3111tf7fv2hes80hjqs3inv7cp
content-type: application/json
content-length: 78
{
  "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
}
```

Figure 444. Add Member to Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:44 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 445. Add Member to Custom Group: Response

Remove Member from Custom Group

Use the Remove Member from Custom Group operation to remove a member from a custom group.

HTTP method and URI

```
POST /api/groups/{group-id}/operations/remove-member
```

In this request, the URI variable *{group-id}* is the object ID of the group

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
object-uri	String	Required	The object URI of the object to be removed from the group

Description

The managed object designated in the request body relinquishes its membership in the custom group identified by *{group-id}*.

The operation is subject to the following restrictions:

- The managed object designated in the request body must currently be a member of the group identified by *{group-id}*.
- The group identified by *{group-id}* must be a custom group defined without a pattern-matching specification

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the custom Group object designated by *{group-id}*
- Object-access permission to the object designated by the request body
- Action/task permission for the **Grouping** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	291	The designated managed object is not a member of the custom group identified by <i>{group-id}</i> .
	294	The group identified by <i>{group-id}</i> was defined using pattern-matching.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The URI's <i>{group-id}</i> does not designate an existing custom Group object, or the API user does not have object-access to the group.
	2	The request body does not designate an existing managed object, or the API user does not have sufficient access to the managed object.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/operations/remove-member HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqs3inv7cp
content-type: application/json
content-length: 78
{
  "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
}
```

Figure 446. Remove Member from Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:45 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 447. Remove Member from Custom Group: Response

List Custom Group Members

Use the List Custom Group Members operation to list custom group members.

HTTP method and URI

```
GET /api/groups/{group-id}/members
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Response body contents

Field name	Type	Description
members	Array of nested objects	Array of nested objects which identify members of the group designated by <i>{group-id}</i> .

Each nested member object contains the following fields:

Field name	Type	Description
object-uri	String/URI	The value of the member object's object-uri property.

Field name	Type	Description
name	String	The value of the member object's name property.

Description

This operation lists the members of the Group object designated by *{group-id}*. The results of this operation only include references to member objects for which the API user has object-access authority.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section. If the group currently has no members, HTTP status code 200 (OK) is returned, along with an empty response body.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Group object.
- Object-access permission to each member object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 739](#).

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The group designated by the URI (<i>{group-id}</i>) does not exist, or the API user does not have sufficient access (as described above).

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/members HTTP/1.1
x-api-session: 42r6t4chltpvd6l4l61wi3111tf7fv2hes80hjqs3inv7cp
```

Figure 448. List Custom Group Members: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:45 GMT
content-type: application/json; charset=UTF-8
content-length: 213
{
  "members": [
    {
      "name": "SS-Web-Svr-1",
      "object-uri": "/api/logical-partitions/576569dc-0db7-11e1-b1f1-f0def14b63af"
    },
    {
      "name": "SS-Web-Svr-2",
      "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
    }
  ]
}

```

Figure 449. List Custom Group Members: Response

Inventory service data

Information about custom groups can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Group objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of the class **"group"** are to be included. An entry for a particular group is included only if the API user has access permission to that object as described in the [Get Custom Group Properties](#) operation.

For each Group object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for ["Get Custom Group Properties"](#) on page 731. That is, the data provided is the same as would be provided if a [Get Custom Group Properties](#) operation were requested targeting this object.

CPC object

A CPC object represents a single Central Processor Complex (CPC) that is being managed by an HMC.

Data model

For definitions of the qualifier abbreviations in the following tables, see ["Property characteristics"](#) on page 68.

This object includes the properties defined in the ["Base managed object properties schema"](#) on page 70, with the following class-specific specialization:

Table 357. CPC object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the CPC object, of the form <code>/api/cpcs/{cpc-id}</code> where <code>{cpc-id}</code> is the value of the object-id property of the CPC object.
parent	(pc)	String/ URI	A CPC object has no parent, so this property is always a null object.
class	—	String	The class of a CPC object is "cpc" .

Table 357. CPC object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
name	(ro)	String (1-8)	The CPC name
description	(ro) if dpm-enabled is false or (w)(pc) if dpm-enabled is true	String (0-1024)	The descriptive text associated with this CPC object. This property is a writable, user-supplied value only when this CPC is enabled for DPM.
status	(sc)	String Enum	The current operational status of the CPC object. One of: • "active" - the CPC is active. This status is applicable only when dpm-enabled is true. • "operating" the CPC is operational. This status is applicable only when dpm-enabled is false. • "not-communicating" - the CPC is not communicating with the HMC. With the exceptions of object-uri, parent, class, name, and status, the values of CPC properties are unpredictable unless stated otherwise in this data model. • "exceptions" - the CPC has one or more unusual conditions • "status-check" - the SE is not communicating with the CPC • "service" - the CPC has been placed in service mode • "not-operating" - the CPC is not operational • "no-power" - the CPC has no power • "service-required" - the CPC is operating on the last redundant part of a particular type • "degraded" - one or more of the CPC elements are degraded.
acceptable-status	(w)(pc)	Array of String Enum	The set of operational status values in which the CPC object can exist and be considered in an acceptable (not alert causing) state. One or more of the values listed for the status property.

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties.

Table 358. CPC object: class specific additional properties

Name	Qualifier	Type	Description
se-version	(pc)	String (1-8)	The current release level of the primary SE internal code. For example, "2.11.1". Note that the alternate SE is normally at the same level, except when installing new internal code levels.
has-hardware-messages	(pc)	Boolean	The CPC object has hardware messages (true), or does not have hardware messages (false).

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
iml-mode	(pc)	String Enum	The Initial Microcode Load (IML) mode type of the CPC object. One of: • "not-set" - the CPC is not IMLed or the CPC is not communicating with the HMC. • "esa390" - the CPC is in ESA/390 mode • "lpar" - the CPC is in logical partition mode • "esa390-tpf" - the CPC is in ESA/390 TPF mode • "dpm" - the CPC is in DPM mode
dpm-enabled	—	Boolean	The CPC is enabled for DPM (true) or not (false).
auto-start-list	(pc)	Array of objects	The array of nested auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. This is an ordered array. The partitions or partition groups that are earlier in the array are started before those that are later in the array. The order in which the individual partitions within a given partition group are started is not specified. An empty array indicates that no partitions are automatically started. The auto-start list is specified via the Set Auto-Start List operation. This property is only present when dpm-enabled is true.
is-cpacf-enabled	—	Boolean	Whether Central Processor Assist for Cryptographic Functions (CPACF) is enabled (true) for the CPC or not (false).
next-activation-profile-name	(w)(pc)	String (1-16)	For a CPC that is not enabled for DPM, the name of the activation profile to be used on the next activation of the CPC. Note: This property is not returned on a Get CPC Properties operation when the CPC is enabled for DPM. Note: This property is not permitted on an Update CPC Properties operation when the CPC is enabled for DPM.
last-used-activation-profile	(pc)	String (0-16)	For a CPC that is not enabled for DPM, the name of the activation profile used on the last activation of the CPC or an empty string. Note: This property is not returned on a Get CPC Properties operation when the CPC is enabled for DPM.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
last-used-iocds	(pc)	String (0-3)	For a CPC that is not enabled for DPM, the name of the IOCDS most recently used by the CPC or an empty string if the CPC has not been IMLed. Note: This property is not returned on a <code>Get CPC Properties</code> operation when the CPC is enabled for DPM.
machine-model	(pc)	String (1-3)	The model of the machine containing this CPC. For example, "M15".
machine-type	(pc)	String (1-4)	The type of the machine containing this CPC. For example, "2817".
machine-serial-number	(pc)	String (1-12)	The serial number of the machine containing this CPC. For example, "00 - SP1D92B".
cpc-serial-number	—	String (1-12)	The serial number of the CPC. For example, "00000SP1D92B".
cpc-node-descriptor	(pc)	String (2)	The hexadecimal number mapped to the device location of the CPC. This property identifies the CPC's relative order among other CPCs, if any, in the machine. For example, "00".
is-cbu-installed	—	Boolean	The Capacity Backup Upgrade (CBU) facility is installed (true), or not installed (false). Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-cbu-enabled	—	Boolean	CBU is enabled (true), or is not enabled (false). Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-cbu-activated	—	Boolean	CBU is activated (true), or is not activated (false). Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-real-cbu-available	—	Boolean	Real CBU is available (true), or not available (false). Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
cbu-activation-date	—	Timestamp	The date of CBU activation. Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
cbu-expiration-date	—	Timestamp	The date of CBU expiration. Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
cbu-number-of-tests-left	—	Integer	The number of CBU tests left. Note: if status is " not-communicating ", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-secure-execution-enabled ³	(pc)	Boolean	If true , Secure Execution for Linux is enabled. If false , Secure Execution for Linux is disabled.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
is-global-key-installed³	(pc)	Boolean	If true , the Global key is installed. If false , the Global key is not installed. If Secure Execution for Linux is not enabled, this property will be returned as a null.
is-host-key-installed³	(pc)	Boolean	If true , the Host key is installed. If false , the Host key is not installed. If Secure Execution for Linux is not enabled, this property will be returned as a null.
is-service-required	—	Boolean	Whether the CPC is operating using the last redundant part of a particular type (true) or not (false). If true, repairs should be made before additional parts fail that would make this CPC non-operational. Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the <i>Support Element Operations Guide</i> .
degraded-status	(pc)	Array of String Enum	The set of degraded status values. If the CPC is not degraded, this property contains "not-degraded" as the only value. Otherwise, this property contains one or more of the following: • "memory" - loss of memory • "io" - loss of I/O channels • "node" - one or more books are no longer functioning • "ring" - the ring connecting the book is open • "cbu" - CBU resources have expired • "mru" - cycle time reduction due to an MRU problem • "ambient-temp" - cycle time reduction due to a temperature problem • "iml" - CPC was IMLed during cycle time reduction.
processor-running-time-type	(w)	String Enum	For a CPC that is not enabled for DPM, denotes how the processor-running-time property value was determined. One of: • "system-determined" - the processor running time is dynamically determined by the system • "user-determined" - the processor running time is set to a constant value. Note: If iml-mode is not "lpar", a null object is returned. Note: This property is not returned on a Get CPC Properties operation when the CPC is enabled for DPM. Note: This property is not permitted on an Update CPC Properties operation when the CPC is enabled for DPM.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
processor-running-time	(w)	Integer	For a CPC that is not enabled for DPM, the amount of continuous time, in milliseconds, allowed for logical processors to perform jobs on shared processors for the CPC object. Note: a null object is returned if the impl-mode is not "lpar" or processor-running-time-type is "system-determined". Note: This property is not returned on a Get CPC Properties operation when the CPC is enabled for DPM. Note: This property is not permitted on an Update CPC Properties operation when the CPC is enabled for DPM.
does-wait-state-end-time-slice	(w) or — if se-version is " 2.14.0 " or later	Boolean	For a CPC that is not enabled for DPM, Logical Partitions of the CPC should lose their share of running time when they enter a wait state (true), or should not lose their share of running time when they enter a wait state (false). Note: a null object is returned if the impl-mode is not "lpar" or processor-running-time-type is "system-determined". Note: This property is not returned on a Get CPC Properties operation when the CPC is enabled for DPM. Note: This property is not permitted on an Update CPC Properties operation when the CPC is enabled for DPM. Note: When se-version is "2.14.0" or later, this property is not permitted on an Update CPC Properties operation, and its value is always false.
is-on-off-cod-installed	—	Boolean	On/Off Capacity on Demand is installed for the CPC object (true), or is not installed (false). Note: if status is "not-communicating", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-on-off-cod-enabled	—	Boolean	On/Off CoD is enabled (true), or is not enabled (false). Note: if status is "not-communicating", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
is-on-off-cod-activated	—	Boolean	On/Off CoD is currently activated for the CPC object (true), or is not currently activated (false). Note: if status is "not-communicating", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
on-off-cod-activation-date	—	Timestamp	The date when On/Off CoD was activated. Note: if status is "not-communicating", a null object is returned. Refer to the <i>Capacity On Demand User's Guide</i> for details.
software-model-permanent	(pc)	String (1-3)	The software model based on the permanent processors. For example, "700".

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
software-model-permanent-plus-billable	(pc)	String (1-3)	The software model based on the permanent plus billable processors. For example, "700".
software-model-permanent-plus-temporary	(pc)	String (1-3)	The software model based on the permanent plus all temporary processors. For example, "700".
msu-permanent	—	Integer	The MSU value associated with the software model based on the permanent processors.
msu-permanent-plus-billable	—	Integer	The MSU value associated with the software model based on the permanent plus billable processors.
msu-permanent-plus-temporary	—	Integer	The MSU value associated with the software model based on the permanent plus all temporary processors.
processor-count-general-purpose	—	Integer	The count of active general purpose processors.
processor-count-service-assist	—	Integer	The count of active service assist processors.
processor-count-aap	—	Integer	The count of active IBM zEnterprise Application Assist Processor (zAAP) processors.
processor-count-ifl	—	Integer	The count of active Integrated Facility for Linux (IFL) processors.
processor-count-icf	—	Integer	The count of active Internal Coupling Facility (ICF) processors.
processor-count-iip	—	Integer	The count of active IBM z Integrated Information Processor (zIIP) processors.
processor-count-cbp²	—	Integer	The count of active Container Based Processors (CBP).
processor-count-defective	—	Integer	The count of defective processors. Includes all processor types.
processor-count-spare	—	Integer	The count of spare processors. Includes all processor types.
processor-count-pending	—	Integer	The combined number of processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records.
processor-count-pending-general-purpose	—	Integer	The number of general purpose processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is "not-communicating" , a null object is returned.
processor-count-pending-service-assist	—	Integer	The number of service assist processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is "not-communicating" , a null object is returned.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
processor-count-pending-aap	—	Integer	The number of Application Assist processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is " not-communicating ", a null object is returned.
processor-count-pending-ifl	—	Integer	The number of Integrated Facility for Linux processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is " not-communicating ", a null object is returned.
processor-count-pending-icf	—	Integer	The number of Integrated Coupling Facility processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is " not-communicating ", a null object is returned.
processor-count-pending-iip	—	Integer	The number of z Integrated Information Processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if status is " not-communicating ", a null object is returned.
processor-count-pending-cbp²	—	Integer	The number of Container Based Processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: If status is " not-communicating ", a null object is returned.
has-temporary-capacity-change-allowed	—	Boolean	Whether API applications are allowed to make changes to temporary capacity (true), or not (false).
ec-mcl-description	—	ec-mcl-description object	Describes the Engineering Change (EC) and MicroCode Level (MCL) for the CPC object. An empty object is returned if the information is unavailable from the SE. Refer to the description of the ec-mcl-description object for details.
has-automatic-se-switch-enabled	—	Boolean	Automatic switching between primary and alternate Support Elements is enabled for the CPC object (true), or is not enabled (false). Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the <i>Support Element Operations Guide</i> .
stp-configuration	—	stp-config object	Describes the Server Time Protocol (STP) configuration. Refer to the description of the stp-config object for details. Note: if the required feature(s) are not installed, the property is not returned.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
lan-interface1-type	(pc)	String Enum	The adapter type of the Support Element's LAN interface 1. One of the following: • "ethernet" • "token-ring" • "unknown"
lan-interface1-address	(pc)	String (1-12)	The MAC address of the Support Element's LAN interface 1.
lan-interface2-type	(pc)	String Enum	The adapter type of the Support Element's LAN interface 2. One of the following: • "ethernet" • "token-ring" • "unknown"
lan-interface2-address	(pc)	String (1-12)	The MAC address of the Support Element's LAN interface 2.
network1-ipv4-mask	(pc)	String (1-15)	The network IP mask value.
network1-ipv4-pri-ipaddr	(pc)	String IPV4 address	The primary IPv4 address or a null object if not configured.
network1-ipv4-alt-ipaddr	(pc)	String IPV4 address	The alternate IPv4 address or a null object if not configured.
network1-ipv6-info	—	Array of ipv6-info objects	A list of objects describing the Support Element's IPv6 network connections. If no IPv6 connections are defined, an empty list is returned.
network2-ipv4-mask	(pc)	String (1-15)	The network IP mask value.
network2-ipv4-pri-ipaddr	(pc)	String IPV4 address	The primary IPv4 address or a null object if not configured.
network2-ipv4-alt-ipaddr	(pc)	String IPV4 address	The alternate IPv4 address or a null object if not configured.
network2-ipv6-info	—	Array of ipv6-info objects	A list of objects describing the Support Element's IPv6 network connections. If no IPv6 connections are defined, an empty list is returned.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
hardware-messages	(c)(pc)	Array of hardware-message objects	The complete list of all CPC hardware messages, each identified by its URI. This list corresponds to the list provided by the <code>List CPC Hardware Messages</code> operation. If the CPC has no hardware messages, then an empty array is provided. The list of returned hardware messages can change as a result of new messages being dynamically added or removed by the infrastructure or due to hardware messages being deleted via the <code>Delete CPC Hardware Message</code> operation. Note: This property is not returned by the <code>Get CPC Properties</code> operation, and only sessions associated with an HMC user with permission to the <code>Hardware Messages</code> task will receive a property-change notification for this property.
storage-total-installed	—	Long	Amount of installed storage, in megabytes.
storage-hardware-system-area	—	Long	Amount of storage, in megabytes, reserved for the base hardware system area (HSA).
storage-customer	—	Long	Amount of storage, in megabytes, for use by the customer.
storage-customer-central	—	Long	Amount of storage, in megabytes, which is the central storage in use across the active partitions.
storage-customer-expanded	—	Long	Amount of storage, in megabytes, which is the expanded storage in use across the active partitions. When the iml-mode is "dpm" this property will be set to 0.
storage-customer-available	—	Long	Amount of storage, in megabytes, which is not in use.
storage-vfm-increment-size¹	—	Long	The increment size of any IBM Virtual Flash Memory (VFM) storage property, in gigabytes (GB), on this CPC and its logical partitions or partitions.
storage-vfm-total¹	—	Long	The total amount of VFM storage, in gigabytes (GB), installed on this CPC. The valid value should be a multiple of the value indicated on the storage-vfm-increment-size property for this CPC.
maximum-hipersockets	—	Integer	The maximum number of HiperSocket adapters that may be created for the CPC when the CPC is enabled for DPM. This property will be omitted if dpm-enabled is false.
maximum-alternate-storage-sites	—	Integer	The maximum number of alternate storage-site instances that may be added to the FICON configuration associated with this CPC. This property will be omitted if dpm-enabled is false.

Table 358. CPC object: class specific additional properties (continued)

Name	Qualifier	Type	Description
available-features-list	—	Array of cpc-feature-info objects	The list of optional features or behavior supported by this CPC. If the CPC has no optional features, then an empty array is provided.
maximum-partitions	—	Integer	The maximum number of partitions that this CPC supports. This property will be omitted if dpm-enabled is false .
Notes: 1. On a Get request, this property is returned only when the SE version is 2.14.0 or later. 2. On a Get request, this property is returned only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. 3. On a Get request, this property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.			

Table 359. ipv6-info object properties

Name	Type	Description
type	String Enum	The IPv6 scope. One of the following values: • "link-local" • "static" • "auto"
prefix	Integer	The number of leading bits of the IPv6 address that represent the network prefix
pri-ip-address	String IPv6 address	The primary IPv6 address
alt-ip-address	String IPv6 address	The alternate IPv6 address or a null object if not configured

Table 360. hardware-message object properties

Name	Type	Description
element-uri	String/ URI	The canonical URI path of the CPC hardware message. The URI is in the following form: /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}, where {hardware-message-id} is the value of the element-id property of the hardware message.
element-id	String (36)	The unique identifier for the hardware message. The element-id is in the form of a UUID.
parent	String/ URI	The parent of a CPC hardware message is the CPC object. The parent value is the canonical URI path for the CPC.
class	String	The class of a hardware message object is " hardware-message ".

Table 360. hardware-message object properties (continued)

Name	Type	Description
timestamp	Timestamp	The timestamp represents the date and time when the hardware message was created.
service-supported	Boolean	Indicates whether or not this hardware message represents a problem for which service may be requested. True is returned if this hardware message supports service, or false is returned if it does not.
text	String	The text of the hardware message.
details	Object	A hardware-message-details-object if there are hardware message details, or null if there are no hardware message details.

Table 361. cpc-feature-info object properties

Name	Type	Description
name	String Enum	The name of the feature. One of: • "dpm-storage-management" - Indicates that the CPC supports Storage Groups and FICON storage resources. FCP and FICON storage resources are defined in Storage Groups, which are then attached to this CPC's partitions. If the CPC does not have this feature, FCP storage resources are represented by HBAs, which must be directly attached to this CPC's Partitions. This feature is applicable only for CPCs enabled for DPM. See Chapter 6, "Firmware features," on page 73 for a list of operations that are affected for each of these features.
description	String	A brief description of the feature.
state	Boolean	Indicates if the feature is currently enabled (true) or disabled (false) for this CPC.

auto-start-entry object

An auto-start-entry object specifies either a partition or a group of partitions that are automatically started when the CPC is started. Every auto-start-entry object contains the following base properties in addition to the type-specific properties:

Table 362. auto-start-entry object base properties

Name	Type	Description
post-start-delay	Integer	Amount of time, in seconds, to wait before starting the next partition or group of partitions. Must be greater than or equal to 0.
type	String Enum	The type of auto-start-entry. One of • "partition" - this entry identifies a single partition. • "partition-group" - this entry identifies a partition group.

An auto-start-entry object with a **type** value of **"partition"** identifies a single partition. In addition to the base properties, it contains the following type-specific properties:

*Table 363. auto-start-entry object type-specific properties when **type** value is "**partition**"*

Name	Type	Description
partition-uri	String/ URI	The URI of the partition to start.

An auto-start-entry object with a **type** value of "**partition-group**" identifies a group of partitions. In addition to the base properties, it contains the following type-specific properties:

*Table 364. auto-start-entry object type-specific properties when **type** value is "**partition-group**"*

Name	Type	Description
name	String (1-64)	The name of this partition group. This name must be unique among all partition groups of this CPC. The length and character requirements on this property are the same as those of the name property described in the "Base managed object properties schema" on page 70.
description	String (0-1024)	The Description for this partition group.
partition-uris	Array of String/ URI	The list of one or more partitions to start, represented by their URIs. This list is unordered. The partitions in a group can be started in any order or simultaneously by the system.

Energy management related additional properties

In addition to the properties defined in "Class specific additional properties" on page 742, this object includes the following additional class-specific properties related to energy management. For further explanation of the various states involved, please see "Special states" on page 947.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 68.

Table 365. CPC object: energy management related additional properties

Name	Qualifier	Type	Description
cpc-power-rating	—	Integer	Specifies the maximum power draw in watts (W) of this CPC. This is a calculated value as indicated by the electrical rating labels or system rating plates of the CPC components.
cpc-power-consumption	(mg)	Integer	Specifies the current power consumption in watts (W) for this CPC. The CPC power consumption includes the power consumption of the zCPC and hardware extensions. If the system does not include any hardware extensions, the CPC power consumption will be equal to the zCPC power consumption.

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
cpc-power-saving	—	String Enum	Specifies the current power saving setting of the CPC. Power saving is used to reduce the energy consumption of a system and can be managed in the Set Power Saving operation. The possible settings include: • "high-performance" - The power consumption and performance of the CPC are not reduced. This is the default setting. • "low-power" - All components of the CPC enabled for power saving will have reduced performance to allow for low power consumption. • "custom" - Custom mode indicates that some, but not all, components of the CPC are in the Low power setting. • "not-supported" - Power saving is not supported for this CPC. • "not-available" - Specifies that cpc-power-saving property could not be read from this CPC. • "not-entitled" - The server is not entitled for Power saving.
cpc-power-saving-state	—	String Enum	Specifies the power saving setting of the CPC set by the user. Please note that this property indicates the user setting and may not match the real state of the hardware compared to the cpc-power-saving property. For more information, see "Group power saving" on page 948. The possible settings include: • "high-performance" - Specifies not reducing the power consumption and performance of the CPC. • "low-power" - Specifies low power consumption for all components of the CPC enabled for power saving. • "custom" - Specifies that the CPC does not control the children. This is the default setting. • "not-supported" - Specifies that power saving is not supported for this CPC. • "not-entitled" - Specifies that the server is not entitled to power saving.
cpc-power-save-allowed	—	String Enum	Should be used to determine if a call of the power save operation is currently allowed. If a value other than "allowed" is returned the caller may reckon that the power save operation will fail. The possible settings include: • "allowed" - Alter power save setting is allowed for this CPC • "unknown" - Unknown reason • "not-supported" - Power saving is not supported for this CPC. • "not-entitled" - Specifies the server is not entitled to power capping.

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
cpc-power-capping-state	—	String Enum	Specifies the current power capping setting of the CPC. Power capping is used to limit peak power consumption of a system and can be managed in the Set Power Cap operation. The possible settings include: • "disabled" - The power cap of the CPC is not set and the peak power consumption is not limited. This is the default setting. • "enabled" - All components of the CPC available for power capping will be capped to limit the peak power consumption of the CPC. • "custom" - The components of the CPC can be individually configured for power capping. • "not-supported" - Power capping is not supported for this CPC. • "not-entitled" - The server is not entitled for Power capping.
cpc-power-cap-minimum	—	Integer	Specifies the minimum value for the CPC cap value in watts (W). This is a sum of the component minimum cap values. If the cpc-power-cap-allowed property value is "not-entitled", the value is null. For more information on entitlement, see Chapter 12, “Energy management,” on page 945
cpc-power-cap-maximum	—	Integer	Specifies the maximum value for the CPC cap value in watts (W). This is a sum of the component maximum cap values. If the cpc-power-cap-allowed property value is "not-entitled", the value is null. For more information on entitlement, see Chapter 12, “Energy management,” on page 945
cpc-power-cap-current	—	Integer	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the CPC and is the sum of the component Cap values. If the cpc-power-cap-allowed property value is "not-entitled", the value is null. For more information on entitlement, see Chapter 12, “Energy management,” on page 945
cpc-power-cap-allowed	—	String Enum	Should be used to determine if a call of the power capping operation is currently allowed. If a value other than "allowed" is returned the caller may reckon that the power capping operation will fail. The possible settings include: • "allowed"- Alter power capping setting is allowed for this CPC • "unknown" - Unknown reason • "not-supported" - Power capping is not supported for this CPC. • "not-entitled" - Specifies the server is not entitled to power capping.

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
zcpc-power-rating	—	Integer	Specifies the maximum power draw in watts (W) of this zCPC. This is a calculated value as indicated by the electrical rating labels or system rating plates of the zCPC components.
zcpc-power-consumption	(mg)	Integer	Specifies the current power consumption of the zCPC in watts (W).
zcpc-power-saving	—	String Enum	Specifies the current power saving setting of the zCPC. Power saving is used to reduce the energy consumption of a system and can be managed in the Set Power Saving operation. The possible settings include: • "high-performance" - The power consumption and performance of the zCPC are not reduced. This is the default setting. • "low-power" - The performance of the zCPC is reduced to allow for low power consumption. • "not-supported" - Power saving is not supported for this zCPC. • "not-available" - Specifies that zcpc-power-saving property could not be read for this zCPC. • "not-entitled" - The server is not entitled for Power saving.
zcpc-power-saving-state	—	String Enum	Specifies the power saving setting of the zCPC set by the user. Please note that this property indicates the user setting and may not match the real state of the hardware compared to the zcpc-power-saving property. For more information, see "Group power saving" on page 948. The possible settings include: • "high-performance" - Specifies not reducing the power consumption and performance of the zCPC. This is the default setting. • "low-power" - Specifies low power consumption for all components of the zCPC enabled for power saving. • "not-supported" - Specifies that power saving is not supported for this zCPC. • "not-entitled" - Specifies that the server is not entitled to power saving.

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
zcpcc-power-save-allowed	—	String Enum	Should be used to determine if a call of the power save operation is currently allowed. If a value other than "allowed" is returned the caller may reckon that the power save operation will fail. The possible settings include: • "allowed" - Alter power save is allowed for this zCPC • "unknown" - Unknown reason • "not-entitled" - Specifies the server is not entitled to power save. • "under-group-control" - The zCPC is under group control and cannot be individually altered. • "not-supported" - Power saving is not supported for this zCPC. • "once-a-day-exceeded" - Power saving mode has been entered at some point during the day and will not be allowed again until the next calendar day.
zcpcc-power-capping-state		String Enum	Specifies the current power capping setting of the zCPC. Power capping is used to limit peak power consumption of a system and can be managed in the Set Power Cap operation. The possible settings include: • "disabled" - The power cap of the zCPC is not set and the peak power consumption is not limited. This is the default setting. • "enabled" - The peak power consumption of the zCPC is limited to the Current cap value. • "custom" - The components of the CPC can be individually configured for power capping. • "not-supported" - Power capping is not supported for this zCPC. • "not-entitled" - The server is not entitled for Power capping.
zcpcc-power-cap-minimum	—	Integer	Specifies the minimum value for the zCPC cap value in watts (W). If the zcpcc-power-cap-allowed property value is "not-entitled", the value is null. For more information on entitlement, see Chapter 12, “Energy management,” on page 945
zcpcc-power-cap-maximum	—	Integer	Specifies the maximum value for the zCPC cap value in watts (W). If the zcpcc-power-cap-allowed property value is "not-entitled", the value is null. For more information on entitlement, see Chapter 12, “Energy management,” on page 945

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
zcpc-power-cap-current	—	Integer	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the zCPC. If the zcpc-power-cap-allowed property value is "not-entitled" , the value is null . For more information on entitlement, see Chapter 12, "Energy management," on page 945
zcpc-power-cap-allowed	—	String Enum	Should be used to determine if a call of the power capping operation is currently allowed. If a value other than "allowed" is returned the caller may reckon that the power capping operation will fail. The possible settings include: • "allowed" - Alter power capping is allowed for this zCPC • "unknown" - Unknown reason • "not-entitled" - Specifies the server is not entitled to power cap. • "not-supported" - Power capping is not supported for this zCPC. • "under-group-control" - Power capping is under group control
zcpc-ambient-temperature	(mg)	Float	Specifies the input air temperature in degrees Celsius (°C) as measured by the system.
zcpc-exhaust-temperature	—	Float	Specifies the exhaust air temperature in degrees Celsius (°C) as calculated by the system. This is useful in determining potential hot spots in the data center.
zcpc-humidity	(mg)	Integer	Specifies the amount of water vapor in the air as measured by the system. The humidity sensor gives a reading of the relative humidity of the air entering the system. The recommended long-term relative humidity for a system with an altitude from sea level to 900 meters (2953 feet) is 60%. The range of acceptable relative humidity is 8% - 80%. For more information, see the chapter related to environmental specifications in the <i>Installation Manual for Physical Planning</i>.
zcpc-dew-point	(mg)	Float	Specifies the air temperature in degrees Celsius (°C) at which water vapor will condense into water. This is a calculated value based on the current temperature and relative humidity. Cooling the server to the dew point can result in condensation on critical internal parts, leading to equipment failure, unless the computer room environment is adequately maintained to prevent it. For more information, see the chapter related to environmental specifications in the <i>Installation Manual for Physical Planning</i>.
zcpc-heat-load	(mg)	Integer	Specifies the amount of heat in Btu/hr. removed from the system.

Table 365. CPC object: energy management related additional properties (continued)

Name	Qualifier	Type	Description
zcpc-heat-load-forced-air	—	Integer	Specifies the amount of heat in Btu/hr. removed from the system by forced-air.
zcpc-heat-load-water	—	Integer	Specifies the amount of heat in Btu/hr. removed from the system by chilled water. The value is always 0 on an air cooled system.
zcpc-maximum-potential-power	—	Integer	Specifies the maximum potential power consumption of a system in watts (W). This value is based on the configuration of the system and can be used for power and cooling planning.
zcpc-maximum-potential-heat-load	—	Integer	Specifies the maximum potential heat load of a system in Btu/hr. This value is based on the configuration of the system and can be used for power and cooling planning.
last-energy-advice-time	(pc)	Timestamp	The timestamp of the most recent change to the energy optimization advice for the CPC or null if the timestamp is not available. A value of -1 indicates that no advice has been generated for this CPC. This property is only provided when the associated SE is at version 2.13.1 or later.
zcpc-minimum-inlet-air-temperature	—	Integer	Minimum allowable operating value in degrees Celsius (°C).
zcpc-maximum-inlet-air-temperature	—	Integer	Maximum allowable operating value in degrees Celsius (°C).
zcpc-maximum-inlet-liquid-temperature	—	Integer	Maximum allowable operating value in degrees Celsius (°C).
zcpc-environmental-class	—	String	Class according to ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) Environmental Classes for IT equipment.

List CPC Objects

The `List CPC Objects` operation returns a list of the zManager Web Services API capable CPCs managed by an HMC.

HTTP method and URI

GET /api/cpcs

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
cpcs	Array of cpc-info objects	Array of nested cpc-info objects (described in the next table). If no matching CPC objects are found, an empty array is returned.

Each nested cpc-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the CPC object
name	String	The name of the CPC object
status	String Enum	The current status of the CPC object
has-unacceptable-status	Boolean	The has-unacceptable-status property of the CPC object.
dpm-enabled	Boolean	True if the CPC is enabled for DPM; false otherwise.
se-version	String	The se-version property of the CPC object.

Description

This operation lists the zManager Web Services API capable CPC objects that are managed by this HMC. The object URI, object ID and display name are provided for each CPC returned. CPCs that are not zManager Web Services API capable are not returned.

If the **name** query parameter is specified, the returned list is limited to those CPC objects that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

An object is only included in the list if the API user has object-access permission for that object.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to any CPC object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 760](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs HTTP/1.1
x-api-session: 2jm2h7j25d1e1g5wbygmfriyjiit8tp4iqiw8h09j8kz68i0k6
```

Figure 450. List CPC Objects: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2016 07:18:42 GMT
content-type: application/json; charset=UTF-8
content-length: 725
{
  "cpcs": [
    {
      "dpm-enabled": true,
      "has-unacceptable-status": true,
      "name": "P0LXSM0Z",
      "object-uri": "/api/cpcs/e8753ff5-8ea6-35d9-b047-83c2624ba8da",
      "se-version": "2.13.1",
      "status": "not-operating"
    },
    {
      "dpm-enabled": true,
      "has-unacceptable-status": false,
      "name": "R32",
      "object-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
      "se-version": "2.14.0",
      "status": "operating"
    },
    {
      "dpm-enabled": false,
      "has-unacceptable-status": true,
      "name": "ICHABOD",
      "object-uri": "/api/cpcs/ac15c987-90c6-3526-854e-4c612939260d",
      "se-version": "2.13.1",
      "status": "not-operating"
    }
  ]
}
```

Figure 451. List CPC Objects: Response

Get CPC Properties

The Get CPC Properties operation retrieves the properties of a single CPC object designated by *{cpc-id}*.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
properties	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the CPC object's data model.
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are true and false . The default is false .

Response body contents

On successful completion, the response body provides the current values of the properties for the CPC object as defined in [“Data model” on page 741](#).

Description

Some CPC properties are only available if the HMC is communicating with the SE, and are returned as null objects if the HMC is not communicating with the SE. With the exceptions of **object-uri**, **parent**, **class**, **name**, and **status**, the values of CPC properties are unpredictable unless stated otherwise in the [“Data model” on page 741](#).

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some of this object's property values are periodically fetched from the Support Element and cached on the HMC for quick access by the APIs. Due to the nature of this caching support, the HMC's cached value of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined [“Data model” on page 741](#).

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 762](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	272	Unable to obtain Server Time Protocol (STP) configuration. Retry the request later.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5 HTTP/1.1
x-api-session: 4tcxomq5kfzlv48mn05a8wzmjc1rzhd59tak9zg32jynapvh
```

Figure 452. Get CPC Properties: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 22:11:13 GMT
content-type: application/json; charset=UTF-8
content-length: 4829
{
  "acceptable-status":[
    "operating"
  ],
  "additional-status":"",
  "available-features-list":[
    {
      "description":"The DPM storage management approach in which FCP and FICON storage
resources are defined in Storage Groups, which are attached to Partitions.",
      "name":"dpm-storage-management",
      "state":true
    }
  ],
  "cbu-activation-date":0,
  "cbu-expiration-date":0,
  "cbu-number-of-tests-left":0,
  "class":"cpc",
  "cpc-node-descriptor":"00",
  "cpc-power-cap-allowed":null,
  "cpc-power-cap-current":null,
  "cpc-power-cap-maximum":null,
  "cpc-power-cap-minimum":null,
  "cpc-power-capping-state":null,
  "cpc-power-consumption":742,
  "cpc-power-rating":16628,
  "cpc-power-save-allowed":null,
  "cpc-power-saving":null,
  "cpc-power-saving-state":null,
  "cpc-serial-number":"00000005742D",
  "degraded-status":[
    "not-degraded"
  ],
  "description":"Central Processing Complex (CPC)",
  "does-wait-state-end-time-slice":null,
  "dpm-enabled":false,
  "ec-mcl-description":{
    "action":[
      {
        "activation":"current",
        "pending":false,
        "type":"channel-config"
      },
      {
        "activation":"current",
        "pending":false,
        "type":"coupling-facility-reactivation"
      }
    ]
  }
}

```

Figure 453. Get CPC Properties: Response (Part 1)

```

    {
      "activation": "current",
      "pending": false,
      "type": "power-on-reset-tracking"
    },
    {
      "activation": "next",
      "pending": false,
      "type": "channel-config"
    },
    {
      "activation": "next",
      "pending": false,
      "type": "coupling-facility-reactivation"
    },
    {
      "activation": "next",
      "pending": false,
      "type": "power-on-reset-tracking"
    }
  ],
  "ec": [
    {
      "description": "SE Framework",
      "mcl": [
        {
          "last-update": null,
          "level": "000",
          "type": "retrieved"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "activated"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "accepted"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "installable-concurrent"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "removable-concurrent"
        }
      ],
      "number": "P46598",
      "part-number": "02WF216",
      "type": "Base EC"
    },
  ],

```

Figure 454. Get CPC Properties: Response (Part 2)

```

{
  "description": "Concurrent Upgrade Sync Point",
  "mcl": [
    {
      "last-update": null,
      "level": "000",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46599",
  "part-number": "02WF217",
  "type": "Other Optional EC"
},
{
  "description": "SE Licensed Internal Code Alerts",
  "mcl": [
    {
      "last-update": null,
      "level": "000",
      "type": "retrieved"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "activated"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "accepted"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "installable-concurrent"
    },
    {
      "last-update": null,
      "level": "000",
      "type": "removable-concurrent"
    }
  ],
  "number": "P46600",
  "part-number": "02WF218",
  "type": "Other Optional EC"
},

```

Figure 455. Get CPC Properties: Response (Part 3)

```

    {
      "description": "Embedded Operating System",
      "mcl": [
        {
          "last-update": null,
          "level": "000",
          "type": "retrieved"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "activated"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "accepted"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "installable-concurrent"
        },
        {
          "last-update": null,
          "level": "000",
          "type": "removable-concurrent"
        }
      ],
      "number": "P46597",
      "part-number": "02WF215",
      "type": "Base EC"
    }
  ],
  "has-automatic-se-switch-enabled": false,
  "has-hardware-messages": true,
  "has-temporary-capacity-change-allowed": false,
  "has-unacceptable-status": false,
  "iml-mode": "lpar",
  "is-cbu-activated": false,
  "is-cbu-enabled": false,
  "is-cbu-installed": false,
  "is-cpacf-enabled": false,

```

Figure 456. Get CPC Properties: Response (Part 4)

```

"is-locked":false,
"is-on-off-cod-activated":false,
"is-on-off-cod-enabled":true,
"is-on-off-cod-installed":false,
"is-real-cbu-available":false,
"is-secure-execution-enabled":true,
"is-global-key-installed":true,
"is-host-key-installed":true,
"is-service-required":false,
"lan-interface1-address":"00106f2371e2",
"lan-interface1-type":"ethernet",
"lan-interface2-address":"00106f2371e3",
"lan-interface2-type":"ethernet",
"last-energy-advice-time":-1,
"last-used-activation-profile":"",
"last-used-iocds":"A1",
"machine-model":"T02",
"machine-serial-number":"00000005742D",
"machine-type":"8562",
"msu-permanent":693,
"msu-permanent-plus-billable":693,
"msu-permanent-plus-temporary":693,
"name":"SETR87",
"network1-ipv4-alt-ipaddr":"9.60.15.88",
"network1-ipv4-mask":"255.255.255.0",
"network1-ipv4-pri-ipaddr":"9.60.15.87",
"network1-ipv6-info":[
  {
    "alt-ip-address":"fe80::210:6fff:fe23:749e",
    "prefix":64,
    "pri-ip-address":"fe80::210:6fff:fe23:71e2",
    "type":"link-local"
  }
],
"network2-ipv4-alt-ipaddr":"9.60.14.88",
"network2-ipv4-mask":"255.255.255.0",
"network2-ipv4-pri-ipaddr":"9.60.14.87",
"network2-ipv6-info":[
  {
    "alt-ip-address":"fe80::210:6fff:fe23:749f%eth0",
    "prefix":64,
    "pri-ip-address":"fe80::210:6fff:fe23:71e3%eth0",
    "type":"link-local"
  }
],

```

Figure 457. Get CPC Properties: Response (Part 5)

```

"next-activation-profile-name":"DEFAULT",
"object-id":"d9c47445-64df-39f6-9d74-6376701508b5",
"object-uri":"/api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5",
"on-off-cod-activation-date":0,
"parent":null,
"processor-count-aap":0,
"processor-count-cbp":0,
"processor-count-defective":0,
"processor-count-general-purpose":0,
"processor-count-icf":0,
"processor-count-ifl":0,
"processor-count-iip":0,
"processor-count-pending":0,
"processor-count-pending-aap":0,
"processor-count-pending-cbp":0,
"processor-count-pending-general-purpose":0,
"processor-count-pending-icf":0,
"processor-count-pending-ifl":0,
"processor-count-pending-iip":0,
"processor-count-pending-service-assist":0,
"processor-count-service-assist":0,
"processor-count-spare":0,
"processor-running-time":null,
"processor-running-time-type":"system-determined",
"se-version":"2.15.0",
"software-model-permanent":"Z04",
"software-model-permanent-plus-billable":"Z04",
"software-model-permanent-plus-temporary":"Z04",
"status":"operating",
"storage-customer":360448,
"storage-customer-available":355328,
"storage-customer-central":0,
"storage-customer-expanded":0,
"storage-hardware-system-area":163840,
"storage-total-installed":524288,
"storage-vfm-increment-size":16,
"storage-vfm-total":0,
"zcpc-ambient-temperature":null,
"zcpc-dew-point":null,
"zcpc-exhaust-temperature":null,
"zcpc-heat-load":2534,
"zcpc-heat-load-forced-air":null,
"zcpc-heat-load-water":null,
"zcpc-humidity":null,
"zcpc-maximum-potential-heat-load":47491,
"zcpc-maximum-potential-power":13909,
"zcpc-power-cap-allowed":null,
"zcpc-power-cap-current":null,
"zcpc-power-cap-maximum":null,
"zcpc-power-cap-minimum":null,
"zcpc-power-capping-state":null,
"zcpc-power-consumption":742,
"zcpc-power-rating":16628,
"zcpc-power-save-allowed":null,
"zcpc-power-saving":null,
"zcpc-power-saving-state":null
}

```

Figure 458. Get CPC Properties: Response (Part 6)

Update CPC Properties

The Update CPC Properties operation updates one or more writable properties of the CPC object designated by *{cpc-id}*.

HTTP method and URI

POST /api/cpcs/{cpc-id}

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain one or more field names representing writable CPC properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the CPC object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **System Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	19	The request body contains a field whose corresponding data model property is not writable on this HMC and/or SE version.
	268	The requested update requires that the processor-running-time-type property already contain "user-determined" or that the request body also requests an update of the processor-running-time-type property to "user-determined" .
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Start CPC

The **Start CPC** operation starts the CPC object designated by *{cpc-id}*. The target CPC object must be enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/start
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the **Query Job Status** operation directed at the job URI provided in the response body.

The result document returned by the **Query Job Status** operation is specified in the description for the **Query Job Status** operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 773. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Start CPC is a process that makes a CPC operational, which means either:

- The CPC is ready to have a control program or operating system loaded, or
- The CPC has loaded and is running a control program or operating system.

Start CPC makes a CPC operational by:

- Using predefined information to set the operational capabilities and characteristics of the CPC
- Checking the current status of the CPC, and then performing only the operations necessary to make it operational.

So, using **Start CPC** is not limited to starting the system. Using **Start CPC** is recommended whenever you want to make the CPC or its partition objects operational.

A complete start starts the CPC and its partition objects completely in a single step. The result of a complete start is an operational CPC with partition objects loaded and running operating systems. The current status of the CPC and its partition objects determines which operations are performed during the start to make them operational. The start may include:

1. Turning CPC power on.
2. Performing a power-on reset, this includes allocating system resources to the CPC.
3. Then starting partition objects to support multiple images. Starting each partition object includes:

- a. Initializing it.
- b. Allocating system resources to it.
- c. Loading it with a control program or operating system.

Because the status of the CPC and its partition objects determines which operations must be performed during the start to make them operational, one or more operations listed above may not be performed during the start. For example:

- Starting the CPC does not perform a power-on reset if the CPC has already been power-on reset and the desired applicable settings, such as the operating mode and active input/output configuration data set (IOCDs), are already in effect.
- Starting the CPC does not perform any operations if the CPC is already operational and all desired settings are already in effect.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in [“Job status and reason codes” on page 773](#).

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel` Job operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by `{cpc-id}`
- Action/task permission for the **Start** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 771](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<code>{cpc-id}</code>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC object designated by <code>{cpc-id}</code> is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed, was canceled, or was rejected due to the current CPC status. The CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage note

This operation starts a CPC that is enabled for DPM only. If the targeted CPC object is not enabled for DPM, the Activate CPC operation may be used instead. Refer to [“Activate CPC”](#) on page 775 for details.

Stop CPC

The Stop CPC operation stops the CPC object designated by *{cpc-id}*. The target CPC object must be enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/stop
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve stop status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 775. The **job-results** field is null when this

operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Stop CPC is an orderly process for shutting down and turning off the CPC.

Shutting down and turning off the CPC, referred to also as stopping the CPC, includes:

- Ending hardware and software activity
- Clearing, releasing, and de-allocating hardware resources
- Turning off power.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes seen in [“Job status and reason codes” on page 775](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Stop** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 773](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC object designated by <i>{cpc-id}</i> is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status. The CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage note

This operation stops a CPC object that is enabled for DPM only. If the targeted CPC object is not enabled for DPM, the Deactivate CPC operation may be used instead. Refer to [“Deactivate CPC”](#) on page 778 for details.

Activate CPC

The Activate CPC operation activates the CPC object designated by *{cpc-id}*. This operation is not permitted when the CPC is enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/activate
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
activation-profile-name	String (1-16)	Optional	The name of the activation profile to be used for the request. If not provided, the request uses the profile name specified in the next-activation-profile-name property for the CPC object.
force	Boolean	Optional	Whether this operation is permitted when the CPC is in "operating" status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve activation status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 777. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Activation is a process that makes a CPC operational, which means either:

- The CPC is ready to have a control program or operating system loaded, or
- The CPC has loaded and is running a control program or operating system.

Activation makes a CPC operational by:

- Using predefined information, referred to as an activation profile, to set the operational capabilities and characteristics of the CPC
- Checking the current status of the CPC, and then performing only the operations necessary to make it operational as specified in the activation profile.

So, using activation is not limited to starting the system. Using activation is recommended whenever you want to make the CPC or its logical partitions operational.

A complete activation activates the CPC and its logical partitions completely in a single step. The result of a complete activation is an operational CPC with logical partitions loaded and running operating systems. The current status of the CPC and its logical partitions determines which operations are performed during activation to make them operational. Activation may include:

1. Turning CPC power on.
2. Performing a power-on reset, this includes allocating system resources to the CPC.
3. Then activating logical partitions to support multiple images. Activating each logical partition includes:
 - a. Initializing it.
 - b. Allocating system resources to it.
 - c. Loading it with a control program or operating system.

Because the status of the CPC and its logical partitions determines which operations must be performed during activation to make them operational, one or more operations listed above may not be performed during activation. For example:

- Activating the CPC does not perform a power-on reset if the CPC has already been power-on reset and the applicable settings in its assigned activation profile, such as the operating mode and active input/output configuration data set (IOCDs), are already in effect.

- Activating the CPC does not perform any operations if the CPC is already operational and all settings in its assigned activation profile are already in effect.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in [“Job status and reason codes” on page 777](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Activate** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 775](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be performed because the CPC object designated by <i>{cpc-id}</i> is currently enabled for DPM, which is not supported in this operation.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.

Job status code	Job reason code	Description
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status and use of the force=false parameter. If rejected due to force=false, the CPC status is unchanged. If the operation failed, the CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage note

This operation activates a CPC object that is not enabled for DPM. If the targeted CPC object is enabled for DPM, the `Start CPC` operation may be used instead. Refer to [“Start CPC”](#) on page 771 for details.

Deactivate CPC

The `Deactivate CPC` operation deactivates the CPC object designated by *{cpc-id}*. This operation is not permitted when the CPC is enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/deactivate
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
force	Boolean	Optional	Whether this operation is permitted when the CPC is in "operating" status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve activation status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 780. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Deactivation is an orderly process for shutting down and turning off the CPC.

Shutting down and turning off the CPC, referred to also as deactivating the CPC, includes:

- Ending hardware and software activity
- Clearing, releasing, and de-allocating hardware resources
- Turning off power.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes seen in [“Job status and reason codes” on page 780](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Deactivate** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 778](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be preformed because the CPC object designated by <i>{cpc-id}</i> is currently enabled for DPM, which is not supported in this operation.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status and use of the force=false parameter. If rejected due to force=false, the CPC status is unchanged. If the operation failed, the CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage note

This operation deactivates a CPC object that is not enabled for DPM. If the targeted CPC object is enabled for DPM, the `Stop CPC` operation may be used instead. Refer to [“Stop CPC”](#) on page 773 for details.

Import Profiles

The `Import Profiles` operation imports activation profiles and/or system activity profiles for the CPC from the SE hard drive into the CPC object designated by `{cpc-id}`. This operation is not permitted when the CPC is enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-profiles
```

In this request, the URI variable `{cpc-id}` is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
profile-area	Integer (1-4)	Required	The numbered hard drive area from which the profiles are imported. Use the profile-area value specified on the prior <code>Export Profiles</code> operation.

Description

The Support Element provides four reusable areas on its hard drive from which the data save by a prior `Export Profiles` can be read.

Exporting and importing profiles is necessary only when you intend to have your current system and Support Element replaced with a new system and Support Element. Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **Export/Import Profile Data (API only)** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	This operation cannot be preformed because the CPC designated by the request URI is enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	279	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Export Profiles

The Export Profiles operation exports activation profiles and/or system activity profiles from the CPC object designated by *{cpc-id}* to the SE hard drive. This operation is not permitted when the CPC is enabled for DPM.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/export-profiles

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
profile-area	Integer (1-4)	Required	The numbered hard drive area to which the profiles are exported. Any existing data is overwritten.

Description

The Support Element provides four reusable areas on its hard drive that can be used as temporary save areas. The choice of save area is up to the caller.

Exporting and importing profiles is necessary only when you intend to have your current system and Support Element replaced with a new system and Support Element. Support Element (Version 2.12.1 and newer) information can be found on the console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **Export/Import Profile Data (API only)** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be performed because the CPC designated by the request URI is enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	279	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Set Auto-Start List

The Set Auto-Start List operation identifies a CPC's partitions that are automatically started when the CPC is started. It also specifies the order in which they are started. The target CPC object is designated by *{cpc-id}* and must be enabled for DPM.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/set-auto-start-list

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
auto-start-list	Array of objects	Required	An array of auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. The format of that object is described in the “Class specific additional properties” on page 742.

Description

This operation defines the CPC's partition auto-start list. The auto-start list is an ordered list of partitions and/or groups of partitions that are automatically started when the CPC is started. See the [“Class specific additional properties”](#) on page 742 for the format of the entries in this list. Each partition or partition group is started before those that are later than it in the list. The order in which partitions within a partition group are started is not specified. An empty auto-start list indicates that no partitions are to be automatically started.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing CPC, and the API user must have object-access permission to it, otherwise, status code 404 (Not Found) is returned. Each Partition URI in the request body must designate an existing partition, and the API user must have object-access permission to it, otherwise, status code 404 (Not found) is returned. If the auto-start list contains duplicate partition group names or a partition is listed multiple times in the list or an empty partition group is specified, HTTP status code 400 (Bad Request) is returned.

If the CPC object designated by *{cpc-id}* is not enabled for DPM, HTTP status code 409 (Conflict) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **System Details** task.
- Object-access permission to all partitions specified in the request body.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 366. Set Auto-Start List: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	8	Two or more of the partition group names specified in the auto-start-list in the request body are duplicates.
	333	One or more of the partition URIs specified in the auto-start-list in the request body does not designate a partition object of the CPC object designated by <i>{cpc-id}</i> .
	334	Two or more of the partition URIs specified in the auto-start-list in the request body are duplicates.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	One or more of the partition URIs specified in the auto-start-list in the request body does not designate an existing partition object, or the API user does not have object access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC designated by <i>{cpc-id}</i> is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/e5ae3ab6-ac8d-33bc-9739-eb142d89804d/operations/set-auto-start-  
list HTTP/1.1  
x-api-session: 2kk848szmu8mo00lkj4c19254fiejeanyv316j0d5d4uppgp8t  
content-type: application/json  
content-length: 384  
{  
  "auto-start-list": [  
    {  
      "partition-uri": "/api/partitions/0589baec-d599-11e5-8959-42f2e9105e9b",  
      "post-start-delay": 15,  
      "type": "partition"  
    },  
    {  
      "description": "description for group1",  
      "name": "group1",  
      "partition-uris": [  
        "/api/partitions/39daff36-b51e-11e5-9710-42f2e9105e9b",  
        "/api/partitions/8c6e48a2-ce91-11e5-98b3-42f2e9105e9b"  
      ],  
      "post-start-delay": 0,  
      "type": "partition-group"  
    }  
  ]  
}
```

Figure 459. Set Auto-Start List: Request

```
204 No Content  
server: zSeries management console API web server / 2.0  
cache-control: no-cache  
date: Wed, 17 Feb 2016 20:54:12 GMT  
  
<No response body>
```

Figure 460. Set Auto-Start List: Response

Add Temporary Capacity

The Add Temporary Capacity operation adds temporary processors or increases temporary model capacity to the CPC object designated by *{cpc-id}*.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/add-temp-capacity
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
record-id	String (1-8)	Required	Identifies the capacity record to be used for this request
software-model	String (1-3)	Optional	The target software model. This value must be one of the software models defined within the capacity record. Implicit in this value is the number of general processors added. If not provided, the current software model is not changed.

Field name	Type	Rqd/Opt	Description
processor-info	Array of processor-info objects	Optional	A nested object that defines the number of specialty processors to be added. If not provided, the number of specialty processors is not changed.
force	Boolean	Optional	Whether the operation proceeds if not enough processors are available (true) or not (false). The default is false.
test	Boolean	Required	Whether the request should activate real or test resources for the capacity record. Set true if test or set to false if real. This is mainly used for Capacity Backup Upgrade (CBU) activations. See the <i>Capacity on Demand User's Guide</i> , SC28-6943. For most records, field should be set to false.

processor-info object

Field name	Type	Rqd/Opt	Description
processor-type	String Enum	Required	Identifies the type of specialty processors to be affected. One of: • "aap" - Application Assist Processor • "ifl" - Integrated Facility for Linux processor • "icf" - Internal Coupling Facility processor • "iip" - z Integrated Information Processors • "sap" - System Assist Processor • "cbp" - Container Based Processor¹
num-processor-steps	Integer	Optional	The delta to the current number of processors. If not provided, the number of processors is not changed.

¹This property can be included in the request body only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version.

Description

Removal of these temporary resources can be performed manually via the Remove Temporary Capacity operation or automatically upon expiration of the capacity record.

Refer to the *Capacity on Demand User's Guide* for details on temporary capacity changes.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by {cpc-id}
- Action/task permission to the **Perform Model Conversion** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	271	A duplicate processor-type entry was found in the processor-info array, remove the duplicate entry.
	275	The test value does not match the value stored in the capacity record.
	276	Either the request specifies more resources than available or the requested software model specifies fewer resources than the current software model.
	277	A temporary capacity record is already active. It must be deactivated before a new capacity record can be activated.
	278	The software-model value was not found in the capacity record. Only software models as defined in the target capacity record can be specified.
	298	The operation parameters conflict with the capacity record type: • force=true is permitted only for CBU, CPE and loaner capacity records.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The requested capacity record does not exist.
409 (Conflict)	1	The operation is unavailable in the current CPC state: • The SE is not configured to allow temporary capacity changes via an API • The CPC status property is not "operating" or the CPC iml-mode property is not "lpar" • No physical processors are operating • The CPC is IMLed in a test or debug mode • An IML is required
	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	297	Some, but not all, of the requested resources were added.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Remove Temporary Capacity

The Remove Temporary Capacity operation removes temporary processors or decreases temporary model capacity from the CPC object designated by *{cpc-id}*.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/remove-temp-capacity
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
record-id	String (1-8)	Required	Identifies the capacity record to be used for this request
software-model	String (1-3)	Optional	The target software model. This value must be one of the software models defined within the capacity record. Implicit in this value is the number of general processors removed. If not provided, the current software model is not changed.
processor-info	Array of processor-info objects	Optional	A nested object that defines the number of specialty processors to be removed. If not provided, the number of specialty processors is not changed. Refer to “Request body contents” on page 785 of the <i>Add Temporary Capacity</i> operation for details.

Description

When you are finished using all or part of a capacity upgrade, you can remove processors or decrease model capacity using this operation. You can only remove activated resources for the specific offering. You cannot remove dedicated engines or the last processor of a processor type.

If you remove resources back to the base configuration, the capacity record activation is completed. That is, if you remove the last temporary processor, your capacity record is deactivated. For a CBU and On/Off CoD record, to add resources again, you must use another Add Temporary Capacity operation. For an On/Off CoD test or CPE record, once the record is deactivated, it is no longer available for use. You can then delete the record.

After removal of the resources, the capacity record remains as an installed record. If you want a record deleted, you must manually select the record on the Installed Records page and click Delete.

Refer to the *Capacity on Demand User's Guide* for details on temporary capacity changes.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **Perform Model Conversion** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	271	A duplicate processor-type entry was found in the processor-info array, remove the duplicate entry.
	276	Either the request specifies more resources than are currently active or the requested software model specifies more resources than the current software model.
	278	The software-model value was not found in the capacity record. Only software models as defined in the target capacity record can be specified.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The requested capacity record does not exist.
409 (Conflict)	1	The operation is unavailable in the current CPC state: • The SE is not configured to allow temporary capacity changes via an API • The CPC status property is not "operating" or the CPC iml-mode property is not "lpar".
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Swap Current Time Server

The Swap Current Time Server operation changes the role of the CPC object designated by *{cpc-id}* to the Current Time Server (CTS).

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/swap-cts
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
stp-id	String (1-8)	Required	Identifies the STP. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).

Description

This operation changes the role of the CPC object designated by *{cpc-id}* to the Current Time Server (CTS).

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the Manage System Time and **Modify Assigned Server Roles** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	285	The CPC targeted by this operation is already the Preferred Time Server.
	286	The operation was rejected for one of the following reasons: • The stp-id field value does not match the current CTN identifier • The operation is not permitted for a mixed CTN.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress • no alternate is active • this CPC is not the backup time server
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during the operation.

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Set STP Configuration

The Set STP Configuration operation updates the configuration for an STP-only Coordinated Timing Network.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/set-stp-config

In this request, the URI variable {cpc-id} is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
stp-id	String (1-8)	Required	The current STP identifier for the CTN, used to verify that the CPC is a member of the correct CTN. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).
new-stp-id	String (1-8)	Optional	If provided, the new STP identifier for the CTN, Can contain 0-9, a-z, A-Z, underline (_) and dash (-).
force	Boolean	Required	Whether a disruptive operation is allowed (true) or rejected (false)
preferred-time-server	stp-node object	Required	Identifies the CPC object to be the Preferred Time Server. Refer to Table 297 on page 573 for details.
backup-time-server	stp-node object	Optional	Identifies the CPC object to be the Backup Time Server. If not provided, the STP has no Backup Time Server. Refer to Table 297 on page 573 for details.
arbiter	stp-node object	Optional	Identifies the CPC object to be the Arbiter for the CTN. If not provided, the STP has no Arbiter. Refer to Table 297 on page 573 for details.
current-time-server	String Enum	Required	Identifies the role of the Current Time Server (CTS). One of: • "preferred" - the Preferred Time Server is the CTS • "backup" - the Backup Time Server is the CTS.

Description

The CPC object designated by {cpc-id} must be the system that becomes the Current Time Server (CTS). On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the Manage System Time and **Modify Assigned Server Roles** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	282	The operation was rejected, due to an incomplete preferred, backup or arbiter nested object specification. Refer to Table 297 on page 573 for details.
	284	This operation does not target the Current Time Server CPC.
	285	The operation was rejected, due to one of the following configuration errors: • A backup-time-server object is required when providing an arbiter object • A backup-time-server object is required when current-time-server is backup • The preferred-time-server, backup-time-server and arbiter objects do not reference different CPCs
	286	The operation was rejected for one of the following reasons: • The stp-id field value does not match the current CTN identifier • The operation is not permitted for a mixed CTN
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	An object URI in one of the stp-node objects in the request body does not designate an existing CPC object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress • no alternate is active • the operation is not permitted for a mixed-CTN.
	287	The provided configuration can only be set by specifying force=true.
	288	No communication path between preferred-time-server and backup-time-server.
	289	No communication path to the arbiter.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Change STP-only Coordinated Timing Network

The Change STP-only Coordinated Timing Network operation, sent to the CPC object designated by *{cpc-id}* with the role of Current Time Server (CTS) in an STP-only Coordinated Timing Network (CTN), changes the STP ID portion of the CTN ID for the entire STP-only CTN.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/change-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
stp-id	String (1-8)	Required	The new STP identifier. Can contain 0-9, a-z, A-Z, underline () and dash (-).

Description

This operation, sent to the CPC object designated by *{cpc-id}* with the role of Current Time Server (CTS) in an STP-only Coordinated Timing Network (CTN), changes the STP ID portion of the CTN ID for the entire STP-only CTN.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the Manage System Time and **Rename CTN** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	284	The CPC targeted by the operation is not the Current Time Server. Retry the operation using the object-uri for the Current Time Server CPC.
	286	The operation is not permitted for a mixed CTN.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Join STP-only Coordinated Timing Network

The Join STP-only Coordinated Timing Network operation allows a CPC object designated by *{cpc-id}* to join an STP-only Coordinated Timing Network (CTN).

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/join-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
stp-id	String (1-8)	Required	Identifies the STP to be joined. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).

Description

If the CPC object is already participating in a different STP-only CTN and is the Current Time Server (CTS), the operation is rejected. Otherwise, the CPC object is removed from its current CTN and joins the specified CTN.

If the CPC object has an ETR ID, the ETR ID is removed.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the Manage System Time and **Add Systems to CTN** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Leave STP-only Coordinated Timing Network

The Leave STP-only Coordinated Timing Network operation allows a CPC object designated by *{cpc-id}* to leave the STP-only Coordinated Timing Network (CTN) in which it currently participates.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/leave-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Description

The CPC object cannot be the Current Time Server (CTS) in the CTN in which it is currently participating. On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the Manage System Time and **Remove Systems from CTN** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	286	The operation is not permitted for a mixed CTN.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • this CPC is not a member of a CTN • this CPC is the Current Time Server.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Get CPC Audit Log

The Get CPC Audit Log operation returns the CPC's audit log, filtered according to the query parameters, if specified.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-audit-log
```

In this request, the URI variable `{cpc-id}` is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
end-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON array of JSON objects returned using HTTP chunked transfer encoding. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See [Table 320 on page 593](#) for more information.

Description

This operation returns the CPC's audit log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the CPC itself. If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and that CPC must be at a release level that supports this operation. If any of these conditions is not met, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Audit and Log Management** task; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the audit log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented on console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object specified in the request URI

- Action/task permission to the **Audit and Log Management** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 797](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 367. Get CPC Audit Log: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	307	The request timed out while attempting to communicate with the SE or while attempting to get the log data.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/operations/get-audit-log
HTTP/1.1
x-api-session: 1ui4gmb59aunfkk8of69nrpks5mtmq5xjc613rdxjq53iv0e1j
```

Figure 461. Get CPC Audit Log: Request

```

200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Fri, 03 Oct 2014 00:10:35 GMT
content-type: application/json; charset=ISO-8859-1
[
  {
    "event-data-items": [],
    "event-details": [
      {
        "event-details-data-items": [],
        "event-details-message": "S32 - Disabled"
      }
    ],
    "event-id": "1809",
    "event-message": "Power Cap settings have changed.",
    "event-name": "POWERCAP",
    "event-time": "1412202863030",
    "user-uri": null,
    "userid": null
  },
  {
    "event-data-items": [
      {
        "data-item-number": 0,
        "data-item-type": "string",
        "data-item-value": "HMCCHGM(1.2.3.5)"
      },
      {
        "data-item-number": 1,
        "data-item-type": "string",
        "data-item-value": "S32"
      }
    ],
    "event-details": [],
    "event-id": "734",
    "event-message": "Remote support call generated on S32 is being handled by call-home server HMCCHGM(1.2.3.5).",
    "event-name": "TRSF_OFFER",
    "event-time": "1412262459470",
    "user-uri": null,
    "userid": null
  }
]

```

Figure 462. Get CPC Audit Log: Response

Get CPC Security Log

The Get CPC Security Log operation returns the CPC's security log, filtered according to the query parameters, if specified.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-security-log
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
end-time	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON array of JSON objects returned using HTTP chunked transfer encoding. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See [Table 320 on page 593](#) for more information.

Description

This operation returns the CPC's security log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the CPC itself. If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and that CPC must be at a release level that supports this operation. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **View Security Logs** task; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented on console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object specified in the request URI
- Action/task permission to the **View Security Logs** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in ["Response body contents" on page 800](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 368. Get CPC Security Log: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Table 368. Get CPC Security Log: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	307	The request timed out while attempting to communicate with the SE or while attempting to get the log data.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/operations/get-security-log
HTTP/1.1
x-api-session: 1ui4gmb59aunfkk8of69nrpks5mtmq5xjc613rdxjq53iv0e1j
```

Figure 463. Get CPC Security Log: Request

```

200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Fri, 03 Oct 2014 00:10:35 GMT
content-type: application/json;charset=ISO-8859-1
[
  {
    "event-data-items":[],
    "event-details":[],
    "event-id":"778",
    "event-message":"Mirroring data from the primary Support Element to the
      alternate Support Element started.",
    "event-name":"ASEMIRST",
    "event-time":1412258403480,
    "user-uri":null,
    "userid":null
  },
  {
    "event-data-items":[
      {
        "data-item-number":0,
        "data-item-type":"string",
        "data-item-value":"Communications to the Alternate SE was not
          active."
      }
    ],
    "event-details":[],
    "event-id":"779",
    "event-message":"Mirroring data from the primary Support Element to the
      alternate Support Element failed. Communications to the Alternate SE was not
      active.",
    "event-name":"ASEMIRRN0",
    "event-time":1412258414110,
    "user-uri":null,
    "userid":null
  }
]

```

Figure 464. Get CPC Security Log: Response

List CPC Hardware Messages

The List CPC Hardware Messages operation lists the current set of hardware messages associated with the CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages
```

In this request, the URI variable *{cpc-id}* is the object ID of a CPC object for which hardware messages are to be listed.

Query Parameters

Name	Type	Rqd/Opt	Description
begin-time	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created earlier than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
end-time	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created later than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
hardware-messages	Array of hardware-message-info objects	Array of nested hardware-message-info objects as defined in the next table.

Each nested hardware-message-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	The canonical URI path of the hardware message. The URI is in the following form: <code>/api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}</code>
timestamp	Timestamp	The date and time the hardware message was created
text	String	The text of the hardware message.

Description

This operation returns a set of CPC hardware messages in increasing timestamp order, filtered according to the query parameters, if specified. Each hardware message describes an event or notification that may require the operator's attention. The list of hardware messages can be limited by specifying explicit filtering criteria on the request.

If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

If there are no hardware messages associated with the CPC, or if no hardware messages are to be included in the results due to filtering, an empty array is returned and the operation completes successfully.

The URI path must designate an existing CPC and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the Hardware Messages task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by `{cpc-id}`.
- Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 803.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 369. List CPC Hardware Messages: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The begin-time value is greater than the end-time value.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task or the Hardware Messages task in view-only mode.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 43.

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/hardware-messages HTTP/1.1
x-api-session: 35rr3x40qbnou8zwmx8ad80ldp8koes4f2abc16m1fg5jm4tug
```

Figure 465. List CPC Hardware Messages: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Oct 2014 17:08:58 GMT
content-type: application/json; charset=UTF-8
content-length: 242
{
  "hardware-messages": [
    {
      "element-uri": "/api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/hardware-messages/
43a2922a-4d6b-11e4-972f-42f2e9ccd169",
      "text": "Licensed internal code has detected a problem. [Problem # 49]",
      "timestamp": 1412608381950
    }
  ]
}
```

Figure 466. List CPC Hardware Messages: Response

Get CPC Hardware Message Properties

The Get CPC Hardware Message Properties operation retrieves the properties of a single CPC hardware message.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
```

URI Variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the CPC object.
<i>{hardware-message-id}</i>	Element ID of the hardware message to retrieve.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the CPC hardware message object as defined in [“Data model” on page 741](#). Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation retrieves the properties of a single CPC hardware message specified by *{hardware-message-id}*.

The URI path must designate an existing CPC, and the API user must have object access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the Hardware Messages task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 805](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 370. Get CPC Hardware Message Properties: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Table 370. Get CPC Hardware Message Properties: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task or the Hardware Messages task in view-only mode.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
    9faa758e-58b8-11ea-9fb3-00106f23f56e HTTP/1.1
x-api-session: 4o94wl43xtdv4bi0c0hu1aict0xakizcvnd5umlykpn8cmkrc1
```

Figure 467. Get CPC Hardware Message Properties: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:58:32 GMT
content-type: application/json; charset=UTF-8
content-length: 849
{
  "class": "hardware-message",
  "details": {
    "corrective-action": [
      "Problem Analysis is now complete.",
      "Service is required."
    ],
    "created": 1582735965261,
    "description": [
      "A power failure has been detected.",
      "The system is still operating."
    ],
    "lir-node-data": null,
    "problem-data": [
      {
        "caption": "System name",
        "value": "T204"
      },
      {
        "caption": "Date",
        "value": "Feb 26, 2020"
      },
      {
        "caption": "Time",
        "value": "11:52:45 AM"
      }
    ],
    "repair-impact": [
      "The repair of this problem can most likely be performed concurrent with system operations."
    ],
    "type": "common-problem"
  },
  "element-id": "9faa758e-58b8-11ea-9fb3-00106f23f56e",
  "element-uri": "/api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/9faa758e-58b8-11ea-9fb3-00106f23f56e",
  "parent": "/api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a",
  "service-supported": true,
  "text": "Power problem. [Problem # 165]",
  "timestamp": 1582736056410
}

```

Figure 468. Get CPC Hardware Message Properties: Response

Delete CPC Hardware Message

The Delete CPC Hardware Message operation deletes a single CPC hardware message

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
```

URI Variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the CPC object.
<i>{hardware-message-id}</i>	Element ID of the hardware message to delete.

Description

This operation deletes a specific CPC hardware message. The hardware message to be deleted is identified by the *{hardware-message-id}* variable in the URI.

The URI path must designate an existing CPC and the API user must have object-access permission to it; otherwise status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 371. Delete CPC Hardware Message: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have Object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
DELETE /api/cpcs/0fdde999-5957-3129-99aa-b6f4bfbbc071/hardware-messages/  
d5591a80-43f8-11e4-ac52-42f2e910664b HTTP/1.1  
x-api-session: c8un3odpy8yyp150o3poz1ud4gwyfodlwyq495327bpyn2p0z
```

Figure 469. Delete CPC Hardware Message: Request

```
204 No Content
date: Mon, 09 Feb 2015 20:07:31 GMT
server: zSeries management console API web server / 2.0

<No response body>
```

Figure 470. Delete CPC Hardware Message: Response

Request CPC Service

The Request CPC Service operation electronically transmits problem information to request service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/request-service
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
customer-name	String	Optional	The name of the person that can be contacted about the problem.
customer-phone	String	Optional	The telephone number of the person that can be contacted about the problem.

Description

This operation electronically requests service for the problem reported by this hardware message. Customer contact information may optionally be provided in the request body. Upon successful completion, the service request is prepared and queued for transmission, and this hardware message is deleted.

The URI path must designate an existing CPC, and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. Remotely requesting service must also be enabled and configured on the HMC; otherwise, status code 409 (Conflict) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by {cpc-id}.
- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <i>{hardware-message-id}</i> designates a CPC hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	341	The HMC is not enabled and configured for remotely requesting service.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
9faa758e-58b8-11ea-9fb3-00106f23f56e/operations/request-service HTTP/1.1
x-api-session: 440mepm56w27o7965y61kbkm4g422hja9841rm74mc2wdpf3vw
content-type: application/json
content-length: 56
{
  "customer-name": "Jenny",
  "customer-phone": "867-5309"
}
```

Figure 471. Request CPC Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:00:06 GMT

<No response body>
```

Figure 472. Request CPC Service: Response

Get CPC Service Request Information

The Get CPC Service Request Information operation returns problem information and a telephone number to be used for requesting service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/get-service-information
```

URI Variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the CPC object.
<i>{hardware-message-id}</i>	Element ID of the hardware message.

Response body contents

On successful completion, the response body contains a JSON object with the following fields.

Name	Type	Description
service-phone	String	The telephone number to call for service.
machine-model	String (1-3)	The model of the machine where the problem occurred.
machine-type	String (1-4)	The type of the machine where the problem occurred.
machine-serial-number	String (1-12)	The serial number of the machine where the problem occurred.
problem-type	String	The type of the problem to be reported to service.
problem-number	Integer	The identifying number of the problem to be reported to service.
problem-data	String	Additional problem data to be reported to service.
reference-code	String	The problem reference code to be reported to service.

Description

This operation is used to manually request service for the problem reported by this hardware message. This may be used if remote service is not configured or not functioning to call a service representative directly and provide the problem details. Upon successful completion, problem details to be reported to service are returned, and this hardware message is deleted.

The URI path must designate an existing CPC, and the APIU user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 811.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <i>{hardware-message-id}</i> designates a CPC hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
    ea2ec24e-5963-11ea-9fb3-00106f23f56e/operations/get-service-information HTTP/1.1
x-api-session: 34w3jublxvx3d51s5x0xc69kyuwshfvxy6r94u8chwkbzbu06qd
```

Figure 473. Get CPC Service Request Information: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:01:45 GMT
content-type: application/json; charset=UTF-8
content-length: 231
{
  "machine-model": "T02",
  "machine-serial-number": "0000200273D8",
  "machine-type": "8562",
  "problem-data": "00L5201,1,01KU212,1",
  "problem-number": 167,
  "problem-type": "1",
  "reference-code": "30C02011-6863670B",
  "service-phone": "1-800-IBM-SERV"
}

```

Figure 474. Get CPC Service Request Information: Response

Decline CPC Service

The Decline CPC Service operation declines service for the error and deletes the hardware message designated by the URI path.

HTTP method and URI

POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/decline-service

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message.

Description

This operation is used to decline service for the problem reported by this hardware message. Upon successful completion, the hardware message is deleted.

The URI path must designate an existing CPC, and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by {cpc-id}.
- Action/Task permission to the **Hardware Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	340	The element ID in the URI <i>{hardware-message-id}</i> designates a CPC hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the Hardware Messages task.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI <i>{hardware-message-id}</i> does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
ed6a4052-58ad-11ea-9fb3-00106f23f56e/operations/decline-service HTTP/1.1
x-api-session: 69u9tm62kmcim81nxvi556utt8bxq71kh80jw3ayhq12p6i3b6
content-type: application/json
```

Figure 475. Decline CPC Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:02:52 GMT

<No response body>
```

Figure 476. Decline CPC Service: Response

Export WWPN List

The `Export WWPN List` operation exports the world wide port names (WWPNs) of the HBAs of the specified partitions.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/export-port-names-list
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC that contains all the specified partitions.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
partitions	Array of String/ URI	Required	Array of canonical URI paths, one for each partition whose list of WWPNS is to be exported.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
wwpn-list	String	The comma separated list contains the WWPNS along with each associated partition name , adapter-id , and device-number .

Description

The `Export WWPNS List` operation returns the list of host port WWPNS of the partitions specified by the **partitions** field of the request body. These partitions must be part of the CPC specified by `{cpc-id}`. The list is provided in a JSON object as a single string in Comma-Separated Values (CSV) format. Each line in the string will be represented as: Partition name,Adapter ID,device-number,WWPN.

On successful execution, the WWPNS list for all the HBAs defined in the specified partitions are provided in the response body, and HTTP status code 200 (OK) is returned.

The request URI path must designate an existing CPC object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. If the array of partition URIs is empty, status code 400 (Bad Request) is returned. The URIs in the request body must designate existing Partition objects and API user must have object-access permission to them; otherwise, status code 404 (Not Found) is returned. The partitions must be part of the specified CPC; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have action access permission to the **Export WWPNS** task; otherwise, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in [“Request body contents” on page 815](#). If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the CPC has the **"dpm-storage-management"** feature enabled, 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object specified in the request URI.
- Object-access permission to the partition objects specified in the request body.
- Action/task permission to the **Export WWPNS** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 815](#)

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 372. Export WWPN List: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation.
	149	The partitions array is empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	The object ID <i>{partition-id}</i> does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	12	The operation is not supported when the "dpm-storage-management" feature is enabled on the CPC.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/51342eec-1ba0-3866-b639-a99860807b89/operations/export-port-names-
list HTTP/1.1
x-api-session: 3ud8oxm0p8d8um2r3y2pjhbws8ishkf4r4vqqocrsrbl1e0h0y2
content-type: application/json
content-length: 72
{
  "partitions":[
    "/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192"
  ]
}
```

Figure 477. Export WWPN List: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:59:04 GMT
content-type: application/json;charset=UTF-8
content-length: 312
{
  "wwpn-list": "#Version: 1\r\n#Partition Name,AdapterPortId,DeviceNumber,WWPN\r\n
  \nMyPartition_1,159,1007,0000000000000009\r\nMyPartition_1,158,1002,
  0000000000000006\r\nMyPartition_1,159,1001,0000000000000005\r\nMyPartition_
  1,158,1003,0000000000000007\r\nMyPartition_1,159,1005,000000000000000A\r\n"
}
```

Figure 478. Export WWPN List: Response

Import DPM Configuration

The Import DPM Configuration operation imports DPM objects such as partitions, NICs, HBAs, virtual functions, and their properties. It also restores DPM-specific CPC properties, like the description, auto-start partition list and capacity groups.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-dpm-config
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
cpc-properties	cpc-info object	Optional	An object containing the names of CPC properties and the values to which each of those properties is to be set. This is a very limited set of CPC properties relevant to CPCs enabled for DPM mode.
se-version	String (1-8)	Required	The internal code release level of the primary SE on the machine from which the configuration was exported.
available-features-list	Array of cpc-feature-info objects	Required	The list of optional features or behaviors supported by the CPC on the machine from which the configuration was exported. If the CPC has no optional features, then the array must be empty.
preserve-uris	Boolean	Optional	Controls whether object IDs / element IDs are preserved or if new object IDs / element IDs will be generated. Because those IDs are components of URIs, this field controls whether the URIs of imported objects are preserved. This applies to all objects that will be recreated (Partitions, HBAs, NICs, Virtual Functions, Capacity Groups). Default: false
preserve-wwpns	Boolean	Optional	Controls whether HBA WWPNS are preserved or if new WWPNS will be generated. Default: false
adapter-mapping	Array of adapter-mapping-info objects	Optional	Array of adapter mapping information objects. Required when the I/O adapter configuration on this machine differs from that of the machine from which the configuration was exported.
partitions	Array of objects	Required	Array of objects containing the properties of the Partitions to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Partition Properties operation issued when the configuration information was exported.

Field name	Type	Rqd/Opt	Description
nics	Array of objects	Optional	Array of objects containing the properties of the NICs to be imported. Each element of this array is expected to be the equivalent of a response body from a Get NIC Properties operation issued when the configuration information was exported.
hbas	Array of objects	Optional	Array of objects containing the properties of the HBAs to be imported. Each element of this array is expected to be the equivalent of a response body from a Get HBA Properties operation issued when the configuration information was exported.
virtual-functions	Array of objects	Optional	Array of objects containing the properties of the Virtual Functions to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Virtual Function Properties operation issued when the configuration information was exported.
adapters	Array of objects	Required	Array of objects containing the properties of the Adapters in the exported configuration. Each element of this array is expected to be the equivalent of a response body from a Get Adapter Properties operation issued when the configuration information was exported. This will not recreate any Adapters; it is used for internal purposes only.
virtual-switches	Array of objects	Optional	Array of objects containing the properties of the Virtual Switches in the exported configuration. Each element of this array is expected to be the equivalent of a response body from a Get Virtual Switch Properties operation issued when the configuration information was exported. This will not recreate any Virtual Switches; it is used for internal purposes only.
capacity-groups	Array of objects	Optional	Array of objects containing the properties of the Capacity Groups to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Capacity Group Properties operation issued when the configuration information was exported.
storage-sites	Array of objects	Optional	Array of objects containing the properties of the Storage Sites to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Site Properties operation issued when the configuration information was exported.
storage-subsystems	Array of objects	Optional	Array of objects containing the properties of the Storage Subsystems to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Subsystem Properties operation issued when the configuration information was exported.
storage-fabrics	Array of objects	Optional	Array of objects containing the properties of the Storage Fabrics to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Fabric Properties operation issued when the configuration information was exported.

Field name	Type	Rqd/Opt	Description
storage-switches	Array of objects	Optional	Array of objects containing the properties of the Storage Switches to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Switch Properties operation issued when the configuration information was exported.
storage-control-units	Array of objects	Optional	Array of objects containing the properties of the Storage Control Units to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Control Unit Properties operation issued when the configuration information was exported.
storage-paths	Array of objects	Optional	Array of objects containing the properties of the Storage Paths to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Path Properties operation issued when the configuration information was exported.
storage-groups	Array of objects	Optional	Array of objects containing the properties of the Storage Groups to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Group Properties operation issued when the configuration information was exported.
storage-volumes	Array of objects	Optional	Array of objects containing the properties of the Storage Volumes to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Volume Properties operation issued when the configuration information was exported.
storage-templates	Array of objects	Optional	Array of objects containing the properties of the Storage Templates to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Template Properties operation issued when the configuration information was exported.
storage-template-volumes	Array of objects	Optional	Array of objects containing the properties of the Storage Template Volumes to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Template Volume Properties operation issued when the configuration information was exported.
virtual-storage-resources	Array of objects	Optional	Array of objects containing the properties of the Virtual Storage Resources to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Virtual Storage Resources Properties operation issued when the configuration information was exported.
storage-ports	Array of objects	Optional	Array of objects containing the properties of the Storage Ports to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Storage Port Properties operation issued when the configuration information was exported.

Field name	Type	Rqd/Opt	Description
network-ports	Array of objects	Optional	Array of objects containing the properties of the Network Ports to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Network Port Properties operation issued when the configuration information was exported.

The cpc-info nested object contains the following fields:

Table 373. cpc-info nested object		
Name	Type	Description
description	String (1-1024)	The descriptive text associated with this CPC object
auto-start-list	Array of auto-start-entry objects	An array of auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. The format of that object is described in the “Class specific additional properties” on page 742.

The adapter-mapping-info nested object contains the following fields:

Table 374. adapter-mapping-info nested object		
Name	Type	Description
new-adapter-id	String (3)	The hexadecimal value of the adapter ID (PCHID) on this machine
old-adapter-id	String (3)	The hexadecimal value of the adapter ID (PCHID) on the old machine

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Name	Type	Description
output	Array of output-details objects	An array of output-details objects containing detailed information about the parts of the configuration that were not restored.

Each nested output-details info object contains the following fields:

Name	Type	Description
category	String/ Enum	One of the following, describing the category of the message: • "system-to-switches-not-restored" - Certain storage switches connections could not be restored. The items field contains information describing those connections. • "acc-vf-not-restored" - Accelerator virtual functions for certain partitions were not restored. The items field contains the names of those partitions. • "partition-id-reset" - The Partition ID (partition-id property) for certain partitions was changed. The items field contains the names of those partitions. • "boot-type-reset" - The boot type (boot-device property) for certain partitions was reset from "iso-image" to "none". The items field contains the names of those partitions. • "ssc-password-reset" - The password (ssc-master-pw property) for certain SSC partitions was reset. The items field contains the names of those partitions. • "boot-password-reset" - The FTP password (boot-ftp-password property) for certain partitions was reset. The items field contains the names of those partitions. • "processor-type-not-available" - Certain partitions were not restored because the required processor type is not available. The items field contains the names of those partitions. • "virtual-switch-port-changed" - Certain virtual switch ports have been changed to the default port. The items field contains information identifying those ports.
items	Array of String	Additional details about which resources were modified or not restored. The value of the category field defines the contents of this field.

Description

This operation restores a full DPM configuration with all its artifacts like partitions, HBAs, NICs, Accelerators, and Crypto devices. Unique identifiers like Object IDs and WWPNs are preserved. This task is mainly for migrating a DPM configuration from a z13 or later machine to a new machine. The operation will attempt to restore all provided objects, and stop on the first error. The request body may contain the same set of properties as generated by the `Get Inventory` operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by `{cpc-id}`
- Action/task permission to the **Import Dynamic Partition Manager Configuration** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided. If not all parts of the configuration could be restored, a 200 (OK) is returned and the response body provides additional details.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	A hipersocket adapter failed to recreate. An adapter with the same name, but different adapter-id already exists. The provided adapter mapping contained invalid adapter-id values.
	100	The provided se-version is not supported for the import operation of this CPC
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI <i>{cpc-id}</i> does not designate an existing CPC object, or the API user does not have object-access permission to it.
	4	The SE associated with the CPC designated by the request URI is not on the required code level to support this operation.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: "active" , "service-required" , "degraded" , or "exceptions"
	5	The operation cannot be performed because the CPC designated by the URI is not in DPM mode.
	8	An object with the same URI as one in the request body already exists.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/46a42974-59b1-3574-bade-0d2cc3e2f12c/operations/import-dpm-config HTTP/1.1
x-api-session: 1bj062lh2pj8kjin9sntoytdimuli2edkwom7vpw0cntdh8j
content-type: application/json
content-length: 5791
{
  "adapter-mapping":[
    {
      "new-adapter-id":"1dd",
      "old-adapter-id":"100"
    }
  ],
  "adapters":[
    {
      "adapter-family":"hipersockets",
      "adapter-id":"7c0",
      "allowed-capacity":12288,
      "channel-path-id":"09",
      "class":"adapter",
      "configured-capacity":81,
      "description":"",
      "detected-card-type":"hipersockets",
      "maximum-total-capacity":12288,
      "maximum-transmission-unit-size":8,
      "name":"hipersocket",
      "network-port-uris":[
        "/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb/network-ports/0"
      ],
      "object-id":"bc5c79e6-354f-11e7-911e-00106f0d81cb",
      "object-uri":"/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb",
      "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
      "physical-channel-status":"operating",
      "port-count":1,
      "state":"online",
      "status":"active",
      "type":"hipersockets",
      "used-capacity":27
    }
  ],
}
```

Figure 479. Import DPM Configuration: Request (Part 1)

```

{
  "adapter-family":"ficon",
  "adapter-id":"100",
  "allowed-capacity":64,
  "card-location":"Z22B-D101-J.01",
  "channel-path-id":"0d",
  "class":"adapter",
  "configured-capacity":10,
  "description":"",
  "detected-card-type":"ficon-express-16s-plus",
  "maximum-total-capacity":254,
  "name":"FCP 0100 Z22B-01",
  "object-id":"a44e2648-0a42-11e7-88d2-00106f0d81cb",
  "object-uri":"/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb",
  "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
  "physical-channel-status":"operating",
  "port-count":1,
  "state":"online",
  "status":"active",
  "storage-port-uris":[
    "/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb/storage-ports/0"
  ],
  "type":"fcp",
  "used-capacity":6
},
{
  "available-features-list":[],
  "cpc-properties":{"
    "description":"CPC description "
  }
},
{
  "hbas":[
    {
      "adapter-port-uri":"/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb/storage-ports/0",
      "class":"hba",
      "description":"Systemplatte",
      "device-number":"0150",
      "element-id":"633219d0-5ff6-11e7-92e8-00106f0d81cb",
      "element-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/hbas/633219d0-5ff6-11e7-92e8-00106f0d81cb",
      "name":"Bootadapter",
      "parent":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
      "wwpn":"C05076FFE80006A6"
    }
  ]
},
],

```

Figure 480. Import DPM Configuration: Request (Part 2)

```

"nics":[
  {
    "class":"nic",
    "description":"Device fuer die freie Welt.",
    "device-number":"0001",
    "element-id":"5d0caee6-5ff4-11e7-90cf-00106f0d81cb",
    "element-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/nics/5d0caee6-5ff4-11e7-90cf-00106f0d81cb",
    "mac-address":"02:ff:e8:00:2f:31",
    "name":"Netzwerk_Aussen",
    "parent":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
    "ssc-ip-address":null,
    "ssc-ip-address-type":null,
    "ssc-management-nic":false,
    "ssc-mask-prefix":null,
    "type":"iqd",
    "virtual-switch-uri":"/api/virtual-switches/bc6d7ce6-354f-11e7-a1a3-00106f0d81cb",
    "vlan-id":null,
    "vlan-type":null
  }
],
"partitions":[
  {
    "acceptable-status":[
      "active"
    ],
    "access-basic-counter-set":false,
    "access-basic-sampling":false,
    "access-coprocessor-group-set":false,
    "access-crypto-activity-counter-set":false,
    "access-diagnostic-sampling":false,
    "access-extended-counter-set":false,
    "access-global-performance-data":false,
    "access-problem-state-counter-set":false,
    "auto-start":false,
    "autogenerate-partition-id":true,
    "boot-configuration-selector":0,
    "boot-device":"test-operating-system",
    "boot-ftp-host":null,
    "boot-ftp-insfile":null,
    "boot-ftp-username":null,
  }
]

```

Figure 481. Import DPM Configuration: Request (Part 3)

```
"boot-iso-image-name":null,
"boot-iso-ins-file":null,
"boot-logical-unit-number":"0001000000000000",
"boot-network-device":null,
"boot-os-specific-parameters":"","
"boot-record-lba":"","
"boot-removable-media":null,
"boot-removable-media-type":null,
"boot-storage-device":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/
    hbas/633219d0-5ff6-11e7-92e8-00106f0d81cb",
"boot-timeout":60,
"boot-world-wide-port-name":"50050763070306A6",
"class":"partition",
"cp-absolute-processor-capping":false,
"cp-absolute-processor-capping-value":1.0,
"cp-processing-weight-capped":false,
"cp-processors":6,
"crypto-configuration":null,
"current-cp-processing-weight":1,
"current-ifl-processing-weight":1,
"degraded-adapters":[],
"description":"Ihno legt eine LPAR an.",
"has-unacceptable-status":true,
"hba-uris":[
    "/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/hbas/633219d0-
        5ff6-11e7-92e8-00106f0d81cb"
],
```

Figure 482. Import DPM Configuration: Request (Part 4)

```

        "ifl-absolute-processor-capping":false,
        "ifl-absolute-processor-capping-value":1.0,
        "ifl-processing-weight-capped":false,
        "ifl-processors":0,
        "initial-cp-processing-weight":100,
        "initial-ifl-processing-weight":100,
        "initial-memory":12288,
        "is-locked":false,
        "maximum-cp-processing-weight":999,
        "maximum-ifl-processing-weight":999,
        "maximum-memory":12288,
        "minimum-cp-processing-weight":1,
        "minimum-ifl-processing-weight":1,
        "name":"SUSE_Test",
        "nic-uris":[
            "/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/nics/5d0caee6-5ff4-11e7-90cf-00106f0d81cb"
        ],
        "object-id":"9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
        "object-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
        "os-name":"",
        "os-type":"",
        "os-version":"",
        "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
        "partition-id":null,
        "permit-aes-key-import-functions":true,
        "permit-cross-partition-commands":false,
        "permit-des-key-import-functions":true,
        "processor-management-enabled":false,
        "processor-mode":"shared",
        "reserve-resources":false,
        "reserved-memory":0,
        "short-name":"SUSETEST",
        "status":"stopped",
        "threads-per-processor":0,
        "type":"linux",
        "virtual-function-uris":[]
    }
},
"se-version":"2.13.1",
"virtual-switches":[
    {
        "backing-adapter-uri":"/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb",
        "class":"virtual-switch",
        "description":"",
        "name":"7C0.P0.IQD",
        "object-id":"bc6d7ce6-354f-11e7-a1a3-00106f0d81cb",
        "object-uri":"/api/virtual-switches/bc6d7ce6-354f-11e7-a1a3-00106f0d81cb",
        "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
        "port":0,
        "type":"hipersockets"
    }
]
}

```

Figure 483. Import DPM Configuration: Request (Part 5)

```

204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 11 Jun 2018 10:36:04 GMT

<No response body>

```

Figure 484. Import DPM Configuration: Response

Usage notes

- There is a sample script (exportDpmResourcesToFile.py) to assist in creating the request body for this operation. It is available at <http://www.ibm.com/servers/resource link>. Select **Services**, then select **IBM Z APIs**, and select **Web Services export DPM configuration script**. This script is intended to be run

against a z13 or later system whose DPM configuration is to be imported to a more current system with this operation.

- By default, only the SERVICE user ID on the HMC has the required task permission for this operation, and that user ID is not enabled for Web Services APIs by default.
- By default, object IDs and element IDs are not preserved; instead, new unique IDs are generated when importing the objects. This prevents the possibility of duplicate IDs, which would occur when both the new and old systems are attached to the same HMC after the configuration is imported with this operation. The **preserve-uris** field can be specified as **true** to preserve the IDs, which can be helpful for situations that depend on the IDs, such as a network (PXE) boot. But care must be taken to never attach both systems to the same HMC if that option is used during this operation. Doing so may result in unpredictable behavior due to the presence of duplicate IDs.
- By default, the WWPNS of the HBAs are not preserved; instead, new WWPNS are generated when importing the HBAs. The generated WWPNS will match the I/O serial number of the new machine. If the new machine will have the same I/O serial number as the old machine, then it is appropriate to specify **preserve-wwpns** as **true** so that the HBAs are imported with their original WWPNS.

Inventory service data

Information about CPCs can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the CPC objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular CPC is included only if the API user has access permission to that object as described in the Get CPC Properties operation.

For each CPC object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get CPC Properties” on page 761](#). That is, the data provided is the same as would be provided if a Get CPC Properties operation were requested targeting this object.

Logical Partition object

The Processor Resource/Systems Manager (PR/SM) is a feature of IBM mainframes that enables logical partitioning of the CPC. A logical partition (LPAR) is a virtual machine at the hardware level. Each LPAR operates as an independent server running its own operating environment. Each LPAR runs its own operating system, which can be any mainframe operating system.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

This object includes the properties defined in [“Base managed object properties schema” on page 70](#), with the following class-specific specialization:

Table 375. Logical Partition object: base managed object properties specializations			
Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Logical Partition object, of the form <code>/api/logical-partitions/{<i>logical-partition-id</i>}</code> where <i>{logical-partition-id}</i> is the value of the object-id property of the Logical Partition object.
parent	—	String/ URI	The canonical URI path of the associated CPC object.

Table 375. Logical Partition object: base managed object properties specializations (continued)

Name	Qualifier	Type	Description of specialization
class	—	String	The class of a Logical Partition object is "logical-partition" .
name	(ro)	String (1-8)	The name of the logical partition
description	(ro)	String (0-1024)	The descriptive text associated with this object.
status	(sc)	String Enum	One of the following values: • "operating" - the logical partition has a active control program • "not-operating" - the logical partition's CPC is non operational • "not-activated" - the logical partition does not have an active control program • "exceptions" - the logical partition's CPC has one or more unusual conditions
acceptable-status	(w)(pc)	Array of String Enum	An array of one or more status strings that determine an acceptable status for a logical partition. When a logical partition's status property contains one of the specified acceptable-status values, the has-unacceptable-status property contains false.

Class specific additional properties

In addition to the properties defined via included schemas, this object includes the following additional class-specific properties. Refer to the *Processor Resource/Systems Manager Planning Guide* for more detailed explanations of the various properties.

There are additional notes throughout the table. Please refer to the note list at the end of the table.

Table 376. Logical Partition object: class specific additional properties

Name	Qualifier	Type	Description
os-name ¹	(pc)	String (0-8)	An operating system provided value, used to identify the operating system instance. The format of the value is operating system dependent. If not provided by the operating system, an empty string is returned.
os-type ¹	(pc)	String (0-8)	A human readable form of the operating system provided value for the type of the operating system active in this logical partition. If not provided, an empty string is returned.
os-level ¹	(pc)	String (0-32)	A human readable form of the operating system provided value for the level of the operating system active in this logical partition. If not provided, an empty string is returned.
sysplex-name ¹	(pc)	String (1-8)	Applicable only for z/OS or when os-type is "CFCC" , the name of the sysplex of which this logical partition is a member, or an empty string if the logical partition is not a member of a sysplex. For a logical partition that is not z/OS and os-type is not "CFCC" , a null object is returned.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
is-sub-capacity-boost-active ^{1, 11}	(pc)	Boolean	If true, sub-capacity boost is active. If false, sub-capacity boost is inactive.
is-secure-execution-enabled ^{1, 11}	(pc)	Boolean	If true, Secure Execution for Linux is enabled. If false, Secure Execution for Linux is not enabled.
is-ziip-capacity-boost-active ^{1, 11}	(pc)	Boolean	If true, zIIP-capacity boost is active. If false, zIIP-capacity boost is inactive.
has-operating-system-messages ¹	—	Boolean	If true, object has operating system messages. If false, object does not have operating system messages.
activation-mode	(pc)	String Enum	One of the following values: • "general" - the logical partition is in general mode • "esa390" - the logical partition is in ESA/390 mode • "esa390tpf" - the logical partition is in ESA/390 TPF mode • "coupling-facility" - the logical partition is running as a coupling facility • "linux" - the logical partition is in Linux mode • "zvm" - the logical partition is in z/VM mode • "zaware" - the logical partition is in IBM zAware mode • "ssc" - the logical partition is in IBM Secure Service Container mode • "not-set" - the logical partition is not activated.
next-activation-profile-name	(w)(pc)	String (1-16)	Image activation profile name or load activation profile name to be used on the next activate.
last-used-activation-profile	(pc)	String (0-16)	The last used activation profile name or an empty string.
last-used-load-address	—	String (5)	The load address most recently used by the logical partition or null if the value is not available. The load address is a hexadecimal string of the form <i>cdddd</i> where <i>c</i> is the channel subsystem ID and <i>dddd</i> is the device address of the I/O device that was loaded. If the logical partition has not been loaded, or was loaded from device 0 of channel subsystem 0, a value of "00000" is returned.
last-used-load-parameter	—	String (0-8)	The load parameter most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded, an empty string is returned.
last-used-secure-boot ¹¹	—	Boolean	Verify software signature with distributor most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded, null is returned.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
last-used-world-wide-port-name	—	String (1-16)	The worldwide port name (WWPN), in hexadecimal, of the target SCSI device most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded or dumped using an SCSI device, the string "0" is returned.
last-used-logical-unit-number	—	String (1-16)	The logical unit number (LUN), in hexadecimal, of the target SCSI device most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded or dumped using an SCSI device, the string "0" is returned.
last-used-disk-partition-id	—	Integer (0-30)	The disk-partition-id (also called the boot program selector) of the target SCSI device most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded or dumped using an SCSI device, 0 is returned.
last-used-operating-system-specific-load-parameters	—	String (0-256)	The operating system specific load parameters of the target SCSI device most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded or dumped using an SCSI device, an empty string is returned.
last-used-boot-record-logical-block-address	—	String (1-16)	The boot record logical block address, in hexadecimal, of the target SCSI device most recently used by the logical partition or null if the value is not available. If the logical partition has not been loaded or dumped using an SCSI device, the string "0" is returned.
initial-processing-weight ^{1, 2, 3, 9}	(w)	Integer (1-999)	The relative amount of shared general purpose processor resources allocated to the logical partition.
initial-processing-weight-capped ^{1, 2, 3, 4}	(w)	Boolean	Whether the initial processing weight for general purpose processors is a limit or a target. True Indicates that the initial general purpose processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources. False Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
minimum-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared general purpose processor resources allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be less than or equal to the initial-processing-weight property.
maximum-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Defines the maximum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be greater than or equal to the initial-processing-weight property.
current-processing-weight ^{1, 3}	—	Integer (1-999)	The relative amount of shared general purpose processor resources currently allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
current-processing-weight-capped ^{1, 2, 3}	—	Boolean	Whether the current general purpose processing weight is a limit or a target. True Indicates that the current general purpose processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current general purpose processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
workload-manager-enabled ^{1, 5, 9}	(w)	Boolean	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. True Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. False Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition. Note: When the logical partition is in an activation mode that does not support Workload Manager (e.g. Coupling Facility), including this property in the Update request may not have any effect.
absolute-processing-capping	(w)	absolute-capping object	The amount of absolute capping applied to the general purpose processor. Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
defined-capacity ¹	(w)	Integer	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed to by z/OS Workload Manager for the purpose of software pricing. 0 No defined capacity is specified for this logical partition. 1-nnnn Represents the amount of defined capacity specified for this logical partition.
cluster-name ¹	(pc)	String (0-8)	LPAR cluster name, which identifies membership in a group of logical partitions that are members of the same z/OS Parallel Sysplex®.
partition-number ¹	(pc)	String (2)	The partition number for the logical partition, in hexadecimal.
partition-identifier ¹	(pc)	String (2)	The partition identifier for the logical partition, in hexadecimal.
processor-usage ^{1, 11, 12}	—	String Enum	How processors are allocated to the logical partition. One of the following values: • "dedicated" - all processor types in the logical partition are to be exclusively available to this specific logical partition. • "shared" - all processor types in the logical partition are to be shareable across logical partitions.
number-general-purpose-processors ^{1, 11, 12}	—	Integer	The number of general purpose processors allocated to the logical partition.
number-reserved-general-purpose-processors ^{1, 11, 12}	—	Integer	The number of general purpose processors reserved for the logical partition.
number-general-purpose-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of general purpose processor cores allocated to the logical partition.
number-reserved-general-purpose-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of general purpose processor cores reserved for the logical partition.
number-icf-processors ^{1, 11, 12}	—	Integer	The number of Internal Coupling Facility (ICF) processors allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
number-reserved-icf-processors ^{1, 11, 12}	—	Integer	The number of Internal Coupling Facility (ICF) processors reserved for the logical partition.
number-icf-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of Internal Coupling Facility (ICF) processor cores allocated to the logical partition.
number-reserved-icf-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of Internal Coupling Facility (ICF) processor cores reserved for the logical partition.
number-ifl-processors ^{1, 11, 12}	—	Integer	The number of Integrated Facility for Linux (IFL) processors allocated to the logical partition.
number-reserved-ifl-processors ^{1, 11, 12}	—	Integer	The number of Integrated Facility for Linux (IFL) processors reserved for the logical partition.
number-ifl-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of Integrated Facility for Linux (IFL) processor cores allocated to the logical partition.
number reserved-ifl-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of Integrated Facility for Linux (IFL) processor cores reserved for the logical partition.
number-ziip-processors ^{1, 11, 12}	—	Integer	The number of z Integrated Information Processor (zIIP) processors allocated to the logical partition.
number-reserved-ziip-processors ^{1, 11, 12}	—	Integer	The number of z Integrated Information Processor (zIIP) processors reserved for the logical partition.
number-ziip-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of z Integrated Information Processor (zIIP) processor cores allocated to the logical partition.
number-reserved-ziip-cores ^{1, 11, 12, 13, 14}	(w)	Integer	The number of z Integrated Information Processor (zIIP) processor cores reserved for the logical partition.
initial-aap-processing-weight ^{1, 2, 3, 9}	(w)	Integer (1-999)	The relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
initial-aap-processing-weight-capped ^{1, 2, 3, 4}	(w)	Boolean	Whether the initial processing weight for Application Assist Processor (zAAP) processors is a limit or a target. True Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Application Assist Processor (zAAP) processor resources, regardless of the availability of excess Application Assist Processor (zAAP) processor resources. False Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is not capped. It represents the share of Application Assist Processor (zAAP) processor resources guaranteed to a logical partition when all Application Assist Processor (zAAP) processor resources are in use. Otherwise, when excess Application Assist Processor (zAAP) processor resources are available, the logical partition can use them if necessary.
minimum-aap-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Update: 0 No minimum value for the processing weight. 1-999 Define the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
maximum-aap-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
current-aap-processing-weight ^{1, 3}	—	Integer (1-999)	The current relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
current-aap-processing-weight-capped ^{1, 3}	—	Boolean	Whether the current Application Assist Processor (zAAP) processing weight is a limit or a target. True Indicates that the current Application Assist Processor (zAAP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current Application Assist Processor (zAAP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-aap-capping	(w)	absolute-capping object	The amount of absolute capping applied to the Application Assist Processor (zAAP). Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
initial-ift-processing-weight ^{1, 2, 3, 9}	(w)	Integer (1-999)	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
initial-ift-processing-weight-capped ^{1, 2, 3, 4}	(w)	Boolean	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. True Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources False Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.
minimum-ift-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
maximum-ifl-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
current-ifl-processing-weight ^{1, 3}	—	Integer (1-999)	The current relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
current-ifl-processing-weight-capped ^{1, 3}	—	Boolean	Whether the current Integrated Facility for Linux (IFL) processing weight is a limit or a target. True Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-ifl-capping	(w)	absolute-capping object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor. Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
initial-cbp-processing-weight ^{1, 2, 3, 9, 10}	(w)	Integer (1-999)	The relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.
initial-cbp-processing-weight-capped ^{1, 2, 3, 4, 10}	(w)	Boolean	Whether the initial processing weight for Container Based Processors (CBP) is a limit or a target. True Indicates that the initial Container Based Processor's (CBP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of Container Based Processor (CBP) resources, regardless of the availability of excess Container Based Processor (CBP) resources. False Indicates that the initial Container Based Processor's (CBP) processing weight for the logical partition is not capped. It represents the share of Container Based Processor (CBP) resources guaranteed to a logical partition when all Container Based Processor (CBP) resources are in use. Otherwise, when excess Container Based Processor (CBP) resources are available, the logical partition can use them if necessary.
minimum-cbp-processing-weight ^{1, 2, 3, 9, 10}	(w)	Integer	The minimum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
maximum-cbp-processing-weight ^{1, 2, 3, 9, 10}	(w)	Integer	The maximum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.
current-cbp-processing-weight ^{1, 3, 10}	—	Integer	The current relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.
current-cbp-processing-weight-capped ^{1, 2, 3, 10}	—	Boolean	Whether the current Container Based Processor's (CBP) processing weight is a limit or a target. True Indicates that the current Container Based Processor's (CBP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current Container Based Processor's (CBP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-cbp-capping ¹⁰	(w)	absolute-capping-object	The amount of absolute capping applied to the Container Based Processor's (CBP). Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
initial-ziip-processing-weight ^{1, 2, 3, 9}	(w)	Integer (1-999)	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
initial-ziip-processing-weight-capped ^{1, 2, 3, 4}	(w)	Boolean	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. True Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. False Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.
minimum-ziip-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
maximum-ziip-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
current-ziip-processing-weight ^{1, 3}	—	Integer (1-999)	The current relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
current-ziip-processing-weight-capped ^{1, 3}	—	Boolean	Whether the current z Integrated Information Processors (zIIP) processing weight is a limit or a target. True Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-ziip-capping	(w)	absolute-capping object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor. Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
initial-cf-processing-weight ^{1, 2, 3, 9}	(w)	Integer (1-999)	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
initial-cf-processing-weight-capped ^{1, 2, 3, 4}	(w)	Boolean	Indicates whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. True Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. False Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition is not capped. It represents the share of Internal Coupling Facility (ICF) processor resources guaranteed to a logical partition when all Internal Coupling Facility (ICF) processor resources are in use. Otherwise, when excess Internal Coupling Facility (ICF) processor resources are available, the logical partition can use them if necessary.
minimum-cf-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: 1-999 The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
maximum-cf-processing-weight ^{1, 2, 3, 9}	(w)	Integer	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
current-cf-processing-weight ^{1, 3}	—	Integer (1-999)	The current relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
current-cf-processing-weight-capped ^{1, 3}	—	Boolean	Indicates whether the current Internal Coupling Facility (ICF) processing weight is a limit or a target. True Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. False Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-cf-capping	(w)	absolute-capping object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor. Note: Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
program-status-word-information ¹	—	Array of psw-description objects	Describes the current PSW information for each CP associated with the logical partition. The information is obtained on each Get Logical Partition Properties request and is not cached. Refer to the description of the psw-description object for details.
initial-vfm-storage ⁸	—	Long	The initial amount of Virtual Flash Memory (VFM) storage, in gigabytes (GB), to be allocated to this logical partition at activation. The valid range is 0 to the value indicated on the storage-vfm-total property in a multiple of the value indicated on the storage-vfm-increment-size property for the associated CPC.
maximum-vfm-storage ⁸	—	Long	The maximum amount of VFM storage, in gigabytes (GB), that can be allocated to this logical partition while it is running. The valid range is 0 to the value indicated on the storage-vfm-total property in a multiple of the value indicated on the storage-vfm-increment-size property for the associated CPC.
current-vfm-storage ⁸	—	Long	The current amount of VFM storage, in gigabytes (GB), that is allocated to this logical partition. The valid range is 0 to the value indicated on the storage-vfm-total property in a multiple of the value indicated on the storage-vfm-increment-size property for the associated CPC.
os-ipl-token ¹	(pc)	String (1-16)	Applicable only to z/OS, a value provided when z/OS is IPLed that uniquely identifies the instance of the operating system. Used by z/OS to obtain knowledge about the status of another system in the sysplex, and upon the demise of the system, potentially partition the system out of the sysplex immediately and reset the demised system. The value is a string of hexadecimal characters (0-9,A-Z), left justified.
group-profile-capacity ¹	—	Integer	The current value of the effective-capacity property of the Group Profile with which the logical partition is associated. A null object is returned if the logical partition is not assigned to an LPAR group.
group-profile-uri ¹	—	String/ URI	The canonical URI of the Group Profile associated with the logical partition. A null object is returned if the logical partition is not assigned to an LPAR group.
zaware-host-name ⁶	(w)	String (1-64)	The IBM zAware host name. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and colon(:)
zaware-master-userid ⁶	(w)	String (1-32)	The IBM zAware master userid. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and underscore (_)
zaware-master-pw ⁶	(wo)	String (8-256)	The IBM zAware master password. Valid characters are: a-z,A-Z,0-9 and !@#\$%^&*()_+{} <>?=- This property is not returned on a Get request, it can only be specified on an Update request.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
zaware-network-info⁶	(w)	Array of zaware-network objects	The set of networks available to IBM zAware. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
zaware-gateway-info⁶	(w)	ip-info object	The default gateway IP address information. A null object indicates no default gateway IP address is specified.
zaware-dns-info⁶	(w)	Array of ip-info objects	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.
ssc-host-name⁷	(w) or — if se-version is "2.14.0" or later	String (1-64)	The Secure Service Container name. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and colon(:).
ssc-master-userid⁷	(w) or — if se-version is "2.14.0" or later	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and underscore(_).
ssc-master-pw⁷	(wo) or — if se-version is "2.14.0" or later	String (8-256)	The Secure Service Container master user password. Valid characters are: a-z, A-Z, 0-9, and !@#\$%^&*()_+{} <>?-=. This property is not returned on a Get request; it can be specified on an Update request.
ssc-network-info⁷	(w) or — if se-version is "2.14.0" or later	Array of ssc-network objects	The set of networks available to the Secure Service Container. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
ssc-gateway-info⁷	(w) or — if se-version is "2.14.0" or later	ip-info object	The default gateway IP address information for the Secure Service Container. A null object indicates no default gateway IP address is specified. Only IPv4 address types are supported.
ssc-dns-info⁷	(w) or — if se-version is "2.14.0" or later	Array of ip-info objects	The DNS IP address information for the Secure Service Container. A minimum of 0 entries and a maximum of 2 entries are permitted On an Update request, this property fully replaces the existing set.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
storage-central-allocation	—	Array of central-storage-allocation objects	A nested object that details the central storage allocated to the logical partition. Refer to Table 377 on page 849 for details. If there are no central-storage-allocation objects, a null object is returned.
storage-expanded-allocation	—	Array of expanded-storage-allocation objects	A nested object that details the expanded storage allocated to the logical partition. Refer to Table 378 on page 850 for details. If there are no expanded-storage-allocation objects, a null object is returned.

Table 376. Logical Partition object: class specific additional properties (continued)

Name	Qualifier	Type	Description
Notes: 1. If the logical partition status property is "not-activated", a null object is returned instead of the documented field type. 2. An Update of this property is only valid for an object-id that represents a logical partition with at least one shared processor of the corresponding type. 3. The value returned from a Get request is a null object for an object-id that does not represent a logical partition with at least one shared processor of the corresponding type. 4. This property and the workload-manager-enabled property are mutually exclusive and cannot both be enabled at the same time. Therefore in order to enable this property it might be necessary to first disable the workload-manager-enabled property. 5. This property and the various capping properties are mutually exclusive and cannot be enabled at the same time. Therefore in order to enable this property it may be necessary to first disable any capping property that is currently enabled. 6. On a Get request, this property is returned only when activation-mode is "zaware". On an Update request, this property can be updated only when activation-mode is "zaware". 7. On a Get request, this property is returned only when activation-mode is "ssc". On an Update request, this property can be updated only when activation-mode is "ssc". 8. On a Get request, this property is returned only when the SE version is 2.14.0 or later. 9. On an Update request, when workload-manager-enabled is true, the initial processing weight must be greater than or equal to the minimum processing weight, and less than or equal to the maximum processing weight within the same processor type. 10. On a Get request, this property is returned only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. 11. This property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. 12. If the corresponding data is not available, a null object is returned. 13. On an Update request, the input value must be greater than the value that is currently in the logical partition. 14. On an Update request, if a new processor type is being defined (both the current numbers of the allocated and reserved cores of a processor type are 0, and at least one of them is being increased), the following conditions must be met: • The target logical partition is activated. • The activation mode of the target logical partition supports multiple processor types and the newly defined type is one of them. • If the current processor-usage is "shared", the initial processing weight of the corresponding processor type is also included in the request.			

Table 377. central-storage-allocation nested object properties

Name	Qualifier	Type	Description
origin	—	Long	The origin in megabytes of central storage in memory or null if the value is not available.
initial	—	Long	The initial amount of central storage in megabytes that is allocated or null if the value is not available.

Table 377. central-storage-allocation nested object properties (continued)			
Name	Qualifier	Type	Description
current	—	Long	The current amount of central storage in megabytes that is allocated or null if the value is not available.
maximum	—	Long	The maximum amount of central storage in megabytes that is allocated or null if the value is not available.
gap	—	Long	The gap in megabytes from this partition's central storage to the start of the next partition's central storage, or null if the value is not available.
storage-element-type	—	String Enum	Type of storage element. Valid values are: • "central" - central storage • "initial" - initial expanded storage that is used as central storage • "reserved" - reserved expanded storage that is used as central storage.

Table 378. expanded-storage-allocation nested object properties			
Name	Qualifier	Type	Description
origin	—	Long	The origin in megabytes of expanded storage in memory.
initial	—	Long	The initial amount of expanded storage in megabytes that is allocated.
current	—	Long	The current amount of expanded storage in megabytes that is allocated.
maximum	—	Long	The maximum amount of expanded storage in megabytes that is allocated.
gap	—	Long	The gap in megabytes from this partition's expanded storage to the start of the next partition's expanded storage.

List Logical Partitions of CPC

The List Logical Partitions of CPC operation lists the logical partitions of a CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/logical-partitions
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC.

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
logical-partitions	Array of logical-partition-info objects	Array of nested logical-partition-info objects (described in the next table)

Each nested logical-partition-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the Logical Partition object
name	String	The name of the Logical Partition object
status	String Enum	The current status of the Logical Partition object

Description

This operation lists the Logical Partition objects that belong to a CPC. The object URI, display name, and status are provided for each.

If the **name** query parameter is specified, the returned list is limited to those Logical Partition objects that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

An object is only included in the list if the API user has object-access permission for that object.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 850](#).

If the CPC is in DPM mode, or there are no logical partitions defined to the CPC, or no logical partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Object-access permission to any Logical Partition object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 850](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

HTTP error status code	Reason code	Description
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/logical-partitions HTTP/1.1
x-api-session: 65aw2jahugn1wop51hsq0c6alchkx773dz9ulirrv2z853m4u
```

Figure 485. List Logical Partitions of CPC: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:58:36 GMT
content-type: application/json;charset=UTF-8
content-length: 374
{
  "logical-partitions": [
    {
      "name": "APIVM1",
      "object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
      "status": "operating"
    },
    {
      "name": "ZOS",
      "object-uri": "/api/logical-partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd",
      "status": "not-operating"
    }
  ]
}
```

Figure 486. List Logical Partitions of CPC: Response

List Permitted Logical Partitions

The List Permitted Logical Partitions operation lists logical partitions to which the API user has object-access permission.

HTTP method and URI

```
GET /api/console/operations/list-permitted-logical-partitions
```

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name property.
activation-mode	String Enum	Optional	Filter string to limit returned objects to those that have a matching activation-mode property. Value must be a valid logical partition activation-mode property value.

Name	Type	Rqd/Opt	Description
status	String Enum	Optional	Filter string to limit returned objects to those that have a matching status property. Value must be a valid logical partition status property value.
has-unacceptable-status	Boolean	Optional	Filter string to limit returned objects to those that have a matching has-unacceptable-status property. Valid values are true and false .
cpc-name	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching name property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
logical-partitions	Array of logical-partition-info objects	Array of nested logical-partition-info objects as described in the next table.

Each nested logical-partition-info object contains the following fields:

Field name	Type	Description
name	String	The name property of the Logical Partition object.
object-uri	String/URI	The object-uri property of the Logical Partition object.
activation-mode	String Enum	The activation-mode property of the Logical Partition object.
status	String Enum	The status property of the Logical Partition object.
has-unacceptable-status	Boolean	The has-unacceptable-status property of the Logical Partition object.
cpc-name	String	The name property of the Logical Partition's parent CPC object.
cpc-object-uri	String/URI	The object-uri property of the Logical Partition's parent CPC object.
se-version	String	The se-version property of the Logical Partition's parent CPC object.

Description

This operation lists the Logical Partition objects to which the API user has object-access permission. Some basic properties are provided for each logical partition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those logical partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

If the **activation-mode** query parameter is specified, the parameter is validated to ensure it is a valid logical partition **activation-mode** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those logical partitions that have

an **activation-mode** property matching the specified value. If the **activation-mode** parameter is omitted, no such filtering is performed.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid logical partition **status** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those logical partitions that have a **status** property matching the specified value. If the **status** parameter is omitted, no such filtering is performed.

If the **has-unacceptable-status** query parameter is specified, the returned list is limited to those logical partitions that have a **has-unacceptable-status** property matching the specified value. If the **has-unacceptable-status** parameter is omitted, no such filtering is performed.

If the **cpc-name** query parameter is specified, the returned list is limited to those logical partitions whose parent CPC's **name** property matches the specified filter pattern. If the **cpc-name** parameter is omitted, no such filtering is performed.

A logical partition is included in the list only if the API user has object-access permission to that object. If there is a logical partition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no logical partitions known to the HMC or if no logical partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Logical Partition objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 853](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/console/operations/list-permitted-logical-partitions HTTP/1.1
x-api-session: 3y0qwkkon9m03c5o81oo2fz7ozezkinczdwwr4gauzgogb5xq
```

Figure 487. List Permitted Logical Partitions: Request

```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 28 Aug 2017 18:07:25 GMT
content-type: application/json; charset=UTF-8
content-length: 860
{
  "logical-partitions":[
    {
      "activation-mode":"general",
      "cpc-name":"M87",
      "cpc-object-uri":"/api/cpcs/f2eb7e56-0c87-3646-887e-ec735b3844cd",
      "has-unacceptable-status":false,
      "name":"S50",
      "object-uri":"/api/logical-partitions/d39347d9-855a-3199-9ef6-a1701b7b17b4",
      "se-version":"2.14.0",
      "status":"operating"
    },
    {
      "activation-mode":"esa390",
      "cpc-name":"S15",
      "cpc-object-uri":"/api/cpcs/f6f629ca-f2c5-3f71-a80f-d9b91a492549",
      "has-unacceptable-status":false,
      "name":"APIVM2",
      "object-uri":"/api/logical-partitions/c53c7aa7-444b-3f05-87b4-fb94802240b8",
      "se-version":"2.13.1",
      "status":"operating"
    },
    {
      "activation-mode":"not-set",
      "cpc-name":"S15",
      "cpc-object-uri":"/api/cpcs/f6f629ca-f2c5-3f71-a80f-d9b91a492549",
      "has-unacceptable-status":true,
      "name":"Z0S",
      "object-uri":"/api/logical-partitions/17c4bc30-96b5-327e-8b7d-59e1b4b7261e",
      "se-version":"2.13.1",
      "status":"not-activated"
    }
  ]
}

```

Figure 488. List Permitted Logical Partitions: Response

Usage note

The response body of this operation is similar to that of the `Get Inventory` operation, but it returns only a subset of logical partition properties. The response also includes some properties of the parent CPC, regardless of whether the API user has object-access permission to that CPC.

Get Logical Partition Properties

The `Get Logical Partition Properties` operation retrieves the properties of a single Logical Partition object designated by `{logical-partition-id}`.

HTTP method and URI

GET `/api/logical-partitions/{logical-partition-id}`

In this request, the URI variable `{logical-partition-id}` is the object ID of the target Logical Partition object.

Query parameters:

Name	Type	Rqd/Opt	Description
properties	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the Logical Partition object's data model.

Name	Type	Rqd/Opt	Description
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are true and false .

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body provides the current values of the properties for the Logical Partition object as defined in [“Data model” on page 828](#).

Description

The URI path must designate an existing Logical Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some logical partition property values are periodically fetched from the Support Element and cached on the HMC for quick access by the APIs. Due to the nature of this caching support, the HMC's cached value of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined in [“Data model” on page 828](#).

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 856](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9 HTTP/1.1  
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 489. Get Logical Partition Properties: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Tue, 05 Mar 2019 17:16:16 GMT
content-type: application/json;charset=UTF-8
content-length: 3509
{
  "absolute-aap-capping":{"type": "none"},
  "absolute-cbp-capping":{"type": "none"},
  "absolute-cf-capping":{"value": 88.52, "type": "processors"},
  "absolute-ifl-capping":{"type": "none"},
  "absolute-processing-capping":{"value": 0.01, "type": "processors"},
  "absolute-ziip-capping":{"value": 2.01, "type": "processors"},
  "acceptable-status": [
    "operating"
  ],
  "activation-mode": "esa390",
  "additional-status": "",
  "class": "logical-partition",
  "cluster-name": "",
  "current-aap-processing-weight": null,
  "current-aap-processing-weight-capped": null,
  "current-cbp-processing-weight": null,
  "current-cbp-processing-weight-capped": null,
  "current-cf-processing-weight": null,
  "current-cf-processing-weight-capped": null,
  "current-ifl-processing-weight": null,
  "current-ifl-processing-weight-capped": null,
  "current-processing-weight": 100,
  "current-processing-weight-capped": false,
  "current-vfm-storage": 256,
  "current-ziip-processing-weight": null,
  "current-ziip-processing-weight-capped": null,
  "defined-capacity": 0,
  "description": "LPAR Image",
  "group-profile-capacity": null,
  "group-profile-uri": null,
  "has-operating-system-messages": false,
  "has-unacceptable-status": false,
  "initial-aap-processing-weight": null,
  "initial-aap-processing-weight-capped": null,
  "initial-cbp-processing-weight": null,
  "initial-cbp-processing-weight-capped": null,
  "initial-cf-processing-weight": null,
  "initial-cf-processing-weight-capped": null,
  "initial-ifl-processing-weight": null,
  "initial-ifl-processing-weight-capped": null,
  "initial-processing-weight": 100,
  "initial-processing-weight-capped": false,
  "initial-vfm-storage": 64,
  "initial-ziip-processing-weight": null,
  "initial-ziip-processing-weight-capped": null,
  "is-locked": false,
  "is-sub-capacity-boost-active": false,
  "is-ziip-capacity-boost-active": false,
  "last-used-activation-profile": "APIVM1",
  "last-used-boot-record-logical-block-address": "C8",
  "last-used-disk-partition-id": 0,
  "last-used-load-address": "05402",
  "last-used-load-parameter": "TESTMODE",
  "last-used-logical-unit-number": "0015000000000000",
  "last-used-operating-system-specific-load-parameters": "0002,0,fa163e057ee54",
  "last-used-secure-boot": false,
  "last-used-world-wide-port-name": "50017380EB0B0142",
  "maximum-aap-processing-weight": null,
  "maximum-cbp-processing-weight": null,
  "maximum-cf-processing-weight": null,
  "maximum-ifl-processing-weight": null,

```

Figure 490. Get Logical Partition Properties: Response (Part 1)


```

"maximum-processing-weight": 200,
"maximum-vfm-storage": 512,
"maximum-ziip-processing-weight": null,
"minimum-aap-processing-weight": null,
"minimum-cbp-processing-weight": null,
"minimum-cf-processing-weight": null,
"minimum-ifl-processing-weight": null,
"minimum-processing-weight": 50,
"minimum-ziip-processing-weight": null,
"name": "APIVM1",
"next-activation-profile-name": "APIVM1",
"number-general-purpose-cores": 5,
"number-general-purpose-processors": 5,
"number-icf-cores": 0,
"number-icf-processors": 0,
"number-ifl-cores": 0,
"number-ifl-processors": 0,
"number-reserved-general-purpose-cores": 2,
"number-reserved-general-purpose-processors": 2,
"number-reserved-icf-cores": 0,
"number-reserved-icf-processors": 0,
"number-reserved-ifl-cores": 0,
"number-reserved-ifl-processors": 0,
"number-reserved-ziip-cores": 1,
"number-reserved-ziip-processors": 1,
"number-ziip-cores": 1,
"number-ziip-processors": 1,
"object-id": "c7eb8134-826e-3a71-8d1a-00d706c874e9",
"object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
"os-ipl-token": "0000000000000000",
"os-level": "6.2.0",
"os-name": "APIVM1",
"os-type": "z/VM",
"parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
"partition-number": 1,
"processor-usage": "shared",
"program-status-word-information": [
  {
    "cpid": "00",
    "psw": "07064000800000000000000000000000"
  },
  {
    "cpid": "01",
    "psw": "07064000800000000000000000000000"
  }
],
"status": "operating",
"storage-central-allocation": [
  {
    "origin": 16697344,
    "initial": 1024,
    "current": 1024,
    "maximum": 1024,
    "gap": 1024,
    "storage-element-type": "central"
  },
  {
    "origin": 16696832,
    "initial": null,
    "current": 512,
    "maximum": null,
    "gap": 0,
    "storage-element-type": "initial"
  }
],

```

Figure 491. Get Logical Partition Properties: Response (Part 2)

```

    "storage-expanded-allocation": [
      {
        "origin": 16776704,
        "initial": 512,
        "current": 512,
        "maximum": 512,
        "gap": 0
      }
    ],
    "sysplex-name": "SSICAPI1",
    "workload-manager-enabled": true
  }

```

Figure 492. Get Logical Partition Properties: Response (Part 3)

Update Logical Partition Properties

The Update Logical Partition Properties operation updates one or more writable properties of the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain one or more field names representing writable logical partition properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Logical Partition object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- If the **next-activation-profile-name** property is to be updated, action/task permission for the **Change Object Options** task or the **Customize/Delete Activation Profiles** task.
- Object-access permission to the logical partition's parent CPC object.
- If any of the **ssc-*** and **zaware-*** properties is to be updated, action/task permission for the **Firmware Details** task.
- If any of the numbers of allocated or reserved cores is included, action/task permission for the **Logical Processor Add** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Activate Logical Partition

The Activate Logical Partition operation activates the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/activate
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
activation-profile-name	String (1-16)	Optional	The name of the activation profile to be used for the request. If not provided, the request uses the profile name specified in the next-activation-profile-name property for the Logical Partition object.
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 863. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Activation is a process that makes a logical partition operational, which means either:

- The logical partition is ready to have a control program or operating system loaded, or
- The logical partition has loaded and is running a control program or operating system.

Activating a logical partition includes:

- Initializing the logical partition
- Allocating system resources to the logical partition
- Loading the logical partition with a control program or operating system.

Since the status of the logical partition determines which operations must be performed during activation to make the logical partition operational, one or more operations listed above may not be performed during activation.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See ["Query Job Status"](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See ["Job status and reason codes"](#) on page 863.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Activate** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in ["Response body contents"](#) on page 861.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Deactivate Logical Partition

The Deactivate Logical Partition operation deactivates the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/deactivate
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 865. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Deactivation is an orderly process for terminating a logical partition.

Deactivating a logical partition includes:

- Unloading the logical partition's control program or operating system
- Freeing system resources allocated to the logical partition.

After the logical partition is deactivated, the logical partition is no longer operational

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See ["Query Job Status"](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See ["Job status and reason codes"](#) on page 865.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Deactivate** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in ["Response body contents"](#) on page 863.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Reset Normal

The `Reset Normal` operation initializes a system or logical partition by clearing its pending interruptions, resetting its channel subsystem and resetting its processors. A reset prepares a system or logical partition for loading it with an operating system.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/reset-normal
```

In this request, the URI variable `{logical-partition-id}` is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.
os-ipl-token	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the os-ipl-token property. This ensures that this operation is targeting the same IPL instance as when the os-ipl-token property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the os-ipl-token parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 867. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status”](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes”](#) on page 867.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Reset Normal** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 865.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Reset Clear

The `Reset Clear` operation initializes system or logical partition by clearing its pending interruptions, resetting its channel subsystem and resetting its processors. A reset prepares a system or logical partition for loading it with an operating system and clears main memory of the system or logical partition.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/reset-clear
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.
os-ipl-token	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the os-ipl-token property. This ensures that this operation is targeting the same IPL instance as when the os-ipl-token property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the os-ipl-token parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9,A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 868. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status”](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes”](#) on page 868.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Reset Clear** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 867.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.

HTTP error status code	Reason code	Description
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Load Logical Partition

The Load Logical Partition operation resets a logical partition, to prepare it for loading an operating system, and loads the operating system.

HTTP method and URI

POST /api/logical-partitions/{*logical-partition-id*}/operations/load

In this request, the URI variable {*logical-partition-id*} is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
load-address	String (1-5)	Optional	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". When load-address is not supplied, the operation will try to load using the address used in the last load.
load-parameter	String (1-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Omitting this field indicates that the value for this field is to be retrieved from the current IOCDs. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, \$, #) are also allowed when the se-version property of the associated CPC is " 2.14.0 " or later.
clear-indicator	Boolean	Optional	Whether memory should be cleared before performing the Load (true) or not cleared (false). The default value is true.
timeout	Integer (60-600)	Optional	Amount of time, in seconds, to wait for the Load to complete. The default timeout value is 60 seconds.
store-status-indicator	Boolean	Optional	Whether status should be stored before performing the Load (true) or not stored (false). The default is false.

Field name	Type	Rqd/ Opt	Description
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.
os-ipl-token	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the os-ipl-token property. This ensures that this operation is targeting the same IPL instance as when the os-ipl-token property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the os-ipl-token parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 871. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See ["Query Job Status"](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See ["Job status and reason codes"](#) on page 871.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Load** task.

- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 870.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
	306	This operation is not valid in the current activation mode.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

PSW Restart

The PSW Restart operation restarts the first available processor of the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/psw-restart
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 873. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Information about PSW Restart can be found on console help system.

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See ["Query Job Status"](#) on page 103 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See ["Job status and reason codes"](#) on page 873.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **PSW Restart** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in ["Response body contents"](#) on page 871.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 50 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Start Logical Partition

The `Start Logical Partition` operation starts the processors to process instructions of the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/start
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes”](#) on page 874. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes” on page 874](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Start** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 873](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Stop Logical Partition

The Stop Logical Partition operation stops the processors from processing instructions of the Logical Partition object designated by *{logical-partition-id}*.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/stop
```


In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 876](#). The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes” on page 876](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Stop** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 875](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Send OS Command

The Send OS Command operation sends a command to the operating system running in a logical partition.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/send-os-cmd
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
is-priority	Boolean	Optional	An indication of whether this is a priority operating system command. Set to true for priority operating system commands or false for non-priority operating system commands. The default is false .
operating-system-command-text	String (1-200)	Required	The text of the operating system command.

Description

This operation sends a command to the operating system running in the Logical Partition targeted by the request URI.

The URI path must designate an existing Logical Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

In addition to having object-access permission to the Logical Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the Logical Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

On successful execution, the command is sent to the operating system running in the target Logical Partition object and status code 204 (No Content) is returned without supplying a response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	332	The messages interface is not available.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9/operations/send-os-cmd
HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifo4qymqa8ktzjep7dbyrl10k
content-type: application/json
content-length: 69
{
  "is-priority": false,
  "operating-system-command-text": "help"
}
```

Figure 493. Send OS Command: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT

<No response body>
```

Figure 494. Send OS Command: Response

Open OS Message Channel

The Open OS Message Channel operation opens a message channel to the operating system running in a logical partition.

HTTP method and URI

POST /api/logical-partitions/{*logical-partition-id*}/operations/open-os-message-channel

In this request, the URI variable {*logical-partition-id*} is the object ID of the target Logical Partition object.

Request body contents

An optional request body can be specified as a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
include-refresh-messages	Boolean	Optional	An indication of whether refresh operating system messages should be sent. Set to true to receive refresh messages, or false to prevent refresh messages. The default is true .

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
topic-name	String (1-128)	The name of the os-message-notification topic.

Description

This operation opens a message channel to the operating system running in the Logical Partition targeted by the request URI. The message channel is implemented as a JMS topic, specifically as an os-message-notification topic. See [Chapter 4, “Asynchronous notification,”](#) on page 57 for information on JMS usage on the HMC. The API user can connect to this topic to start the flow of new (and refreshed) operating system messages.

The URI path must designate an existing Logical Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Logical Partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the Logical Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If an os-message-notification topic already exists for this logical partition for the current API session, the operation fails.

On successful execution, the message channel is opened and the os-message-notification topic name is returned in the response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 878.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 379. Open OS Message Channel: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	331	An os-message-notification topic already exists for this logical partition for the current API session. Use the Get Notification Topics operation to determine the topic name.
	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9/operations/  
  open-os-message-channel HTTP/1.1  
x-api-session: 2ltfe2c2q3ti2b2pq1wfwuzifo14qymqa8ktzjep7dbyr1l0k  
content-type: application/json  
content-length: 37  
{ "include-refresh-messages": true }
```

Figure 495. Open OS Message Channel: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json; charset=UTF-8
{ "topic-name": "mikeuser.1osmsg.cpc1.lpar1" }
```

Figure 496. Open OS Message Channel: Response

List OS Messages of a Logical Partition

The List OS Messages of a Logical Partition operation lists all currently available operating system (OS) messages for a logical partition.

HTTP method and URI

GET /api/logical-partitions/{*logical-partition-id*}/operations/list-os-messages

In this request, the URI variable {*logical-partition-id*} is the object ID of the target logical partition.

Query parameters:

Name	Type	Rqd/ Opt	Description
begin-sequence-number	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number less than this are omitted from the results. If not specified, then no such filtering is performed.
end-sequence-number	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number greater than this are omitted from the results. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
os-messages	Array of os-message-info objects	Array of nested os-message-info objects as described in the next table. The array elements are in order of increasing sequence number, unless that number has wrapped back around to 0 in which case the element with sequence number 0 follows the element with the highest sequence number, thus maintaining the temporal order of the messages.

Each nested os-message-info object contains the following fields:

Field name	Type	Description
sequence-number	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message. It can be used for filtering purposes.
message-text	String	The text of the operating system message.
message-id	String	The message identifier of the operating system message.

Field name	Type	Description
timestamp	Timestamp	The time when the operating system message was created or -1 if this information is not available from the corresponding operating system.
sound-alarm	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (true) or not (false).
is-priority	Boolean	Specifies whether the operating system message is a priority message (true) or not (false). A priority message indicates a critical condition that requires immediate attention.
is-held	Boolean	Specifies whether the operating system message is a held message (true) or not (false). A held message is one that requires a response.
prompt-text	String	Specifies the prompt text that is associated with this operating system message or null indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.
os-name	String	Specifies the name of the operating system that generated this operating system message or null indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the logical partition in which the operating system is running.

Description

This operation lists the currently available messages from the operating system running in the specified logical partition. Only a certain amount of OS message data from each logical partition is preserved by the HMC for retrieval by this operation. If the OS produces more than that amount, the oldest non-held, non-priority OS messages are no longer available. A gap in the sequence numbers indicates a loss of messages. A loss may be due to that space limitation, or it may be due to the deletion of messages by a console user or the OS.

If the request URI does not identify a Logical Partition object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned. In addition to having object-access permission to the logical partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the logical partition is not active, there is no operating system running in the logical partition, or when the operating system is not enabled for console integration.

If the **begin-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** less than that are omitted from the response. If the **end-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** greater than that are omitted from the response.

If there are no available OS messages for the specified logical partition or if no OS messages are to be included in the response due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*.
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 880](#).

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/logical-partitions/d39347d9-855a-3199-9ef6-a1701b7b17b4/operations/
    list-os-messages HTTP/1.1
x-api-session: 2cckypxdonb44w6n8dvly3vpw9hekwr2soasrhax41oetlbni
```

Figure 497. List OS Messages of a Logical Partition: Request


```

200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 28 Aug 2017 19:59:00 GMT
content-type: application/json; charset=UTF-8
content-length: 665
{
  "os-messages": [
    {
      "is-held": true,
      "is-priority": false,
      "message-id": 69166,
      "message-text": "*IOS002A AB4D,NO PATHS AVAILABLE\n",
      "os-name": "S50",
      "prompt-text": "",
      "sequence-number": 10541,
      "sound-alarm": false,
      "timestamp": 1503926847000
    },
    {
      "is-held": true,
      "is-priority": false,
      "message-id": 69167,
      "message-text": "*IOS002A AB11,NO PATHS AVAILABLE\n",
      "os-name": "S50",
      "prompt-text": "",
      "sequence-number": 10542,
      "sound-alarm": false,
      "timestamp": 1503926847000
    },
    {
      "is-held": true,
      "is-priority": false,
      "message-id": 69169,
      "message-text": "*IOS002A AB42,NO PATHS AVAILABLE\n",
      "os-name": "S50",
      "prompt-text": "",
      "sequence-number": 10543,
      "sound-alarm": false,
      "timestamp": 1503926847000
    }
  ]
}

```

Figure 498. List OS Messages of a Logical Partition: Response

Delete Logical Partition OS Message

The Delete Logical Partition OS Message operation deletes a single logical partition OS message.

HTTP method and URI

POST /api/logical-partitions/{*logical-partition-id*}/operations/delete-os-message

In this request, the URI variable {*logical-partition-id*} is the object ID of the target logical partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
sequence-number	Long	Required	The sequence-number property of the OS message on the logical partition to delete.

Description

This operation deletes a specific logical partition OS message. The OS message to be deleted is uniquely identified by the combination of the *{logical-partition-id}* variable in the URI and the **sequence-number** in the request body.

The URI path must designate an existing logical partition and the API user must have object-access permission to it; otherwise status code 404 (Not Found) is returned.

The request body must designate an existing OS message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the Operating System Messages task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*.
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission for the Operating System Messages task.
404 (Not Found)	1	The object ID in the URI <i>{logical-partition-id}</i> does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
	336	The sequence-number in the request body does not designate an existing OS message on the logical partition.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
POST /api/logical-partitions/4767c72e-b00d-3a1c-ac89-87f821404a0b/operations/delete-os-message
HTTP/1.1
x-api-session: 6d1q521ruuym4dhqubv3m77678qbjx7c68wpdpv5v75ut8n1iq
content-type: application/json
content-length: 22
{
  "sequence-number":0
}
```

Figure 499. Delete Logical Partition OS Message: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 15 Nov 2018 17:21:00 GMT

<No response body>
```

Figure 500. Delete Logical Partition OS Message: Response

SCSI Load

The SCSI Load operation prepares the logical partition for loading an operating system and loads the operating system from the designated SCSI device.

HTTP method and URI

POST /api/logical-partitions/{*logical-partition-id*}/operations/scsi-load

In this request, the URI variable {*logical-partition-id*} is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
load-address	String (1-5)	Required	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
load-parameter	String (1-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Omitting this field indicates that the value for this field is to be retrieved from the current IOCDs. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, \$, #) are also allowed when the se-version property of the associated CPC is " 2.14.0 " or later.
secure-boot	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. The default value is false . This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
clear-indicator	Boolean	Optional	Whether memory should be cleared before performing the Load (true) or not cleared (false). The default value is true. Note: This field is allowed only when the se-version property of the associated CPC is " 2.14.1 " or later.

Field name	Type	Rqd/Opt	Description
world-wide-port-name	String (1-16)	Required	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal.
logical-unit-number	String (1-16)	Required	The hexadecimal logical unit number (LUN) to be used for the SCSI Load.
disk-partition-id	Integer (0-30)	Optional	The disk-partition-id (also called the boot program selector) to be used for the SCSI Load. The default value is 0.
operating-system-specific-load-parameters	String (1-256)	Optional	The operating system specific load parameters to be used for the SCSI Load. The default value is an empty string.
boot-record-logical-block-address	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the SCSI Load. The default value is hex zeros.
force	Boolean	Optional	Whether this operation is permitted when the logical partition is in "operating" status (true) or not (false). The default is false.
os-ipl-token	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the os-ipl-token property. This ensures that this operation is targeting the same IPL instance as when the os-ipl-token property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the os-ipl-token parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-F), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in ["Job status and reason codes"](#) on page 887. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes” on page 887](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Load** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 886](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

SCSI Dump

The SCSI Dump operation loads a standalone dump program from a designated SCSI device.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/scsi-dump
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
load-address	String (1-5)	Required	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
load-parameter	String (1-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Omitting this field indicates that the value for this field is to be retrieved from the current IOCDS. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, \$, #) are also allowed when the se-version property of the associated CPC is "2.14.0" or later.
secure-boot	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. The default value is false. This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
world-wide-port-name	String (1-16)	Required	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal.
logical-unit-number	String (1-16)	Required	The hexadecimal logical unit number (LUN) to be used for the SCSI Dump.
disk-partition-id	Integer (0-30)	Optional	The disk-partition-id (also called the boot program selector) to be used for the SCSI Dump. The default value is 0.
operating-system-specific-load-parameters	String (1-256)	Optional	The operating system specific load parameters to be used for the SCSI Dump. The default value is an empty string.
boot-record-logical-block-address	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the SCSI Dump. The default value is hex zeros.
os-ipl-token	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the os-ipl-token property. This ensures that this operation is targeting the same IPL instance as when the os-ipl-token property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the os-ipl-token parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
job-uri	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in [“Job status and reason codes” on page 890](#). The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See [“Query Job Status” on page 103](#) for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See [“Job status and reason codes” on page 890](#).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Load** task.
- Object-access permission to the logical partition's parent CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents” on page 889](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{logical-partition-id}</i>) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

List Managed Virtual Machines of a Logical Partition

The List Managed Virtual Machines of a Logical Partition operation lists the Managed Virtual Machines hosted by the z/VM system running in a specified logical partition.

HTTP method and URI

```
GET /api/logical-partitions/{logical-partition-id}/managed-virtual-machines
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the logical partition whose Managed Virtual Machines are to be listed.

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
managed-virtual-machines	Array of managed - virtual-machine-info objects	Array of nested managed-virtual-machine-info objects as described in the next table.

Each nested managed-virtual-machine-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	Canonical URI path of the Managed Virtual Machine object is of the form /api/managed-virtual-machines/cpc{cpc_serial}-{lpar_name}-{mvm_name} where: • {cpc_serial} - is the serial number of the CPC containing the logical partition running the z/VM system containing the Managed Virtual Machine. • {lpar_name} - is the name of the logical partition running the z/VM system containing the Managed Virtual Machine. • {mvm_name} - is the name of the Managed Virtual Machine.
name	String	The name of the Managed Virtual Machine.

Description

This operation lists the Managed Virtual Machines hosted by the z/VM system running in the specified logical partition. Some basic information is provided for each Managed Virtual Machine that is included in the response.

If the request URI does not identify a Logical Partition object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

If the **name** query parameter is specified, the returned list is limited to those Managed Virtual Machines whose name matches the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

A Managed Virtual Machine is included in the list only if the API user has object-access permission to that object. If there is a Managed Virtual Machine to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no Managed Virtual Machines hosted by the z/VM system running in the specified logical partition or if no Managed Virtual Machines are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object identified in the request URI.
- Object-access permission to the Managed Virtual Machine objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 890](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 380. List Managed Virtual Machines of a Logical Partition: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

Table 380. List Managed Virtual Machines of a Logical Partition: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
404 (Not Found)	1	The request URI does not designate an existing logical partition, or it designates one for which the API user does not have object-access permission.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Usage Notes

The APIs provide only minimal support for Managed Virtual Machine objects. There is no data model, and most of the standard operations are not provided. The List Managed Virtual Machines of a Logical Partition operation can be used to list them and to provide their name and object URI. The provided support also allows an API client to manage access to them through the Add Permission to User Role and Remove Permission from User Role operations.

Example HTTP interaction

```
GET /api/logical-partitions/6ed3d9e6-fc2d-3458-8804-fee9d44369c7/managed-  
virtual-machines HTTP/1.1  
x-api-session: 6c1rf8lcmc4tzcpt02a9iz04h5y91qpc8v6mjdbm1lz4pywd0a
```

Figure 501. List Managed Virtual Machines of a Logical Partition: Request

```
200 OK  
server: zSeries management console API web server / 2.0  
cache-control: no-cache  
date: Mon, 14 Dec 2015 19:44:21 GMT  
content-type: application/json; charset=UTF-8  
content-length: 210  
{  
  "managed-virtual-machines": [  
    {  
      "name": "TEST1",  
      "object-uri": "/api/managed-virtual-machines/cpc0000200E9EB6-VMALT1-TEST1"  
    },  
    {  
      "name": "EREP",  
      "object-uri": "/api/managed-virtual-machines/cpc0000200E9EB6-VMALT1-EREP"  
    }  
  ]  
}
```

Figure 502. List Managed Virtual Machines of a Logical Partition: Response

Inventory service data

Information about logical partitions can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Logical Partition objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"logical-partition"** are to be included. An entry for a particular logical partition is included only if the API user has access permission to that object as described in the Get Logical Partition Properties operation.

For each Logical Partition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Logical Partition Properties”](#) on page 855. That is, the data provided is the same as would be provided if a Get Logical Partition Properties operation were requested targeting this object.

Reset activation profile

A Reset activation profile is used by a CPC Activate operation to control the activation of a CPC and, if properly configured with one or more image activation profiles, a set of Logical Partition(s).

An activation profile can only be created or deleted from the Hardware Management Console or the Support Element.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics”](#) on page 68.

This element includes the following properties.

Table 381. Reset activation profile: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Reset Activation Profile object, of the form <code>/api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}</code> where <code>{reset-activation-profile-name}</code> is the value of the name property (Reset Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object.
class	—	String	The class of a Reset Activation Profile object is "reset-activation-profile" .
name	—	String (1-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by <code>{cpc-id}</code> .
description	(w)	String (1-50)	The reset profile description
iocds-name	(w)	String (0-2)	The Input/Output Configuration Data Set name, in hexadecimal. An empty string indicates that the currently active IOCDS will be used. The active IOCDS is the one from the most recent power-on-reset of the CPC or, if using dynamic I/O configuration, the one last activated.
processor-running-time-type	(w)	String Enum	Defines whether the processor running time is determined dynamically or set manually for the CPC (see processor-running-time in this table). One of: "system-determined" "user-determined"

Table 381. Reset activation profile: properties (continued)

Name	Qualifier	Type	Description
processor-running-time	(w)	Integer (0-100)	Amount of continuous time, in milliseconds, for logical processors to perform jobs on shared processors for the CPC, if processor-running-time-type is set to "user-determined" . If processor-running-time-type is "system-determined" , this property's value will always be returned as 0.
end-timeslice-on-wait	(w) or — if se-version is "2.14.0" or later.	Boolean	If true and if processor-running-time-type is set to "user-determined" , CPC Logical Partitions lose their share of running time when they enter a wait state. If processor-running-time-type is "system-determined" , this property's value will always be returned as false. Note: When the se-version property of the associated CPC is "2.14.0" or later, this property is not permitted on an Update Reset Activation Profile Properties operation, and its value is always false .

List Reset Activation Profiles

The List Reset Activation Profiles operation lists the Reset Activation Profiles associated with a particular CPC.

HTTP method and URI

GET /api/cpcs/{cpc-id}/reset-activation-profiles

In this request, the URI variable {cpc-id} is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
reset-activation-profiles	Array of reset-actprof-info objects	Array of nested objects (described in the following table).

Each reset-actprof-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Reset Activation Profile object.
name	String	The name of the Reset Activation Profile.

Description

This operation lists the Reset Activation Profiles associated with a particular CPC.

If the **name** query parameter is specified, the returned list is limited to those Reset Activation Profiles that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 894](#).

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 894](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 503. List Reset Activation Profiles: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:16 GMT
content-type: application/json; charset=UTF-8
content-length: 372
{
  "reset-activation-profiles": [
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-
profiles/
      DEFAULT",
      "name": "DEFAULT"
    },
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-
profiles/
      POWER_ON_RESET",
      "name": "POWER_ON_RESET"
    }
  ]
}

```

Figure 504. List Reset Activation Profiles: Response

Get Reset Activation Profile Properties

The **Get Reset Activation Profile Properties** operation retrieves the properties of a single Reset Activation Profile designated by *{reset-activation-profile-name}*.

HTTP method and URI

GET /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}

URI variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{reset-activation-profile-name}</i>	Reset Activation Profile name

Query parameters:

Name	Type	Rqd/Opt	Description
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are true and false . The default is false .

Response body contents

On successful completion, the response body provides the current values of the properties for the Reset Activation Profile as defined in the [“Data model”](#) on page 893.

Description

The URI path must designate an existing Reset Activation Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

Some of this object's property values are periodically fetched from the Support Element and cached on the HMC for quick access by the APIs. Due to the nature of this caching support, the HMC's cached value

of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Reset Activation Profile object.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 896](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (<i>{reset-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles/DEFAULT HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 505. Get Reset Activation Profile Properties: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 384
{
  "class": "reset-activation-profile",
  "description": "This is the default Reset profile.",
  "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles/DEFAULT",
  "end-timeslice-on-wait": false,
  "iocds-name": "a0",
  "name": "DEFAULT",
  "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
  "processor-running-time": 0,
  "processor-running-time-type": "system-determined"
}

```

Figure 506. Get Reset Activation Profile Properties: Response

Update Reset Activation Profile Properties

The Update Reset Activation Profile Properties operation updates one or more writable properties of the Reset Activation Profile designated by *{reset-activation-profile-name}*.

HTTP method and URI

POST /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}

URI variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{reset-activation-profile-name}</i>	Reset Activation Profile name

Request body contents

The request body is expected to contain one or more field names representing writable Reset Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Reset Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Reset Activation Profile is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Customize/Delete Activation Profiles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	19	The request body contains a field whose corresponding data model property is not writable on this HMC and/or SE version.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (<i>{reset-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Inventory service data

Information about reset activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Reset Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular reset activation profile is included only if the API user has access permission to that object as described in the [Get Reset Activation Profile Properties](#) operation.

For each Reset Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Reset Activation Profile Properties”](#) on page 896. That is, the data provided is the same as would be

provided if a Get Reset Activation Profile Properties operation were requested targeting this object.

Image activation profile

An Image activation profile is used by an Activate operation to activate a logical partition of a previously activated CPC.

An activation profile can only be created or deleted from the Hardware Management Console or the Support Element.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

This element includes the following properties. Some properties have additional notes associated with them. Refer to the table notes at the end of this table.

Table 382. Image activation profile: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Image Activation Profile object, of the form <code>/api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}</code> where <code>{image-activation-profile-name}</code> is the value of the name property (Image Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object.
class	—	String	The class of an Image Activation Profile object is "image-activation-profile" .
name	—	String (0-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by <code>{cpc-id}</code> .
description	(w)	String (0-50)	The activation profile description
ipl-address ¹⁵	(w)	String (0-5)	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value will be right justified and padded with zeros to 5 characters. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
ipl-parameter	(w)	String (0-8)	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, \$, #) are also allowed when the se-version property of the associated CPC is "2.14.0" or later. On an Update, a non-empty string is left justified and right padded with blanks to 8 characters.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
initial-processing-weight¹	(w)	Integer	The relative amount of shared general purpose processor resources allocated to the logical partition. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared general purpose processor. 1-999 Represents the relative amount of shared general purpose processor resources initially allocated to the logical partition. Update: 1-999 Define the relative amount of shared general purpose processor resources allocated to the logical partition.
initial-processing-weight-capped^{1, 2, 3}	(w)	Boolean	Whether the initial processing weight for general purpose processors is a limit or a target. True: Indicates that the initial general purpose processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources. False: Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.
minimum-processing-weight¹	(w)	Integer	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared general purpose processor resources allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be less than or equal to the initial-processing-weight property.
maximum-processing-weight¹	(w)	Integer	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Defines the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Must be greater than or equal to the initial-processing-weight property.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
absolute-general-purpose-capping ²²	(w)	absolute-capping object	The amount of absolute capping applied to the general purpose processor.
workload-manager-enabled ⁴	(w)	Boolean	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. True: Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. False: Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition.
defined-capacity	(w)	Integer	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed to by z/OS Workload Manager for the purpose of software pricing. 0: No defined capacity is specified for this logical partition. 1-nnnn: Represents the amount of defined capacity specified for this logical partition
iplt-type	(w)	String Enum	One of: • "ipltype-standard" - This image activation profile is used to perform a standard load. • "ipltype-scsi" - This image activation profile is used to perform a SCSI load. • "ipltype-scsidump" - This image activation profile is used to perform a SCSI dump.
worldwide-port-name ¹⁵	(w)	String (1-16)	Worldwide port name of the target SCSI device, used for a SCSI load or SCSI dump, in hexadecimal.
disk-partition-id	(w)	Integer (0-30)	The disk partition number (also called the boot program selector) for the activation profile, used for a SCSI load or SCSI dump.
logical-unit-number ¹⁵	(w)	String (1-16)	Logical unit number value for the activation profile, used for a SCSI load or SCSI dump, in hexadecimal.
boot-record-lba ¹⁵	(w)	String (1-16)	Boot record logical block address for the activation profile, used for a SCSI load or SCSI dump, in hexadecimal.
os-specific-load-parameters	(w)	String (0-256)	Operating system-specific load parameters for the activation profile, used for a SCSI load or SCSI dump. On an Update, value is left justified and right padded with blanks to 256 characters.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
initial-aap-processing-weight ^{5, 21}	(w)	Integer	The relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition at activation. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared Application Assist Processor (zAAP) processor. 1-999 Represents the relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: 1-999 Define the relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
initial-aap-processing-weight-capped ^{3, 5, 6, 21}	(w)	Boolean	Whether the initial processing weight for Application Assist Processor (zAAP) processors is a limit or a target. True: Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Application Assist Processor (zAAP) processor resources, regardless of the availability of excess Application Assist Processor (zAAP) processor resources. False: Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is not capped. It represents the share of Application Assist Processor (zAAP) processor resources guaranteed to a logical partition when all Application Assist Processor (zAAP) processor resources are in use. Otherwise, when excess Application Assist Processor (zAAP) processor resources are available, the logical partition can use them if necessary.
minimum-aap-processing-weight ^{5, 21}	(w)	Integer	The minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: 0 No minimum value for the processing weight. 1-999 Define the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
maximum-aap-processing-weight ^{5, 21}	(w)	Integer	The maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
absolute-aap-capping ^{21, 22}	(w)	absolute-capping object	The amount of absolute capping applied to the Application Assist Processor (zAAP).
initial-ifl-processing-weight ⁷	(w)	Integer	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition at activation. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared Integrated Facility for Linux (IFL) processor. 1-999 Represents the relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: 1-999 Define the relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
initial-ifl-processing-weight-capped ^{3, 7, 8}	(w)	Boolean	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. True: Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources. False: Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
minimum-ifl-processing-weight⁷	(w)	Integer	The minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
maximum-ifl-processing-weight⁷	(w)	Integer	The maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
absolute-ifl-capping²²	(w)	absolute-capping object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
initial-internal-cf-processing-weight⁹	(w)	Integer	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition at activation. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared Internal Coupling Facility (ICF) processor. 1-999 Represents the relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: 1-999 Define the relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
initial-internal-cf-processing-weight-capped ^{3, 9, 10}	(w)	Boolean	Indicates whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. True: Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Image associated with the logical partition is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. False: Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the logical partition is not capped. It represents the share of Internal Coupling Facility (ICF) processor resources guaranteed to a logical partition when all Internal Coupling Facility (ICF) processor resources are in use. Otherwise, when excess Internal Coupling Facility (ICF) processor resources are available, the logical partition can use them if necessary.
minimum-internal-cf-processing-weight ⁹	(w)	Integer	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition at activation. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: 1-999 Define the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.
maximum-internal-cf-processing-weight ⁹	(w)	Integer	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.
absolute-icf-capping ²²	(w)	absolute-capping object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
initial-cbp-processing-weight ^{23, 24}	(w)	Integer	The relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition at activation. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared Container Based Processor (CBP). 1-999 Represents the relative amount of shared Container Based Processor (CBP) resources initially allocated to the logical partition. Update: 1-999 Defines the relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.
initial-cbp-processing-weight-capped ^{23, 24, 25}	(w)	Boolean	Whether the initial processing weight for Container Based Processors (CBP) is a limit or a target. True: Indicates that the initial Container Based Processor's (CBP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of Container Based Processor (CBP) resources, regardless of the availability of excess Container Based Processor (CBP) resources. False: Indicates that the initial Container Based Processor's (CBP) processing weight for the logical partition is not capped. It represents the share of Container Based Processor (CBP) resources guaranteed to a logical partition when all Container Based Processor (CBP) resources are in use. Otherwise, when excess Container Based Processor (CBP) resources are available, the logical partition can use them if necessary.
minimum-cbp-processing-weight ^{23, 24}	(w)	Integer	The minimum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared Container Based Processor (CBP) resources initially allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Defines the minimum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
maximum-cbp-processing-weight ^{23, 24}	(w)	Integer	The maximum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared Container Based Processor (CBP) resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Defines the maximum relative amount of shared Container Based Processor (CBP) resources allocated to the logical partition.
absolute-cbp-capping ^{22, 23}	(w)	absolute-capping-object	The amount of absolute capping applied to the Container Based Processor (CBP).
initial-ziip-processing-weight ¹¹	(w)	Integer	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition at activation. Get: 0 The Image Activation Profile does not represent a logical partition with at least one shared z Integrated Information Processors (zIIP) processor. 1-999 Represents the relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: 1-999 Define the relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
initial-ziip-processing-weight-capped ^{3, 11, 12}	(w)	Boolean	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. True: Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. False: Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
minimum-ziip-processing-weight¹¹	(w)	Integer	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: 0 There is no minimum value for the processing weight. 1-999 Represents the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: 0 There is no minimum value for the processing weight. 1-999 Define the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
maximum-ziip-processing-weight¹¹	(w)	Integer	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: 0 There is no maximum value for the processing weight. 1-999 Represents the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: 0 There is no maximum value for the processing weight. 1-999 Define the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
absolute-ziip-capping²²	(w)	absolute-capping object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor.
group-profile-uri	(w)	String/ URI	The canonical URI of the Group profile to be used for the logical partition activated by this profile, which provides the group capacity value. On a Get, a null object is returned if no Group profile is associated with this activation profile. On an Update, a null object indicates that no Group profile is to be associated with this activation profile.
load-at-activation	(w)	Boolean	If true, the logical partition will be loaded at the end of the activation.
secure-boot²⁸	(w)	Boolean	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. This property is only applied to ipl-type of ipltype-scsi or ipltype-scsidump.
central-storage	(w)	Integer	Defines the amount of central storage, measured in megabytes (MB), to be allocated for the logical partition's exclusive use at activation. This value must be a multiple of the storage granularity value.
reserved-central-storage	(w)	Integer	Defines the amount of central storage, measured in megabytes (MB), dynamically reconfigurable to the logical partition after activation. This value must be a multiple of the storage granularity value.
expanded-storage	(w)	Integer	Defines the amount of expanded storage, measured in megabytes (MB), to be allocated for the logical partition's exclusive use at activation. This value must be a multiple of the storage granularity value.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
reserved-expanded-storage	(w)	Integer	Defines the amount of expanded storage, measured in megabytes (MB), dynamically reconfigurable to the logical partition after activation. This value must be a multiple of the storage granularity value.
initial-vfm-storage ¹⁸	(w)	Long	Defines the amount of Virtual Flash Memory (VFM) storage, measured in gigabytes (GB), to be allocated for the logical partition's exclusive use at activation. The valid range is 0 to the value indicated on the storage-vfm-total property in a multiple of the value indicated on the storage-vfm-increment-size property for the associated CPC. This value must not be greater than the value specified in the maximum-vfm-storage property in this Image Activation Profile.
maximum-vfm-storage ¹⁸	(w)	Long	Defines the maximum amount of VFM storage, measured in gigabytes (GB), that can be allocated to the logical partition while it is running. The valid range is 0 to the value indicated on the storage-vfm-total property in a multiple of the value indicated on the storage-vfm-increment-size property for the associated CPC. This value must not be smaller than the value specified in the initial-vfm-storage property in this Image Activation Profile.
processor-usage	(w)	String Enum	Defines how processors are allocated to the logical partition. One of the following values: • "dedicated" - all processor types in the logical partition are to be exclusively available to this specific logical partition. • "shared" - all processors types in the logical partition are to be shareable across logical partitions.
number-dedicated-general-purpose-processors ¹³	(w)	Integer	Defines the number of general purpose processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-general-purpose-processors ¹³	(w)	Integer	Defines the number of dedicated general purpose processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-aap-processors ^{13, 21}	(w)	Integer	Defines the number of Application Assist Processor (zAAP) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-aap-processors ^{13, 21}	(w)	Integer	Defines the number of dedicated Application Assist Processor (zAAP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-ifl-processors ¹³	(w)	Integer	Defines the number of Integrated Facility for Linux (IFL) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-ifl-processors ¹³	(w)	Integer	Defines the number of dedicated Integrated Facility for Linux (IFL) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-icf-processors ¹³	(w)	Integer	Defines the number of Integrated Coupling Facility (ICF) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-icf-processors ¹³	(w)	Integer	Defines the number of dedicated Integrated Coupling Facility (ICF) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-cbp-processors ^{13, 23}	(w)	Integer	Defines the number of Container Based Processors (CBP) to be allocated for the local partition's exclusive use at activation.
number-reserved-dedicated-cbp-processors ^{13, 23}	(w)	Integer	Defines the number of dedicated Container Based Processors (CBP) to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-ziip-processors ¹³	(w)	Integer	Defines the number of z Integrated Information Processors (zIIP) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-ziip-processors ¹³	(w)	Integer	Defines the number of dedicated z Integrated Information Processors (zIIP) processors to be reserved for the logical partition, which can be dynamically configured after activation.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
number-shared-general-purpose-processors¹⁴	(w)	Integer	Defines the number of shared general purpose processors to be allocated for the logical partition at activation.
number-reserved-shared-general-purpose-processors¹⁴	(w)	Integer	Defines the number of shared general purpose processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-aap-processors^{14, 21}	(w)	Integer	Defines the number of shared Application Assist Processor (zAAP) processors to be allocated for the logical partition at activation.
number-reserved-shared-aap-processors^{14, 21}	(w)	Integer	Defines the number of shared Application Assist Processor (zAAP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-ifl-processors¹⁴	(w)	Integer	Defines the number of shared Integrated Facility for Linux (IFL) processors to be allocated for the logical partition at activation.
number-reserved-shared-ifl-processors¹⁴	(w)	Integer	Defines the number of shared Integrated Facility for Linux (IFL) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-icf-processors¹⁴	(w)	Integer	Defines the number of shared Integrated Coupling Facility (ICF) processors to be allocated for the logical partition at activation.
number-reserved-shared-icf-processors¹⁴	(w)	Integer	Defines the number of shared Integrated Coupling Facility (ICF) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-cbp-processors^{14, 23}	(w)	Integer	Defines the number of shared Container Based Processors (CBP) to be allocated for the logical partition at activation.
number-reserved-shared-cbp-processors^{14, 23}	(w)	Integer	Defines the number of shared Container Based Processors (CBP) to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-ziip-processors¹⁴	(w)	Integer	Defines the number of shared z Integrated Information Processors (zIIP) processors to be allocated for the logical partition at activation.
number-reserved-shared-ziip-processors¹⁴	(w)	Integer	Defines the number of shared z Integrated Information Processors (zIIP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
basic-cpu-counter-authorization-control	(w)	Boolean	If true, the basic CPU counter facility for the logical partition is enabled.
problem-state-cpu-counter-authorization-control	(w)	Boolean	If true, the problem state CPU counter facility for the logical partition is enabled.
crypto-activity-cpu-counter-authorization-control	(w)	Boolean	If true, the crypto activity CPU counter facility for the logical partition is enabled.
extended-cpu-counter-authorization-control	(w)	Boolean	If true, the extended CPU counter facility for the logical partition is enabled.
coprocessor-cpu-counter-authorization-control	(w)	Boolean	If true, the coprocessor group CPU counter facility for the logical partition is enabled.
basic-cpu-sampling-authorization-control	(w)	Boolean	If true, the basic CPU sampling facility for the logical partition is enabled.
permit-aes-key-import-functions²⁷	(w)	Boolean	If true, importing of Advanced Encryption Standard (AES) keys for the logical partition is enabled.
permit-des-key-import-functions²⁷	(w)	Boolean	If true, importing of Data Encryption Standard (DES) keys for the logical partition is enabled.
permit-ecc-key-import-functions^{26, 27}	(w)	Boolean	If true, importing of Elliptic Curve Cryptography (ECC) keys for the logical partition is enabled.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
liccc-validation-enabled	(w)	Boolean	If true, ensure that the image profile data conforms to the current maximum Licensed Internal Code Configuration Control (LICCC) configuration.
global-performance-data-authorization-control	(w)	Boolean	If true, the logical partition can be used to view the processing unit activity data for all other logical partitions activated on the same CPC.
io-configuration-authorization-control	(w)	Boolean	If true, the logical partition can be used to read and write any Input/Output Configuration Data Set (IOCDs) in the configuration.
cross-partition-authority-authorization-control	—	Boolean	If true, the logical partition can be used to issue control program instructions that reset or deactivate other logical partitions.
logical-partition-isolation-control	(w)	Boolean	If true, reconfigurable channel paths assigned to the logical partition are reserved for its exclusive use.
operating-mode	(w)	String Enum	The operating mode for the logical partition: • "general" • "esa390" • "esa390-tpf" • "coupling-facility" • "linux-only" • "zvm" • "zaware" • "ssc" For an Update operation, "ssc" is only valid when the associated Support Element is at version 2.13.1 or later and "zaware" is only valid when the associated Support Element is at version 2.13.0 or earlier. For an Update operation, "general" is only valid when the associated Support Element is at version 2.14.0 or later and "esa390" or "esa390-tpf" is only valid when the associated Support Element is at version 2.13.1 or earlier.
clock-type	(w)	String Enum	One of: • "standard" - Set the logical partition's clock is set to the same time set for the CPC's time source. • "lpar" - Set the logical partition's clock using an offset from the External Time Source's time of day.
time-offset-days	(w)	Integer (0-999)	The number of days the logical partition's clock is to be offset from the External Time Source's time of day.
time-offset-hours	(w)	Integer (0-23)	The number of hours the logical partition's clock is to be offset from the External Time Source's time of day.
time-offset-minutes	(w)	Integer Enum	The number of minutes the logical partition's clock is to be offset from the External Time Source's time of day. Allowable values are 0, 15, 30 or 45.
time-offset-increase-decrease	(w)	String Enum	One of: • "increase" - Set the logical partition's clock ahead of the External Time Source's time of day. • "decrease" - Set the logical partition's clock back from the External Time Source's time of day.
zaware-host-name ¹⁹	(w)	String (1-64)	The IBM zAware host name. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and colon(:)
zaware-master-userid ¹⁹	(w)	String (1-32)	The IBM zAware master userid. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and underscore(_)

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
zaware-master-pw ¹⁹	(wo)	String (8-256)	The IBM zAware master password. Valid characters are: a-z,A-Z,0-9 and !@#\$\$%^&*()_+{} <>?-= This property is not returned on a Get request, it can only be specified on an Update request.
zaware-network-info ¹⁹	(w)	Array of zaware-network objects	The set of networks available to IBM zAware. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
zaware-gateway-info ¹⁶	(w)	ip-info object	The default gateway IP address information. A null object indicates no default gateway IP address is specified.
zaware-dns-info ¹⁶	(w)	Array of ip-info objects	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.
ssc-host-name ²⁰	(w)	String (1-64)	The Secure Service Container host name. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-) and colon(:).
ssc-master-userid ²⁰	(w)	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-) and underscore(_).
ssc-master-pw ²⁰	(wo)	String (8-256)	The Secure Service Container master password. Valid characters are: a-z, A-Z, 0-9, and !@#\$\$%^&*()_+{} <>?-=. This property is not returned on a Get request; it can be specified on an Update request.
ssc-network-info ²⁰	(w)	Array of ssc-network objects	The set of networks available to the Secure Service Container. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
ssc-gateway-info ¹⁷	(w)	ip-info objects	The IPv4 default gateway IP address information for the Secure Service Container. A null object indicates no IPv4 default gateway IP address is specified.
ssc-gateway-ipv6-info ^{17, 18}	(w)	ip-info objects	The IPv6 default gateway IP address information for the Secure Service Container. A null object indicates no IPv6 default gateway IP address is specified.
ssc-dns-info ¹⁷	(w)	Array of ip-info objects	The DNS IP address information for the Secure Service Container. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.
ssc-boot-selection ²⁰	(w)	String Enum	Indicates whether to run the Secure Service Container appliance installer or the Secure Service Container appliance itself. One of: • "installer" - Boot the Secure Service Container appliance installer to install the Secure Service Container appliance and then start it. • "appliance" - Reload the most recently installed Secure Service Container appliance and resume its execution from where it was when the image was deactivated.

Table 382. Image activation profile: properties (continued)

Name	Qualifier	Type	Description
Notes:			
1. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared general purpose processor. 2. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition or the Image Activation Profile does not represent a logical partition with at least one shared general purpose processor. 3. This property and the workload-manager-enabled property are mutually exclusive and cannot both be enabled at the same time. Therefore in order to enable this property it might be necessary to first disable the workload-manager-enabled property. 4. This property and the various capping properties are mutually exclusive and cannot be enabled at the same time. Therefore, in order to enable this property it may be necessary to first disable any capping property that is currently enabled. 5. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Application Assist Processor (zAAP) processor. 6. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Application Assist Processor (zAAP) processor. 7. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Integrated Facility for Linux (IFL) processor. 8. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Integrated Facility for Linux (IFL) processor. 9. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Internal Coupling Facility (ICF) processor. 10. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Internal Coupling Facility (ICF) processor. 11. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared z Integrated Information Processors (zIIP) processor. 12. The value returned for a GET request is always false when the Image Activation profile does not represent a logical partition with at least one shared z Integrated Information Processors (zIIP) processor. 13. The value of this property is a null object if the processor-usage property is "shared" 14. The value of this property is a null object if the processor-usage property is "dedicated" 15. An Update request accepts any mixture of [a-f,A-F,0-9], however the original string value is not saved and a subsequent Get request may not return the exact same set of lower/upper case letters. 16. On a Get request, this property is returned only when operating-mode is "zaware". On an Update request, this property can be specified only when operating-mode is "zaware". 17. On a Get request, this property is returned only when operating-mode is "ssc". On an Update request, this property can be specified only when operating-mode is "ssc". 18. On a Get request, this property is returned only when the SE version is 2.14.0 or later. On an Update request, this property is allowed only when the SE version is 2.14.0 or later. 19. On an Update request, this property can be specified only when operating-mode is set to "zaware" in the same request. 20. On an Update request, this property can be specified only when operating-mode is set to "ssc" in the same request. 21. On a Get request, this property is not returned when the SE version is 2.13.0 or later. On an Update request, this property is ignored when the SE version is 2.13.0 or later. 22. On a Get request, this property is not returned for a profile whose processor-usage property is "dedicated". On an Update request, this property is not allowed for a profile whose processor-usage property is "dedicated". 23. On a Get request, this property is returned only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. 24. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one Container Based Processor (CBP). 25. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one Container Based Processor (CBP). 26. On a Get Request, this property is returned only when the SE version is 2.15.0 or later. On an Update request, this property is only allowed when the SE version is 2.15.0 or later. 27. Updating this property will have no effect if the SE does not have CP Assist for Cryptographic functions (CPACF) enabled. 28. On a Get Request, this property is returned only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.			

List Image Activation Profiles

The List Image Activation Profiles operation lists the Image Activation Profiles for the associated CPC object.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/image-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
image-activation-profiles	Array of image-actprof-info objects	Array of nested objects (described in the following table).

Each image-actprof-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Image Activation Profile object.
name	String	The name of the Image Activation Profile.

Description

This operation lists the Image Activation Profiles associated with a particular CPC.

If the **name** query parameter is specified, the returned list is limited to those Image Activation Profiles that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 915](#).

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to the same named logical partition of that CPC, for each object to be included in the response.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 915.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 507. List Image Activation Profiles: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 506
{
  "image-activation-profiles": [
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
      APIVM1",
      "name": "APIVM1"
    },
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
      DEFAULT",
      "name": "DEFAULT"
    },
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
      ZOS1",
      "name": "ZOS1"
    }
  ]
}

```

Figure 508. List Image Activation Profiles: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI via the List CPC Objects operation. Instead, such a user may use either the List Permitted Logical Partitions operation or the Get Inventory operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Get Image Activation Profile Properties

The Get Image Activation Profile Properties operation retrieves the properties of a single Image Activation Profile designated by *{image-activation-profile-name}*.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}
```

URI variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{image-activation-profile-name}</i>	Image Activation Profile name

Query parameters:

Name	Type	Rqd/Opt	Description
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are true and false . The default is false .

Response body contents

On successful completion, the response body provides the current values of the properties for the Image Activation Profile as defined in the [“Data model”](#) on page 900.

Description

The URI path must designate an existing Image Activation Profile and the API user must have object-access permission to the associated CPC object. If either of these conditions is not met, HTTP status code 404 (Not Found) is returned.

Some of this object's property values are periodically fetched from the Support Element and cached on the HMC for quick access by the APIs. Due to the nature of this caching support, the HMC's cached value of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Image Activation Profile object.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to the same named Logical Partition of that CPC.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 918.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
	260	The activation profile name in the URI (<i>{image-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles/ZOS HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 509. Get Image Activation Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 3411
{
  "basic-cpu-counter-authorization-control": false,
  "basic-cpu-sampling-authorization-control": false,
  "boot-record-lba": "0",
  "central-storage": 4096,
  "class": "image-activation-profile",
  "clock-type": "standard",
  "coprocessor-cpu-counter-authorization-control": false,
  "cross-partition-authority-authorization-control": false,
  "crypto-activity-cpu-counter-authorization-control": false,
  "defined-capacity": 0,
  "description": "This is the ZOS Image profile.",
  "disk-partition-id": 0,
  "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles/ZOS",
  "expanded-storage": 0,
  "extended-cpu-counter-authorization-control": false,
  "global-performance-data-authorization-control": true,
  "group-profile-uri": null,
  "initial-aap-processing-weight": 0,
  "initial-aap-processing-weight-capped": false,
  "initial-cbp-processing-weight": 0,
  "initial-cbp-processing-weight-capped": false,
  "initial-ifl-processing-weight": 0,
  "initial-ifl-processing-weight-capped": false,
  "initial-internal-cf-processing-weight": 0,
  "initial-internal-cf-processing-weight-capped": false,
  "initial-processing-weight": 44,
  "initial-processing-weight-capped": false,
  "initial-vfm-storage": 64,
  "initial-ziip-processing-weight": 0,
  "initial-ziip-processing-weight-capped": false,
  "io-configuration-authorization-control": true,
  "ipl-address": "00000",
  "ipl-parameter": " ",
  "ipl-type": "ipltype-standard",
  "liccc-validation-enabled": true,
  "load-at-activation": false,
  "logical-partition-isolation-control": false,
  "logical-unit-number": "0",
  "maximum-aap-processing-weight": 0,
  "maximum-cbp-processing-weight": 0,
  "maximum-ifl-processing-weight": 0,
  "maximum-internal-cf-processing-weight": 0,
  "maximum-processing-weight": 44,
  "maximum-vfm-storage": 512,
  "absolute-aap-capping": {"type": "processors", "value": 67.95},
  "absolute-cbp-capping": {"type": "none"},
  "absolute-icf-capping": {"type": "none"},
  "absolute-ifl-capping": {"type": "none"},
  "absolute-general-purpose-capping": {"type": "processors", "value": 3.03},
  "absolute-ziip-capping": {"type": "processors", "value": 95.95},
}
```

Figure 510. Get Image Activation Profile Properties: Response (Part 1)

```

"maximum-ziip-processing-weight": 0,
"minimum-aap-processing-weight": 0,
"minimum-cbp-processing-weight": 0,
"minimum-ifl-processing-weight": 0,
"minimum-internal-cf-processing-weight": 0,
"minimum-processing-weight": 44,
"minimum-ziip-processing-weight": 0,
"name": "ZOS",
"number-dedicated-aap-processors": null,
"number-dedicated-cbp-processors": null,
"number-dedicated-general-purpose-processors": null,
"number-dedicated-icf-processors": null,
"number-dedicated-ifl-processors": null,
"number-dedicated-ziip-processors": null,
"number-reserved-dedicated-aap-processors": null,
"number-reserved-dedicated-cbp-processors": null,
"number-reserved-dedicated-general-purpose-processors": null,
"number-reserved-dedicated-icf-processors": null,
"number-reserved-dedicated-ifl-processors": null,
"number-reserved-dedicated-ziip-processors": null,
"number-reserved-shared-aap-processors": 0,
"number-reserved-shared-cbp-processors": 0,
"number-reserved-shared-general-purpose-processors": 0,
"number-reserved-shared-icf-processors": 0,
"number-reserved-shared-ifl-processors": 0,
"number-reserved-shared-ziip-processors": 0,
"number-shared-aap-processors": 0,
"number-shared-cbp-processors": 0,
"number-shared-general-purpose-processors": 1,
"number-shared-icf-processors": 0,
"number-shared-ifl-processors": 0,
"number-shared-ziip-processors": 0,
"operating-mode": "esa390",
"os-specific-load-parameters": " ",
"parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
"permit-aes-key-import-functions": true,
"permit-des-key-import-functions": true,
"permit-ecc-key-import-functions": true,
"problem-state-cpu-counter-authorization-control": false,
"processor-usage": "shared",
"reserved-central-storage": 0,
"reserved-expanded-storage": 0,
"secure-boot": true,
"time-offset-days": 0,
"time-offset-hours": 0,
"time-offset-increase-decrease": "decrease",
"time-offset-minutes": 0,
"workload-manager-enabled": false,
"worldwide-port-name": "0"
}

```

Figure 511. Get Image Activation Profile Properties: Response (Part 2)

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI via the List CPC Objects operation. Instead, such a user may use either the List Permitted Logical Partitions operation or the Get Inventory operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Update Image Activation Profile Properties

The Update Image Activation Profile Properties operation updates one or more writable properties of the Image Activation Profile designated by *{image-activation-profile-name}*.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}
```

URI variables:

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{image-activation-profile-name}</i>	Image Activation Profile name

Request body contents

The request body is expected to contain one or more field names representing writable Image Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Image Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Image Activation Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Customize/Delete Activation Profiles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
	306	One or more of the provided update values would result in an illegal state involving the operating-mode property.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	260	The activation profile name in the URI (<i>{image-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Inventory service data

Information about image activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Image Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular image activation profile is included only if the API user has access permission to that object as described in the [Get Image Activation Profile Properties](#) operation.

For each Image Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Image Activation Profile Properties”](#) on page 917. That is, the data provided is the same as would be provided if a [Get Image Activation Profile Properties](#) operation were requested targeting this object.

Load activation profile

A Load activation profile is used to load a previously activated logical partition with a control program or operating system.

An activation profile can only be created or deleted from the Hardware Management Console or the Support Element.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

This element includes the following properties.

Table 383. Load activation profile: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Load Activation Profile object, of the form <code>/api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}</code> where <code>{load-activation-profile-name}</code> is the value of the name property (Load Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object
class	—	String	The class of a Load Activation Profile object is "load-activation-profile" .
name	—	String (1-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by <code>{cpc-id}</code> .
description	(w)	String (1-50)	The load profile description
ipl-address	(w)	String (0-5)	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value will be right justified and padded with zeros to 5 characters. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
ipl-parameter¹	(w)	String (0-8)	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank, and period. Three additional characters, (@, \$, #) are also allowed when the se-version property of the associated CPC is "2.14.0" or later. On an Update, a non-empty string is left justified and right padded with blanks to 8 characters.
ipl-type²	(w)	String Enum	One of: "ipltype-standard" - Associated image activation profile is used to perform a standard load. "ipltype-scsi" - Associated image activation profile is used to perform a SCSI load. "ipltype-scsidump" - Associated image activation profile is used to perform a SCSI dump.

Table 383. Load activation profile: properties (continued)

Name	Qualifier	Type	Description
secure-boot ³	(w)	Boolean	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. This property is only applied to ipl-type of ipltype-scsi or ipltype-scsidump.
worldwide-port-name ¹	(w)	String (1-16)	Worldwide port name value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal.
disk-partition-id	(w)	Integer (0-30)	Disk partition number (also called the boot program selector) for the activation profile, used for SCSI load and SCSI dump.
logical-unit-number ¹	(w)	String (1-16)	Logical unit number value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal.
boot-record-lba ¹	(w)	String (1-16)	Boot record logical block address for the activation profile, used for SCSI load and SCSI dump, in hexadecimal.
os-specific-load-parameters	(w)	String (0-256)	Operating system-specific load parameters for the activation profile, used for SCSI load and SCSI dump.
clear-indicator ²	(w)	Boolean	Whether memory should be cleared before performing the Load (true) or not cleared (false). This property cannot be set to false when the ipl-type property is " ipltype-scsi " and the se-version property of the associated CPC is " 2.14.0 " or earlier, or the ipl-type property is " ipltype-scsidump ".
store-status-indicator ²	(w)	Boolean	When ipl-type is "ipltype-standard", this property indicates whether the store status function should be invoked before performing the Load (true) or not (false). The store status function stores the current values of the processing unit timer, the clock comparator, the program status word, and the contents of the processor registers in their assigned absolute storage locations. This property cannot be set to true when clear-indicator is true. When ipl-type is "ipltype-scsi", this property has no effect, and it cannot be set to true due to design constraints. When ipl-type is "ipltype-scsidump", this property has no effect, and it cannot be set to false due to design constraints.

Table 383. Load activation profile: properties (continued)

Name	Qualifier	Type	Description
1. An Update request accepts any mixture of [a-f, A-F, 0-9], however the original string value is not saved and a subsequent Get request may not return the exact same set of lower/upper case letters. 2. If the clear-indicator or store-status-indicator properties are not included in the request body of an Update Load Activation Profile Properties operation, certain conditions will cause their values to be set by the system as follows: - When ipl-type is "ipltype-standard" and clear-indicator is true, the store-status-indicator will be set to false. - When ipl-type is "ipltype-scsi", the store-status-indicator will be set to false. When the se-version property of the associated CPC is "2.14.0" or earlier, the clear-indicator will be set to true. - When ipl-type is "ipltype-scsidump", the clear-indicator will be set to true and the store-status-indicator will be set to true. - When ipl-type is being changed from "ipltype-scsidump" to "ipltype-standard", the store-status-indicator will be set to false. It is recommended that an Update Load Activation Profile Properties operation that includes ipl-type, clear-indicator or store-status-indicator should include all three of those properties to ensure that the desired values are in effect. 3. On a Get request, this property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.			

List Load Activation Profiles

The List Load Activation Profiles operation lists the Load Activation Profiles for the associated CPC object.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/load-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
load-activation-profiles	Array of load-actprof-info objects	Array of nested objects (described in the next table).

Each load-actprof-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Load Activation Profile object.
name	String	The name of the Load Activation Profile.

Description

This operation lists the Load Activation Profiles for the associated CPC object.

If the name query parameter is specified, the returned list is limited to those Load Activation Profiles that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 925](#).

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to at least one Logical Partition of that CPC.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 925](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 512. List Load Activation Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json;charset=UTF-8
content-length: 363
{
  "load-activation-profiles": [
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-
        profiles/DEFAULTLOAD",
      "name": "DEFAULTLOAD"
    },
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-
        profiles/MODIFYL",
      "name": "MODIFYL"
    }
  ]
}
```

Figure 513. List Load Activation Profiles: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI via the List CPC Objects operation. Instead, such a user may use either the List Permitted Logical Partitions operation or the Get Inventory operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Get Load Activation Profile Properties

The Get Load Activation Profile Properties operation retrieves the properties of a single Load Activation Profile designated by *{load-activation-profile-name}*.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}
```

URI variables

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{load-activation-profile-name}</i>	Load Activation Profile name.

Query parameters:

Name	Type	Rqd/Opt	Description
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are true and false . The default is false .

Response body contents

On successful completion, the response body provides the current values of the properties for the Load Activation Profile as defined in the [“Data model”](#) on page 923.

Description

The URI path must designate an existing Load Activation Profile and the API user must have object-access permission to the associated CPC object or at least one of its logical partitions. If either of these conditions is not met, status code 404 (Not Found) is returned.

Some of this object's property values are periodically fetched from the Support Element and cached on the HMC for quick access by the APIs. Due to the nature of this caching support, the HMC's cached value of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for Load Activation Profiles.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to at least one Logical Partition of that CPC.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents”](#) on page 928.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
	260	The activation profile name in the URI (<i>{load-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles/DEFAULTLOAD HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 514. Get Load Activation Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json; charset=UTF-8
content-length: 812
{
  "boot-record-lba": "abcdef0123456789",
  "class": "load-activation-profile",
  "clear-indicator": true,
  "description": "This is the default Load profile.",
  "disk-partition-id": 0,
  "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles/DEFAULTLOAD",
  "ipl-address": "00D00",
  "ipl-parameter": "",
  "ipl-type": "ipltype-scsi",
  "logical-unit-number": "0",
  "name": "DEFAULTLOAD",
  "os-specific-load-parameters": "",
  "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
  "secure-boot": false,
  "store-status-indicator": false,
  "worldwide-port-name": "0"
}
```

Figure 515. Get Load Activation Profile Properties: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI via the List CPC Objects operation. Instead, such a user may use either the List Permitted Logical Partitions operation or the Get Inventory operation to obtain the properties of a Logical Partition object to which

the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Update Load Activation Profile Properties

The Update Load Activation Profile Properties operation updates one or more writable properties of the Load Activation Profile designated by *{load-activation-profile-name}*.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}
```

URI variables

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{load-activation-profile-name}</i>	Load Activation Profile name.

Request body contents

The request body is expected to contain one or more field names representing writable Load Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Load Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Load Activation Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Customize/Delete Activation Profiles** task

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (<i>{load-activation-profile-name}</i>) does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Inventory service data

Information about load activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Load Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular load activation profile is included only if the API user has access permission to that object as described in the Get Load Activation Profile Properties operation.

For each Load Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Load Activation Profile Properties” on page 927](#). That is, the data provided is the same as would be provided if a Get Load Activation Profile Properties operation were requested targeting this object.

Group profile

A Group profile is used to define the group capacity value for all logical partitions belonging to that group.

A logical partition becomes a member of a group profile by placing the group profile's URI in the image activation profile used to activate the logical partition.

A group profile can only be created or deleted from the Hardware Management Console or the Support Element.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

The Group Profile object's data model includes effective properties, denoted by the (e) qualifier. Those properties are applicable when at least one of the member logical partitions is currently activated; the **effective-properties-apply** property has a value of **true** when that is the case. Changing the value of an effective property updates the corresponding characteristic of currently active logical partitions in the group, and the updated value applies to subsequently activated logical partitions in the group. Once the last active logical partition in the group has been deactivated, all changes to effective properties will be discarded and the effective properties return to their base value.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics” on page 68](#).

This element includes the following properties.

Table 384. Group profile: properties			
Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the group profile object, of the form <code>/api/cpcs/{cpc-id}/group-profiles/{group-profile-name}</code> where <code>{group-profile-name}</code> is the value of the name property (group profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object
class	—	String	The class of a Group Profile object is "group-profile" .
name	—	String (1-16)	The group profile name, which uniquely identifies this profile within the set of group profiles for the CPC object designated by <code>{cpc-id}</code>
description	(w)	String (1-50)	The group profile description
capacity	(w)	Integer	The upper bound, in MSUs, beyond which the rolling 4-hour average CPU utilization cannot exceed for the group. A value of 0 indicates the setting is unused.
effective-capacity	(e)(w)	Integer	The effective value of the capacity property.
absolute-icf-capping¹	(w)	absolute-capping-object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.
effective-absolute-icf-capping¹	(e)(w)	absolute-capping-object	The effective value of the absolute-icf-capping property.
absolute-ifl-capping¹	(w)	absolute-capping-object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
effective-absolute-ifl-capping¹	(e)(w)	absolute-capping-object	The effective value of the absolute-ifl-capping property.

Table 384. Group profile: properties (continued)

Name	Qualifier	Type	Description
absolute-general-purpose-capping¹	(w)	absolute-capping-object	The amount of absolute capping applied to the general purpose processor.
effective-absolute-general-purpose-capping¹	(e)(w)	absolute-capping-object	The effective value of the absolute-general-purpose-capping property.
absolute-cbp-capping²	(w)	absolute-capping-object	The amount of absolute capping applied to the Container Based Processor (CBP).
effective-absolute-cbp-capping²	(e)(w)	absolute-capping-object	The effective value of the absolute-cbp-capping property.
absolute-ziip-capping¹	(w)	absolute-capping-object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor.
effective-absolute-ziip-capping¹	(e)(w)	absolute-capping-object	The effective value of the absolute-ziip-capping property.
effective-properties-apply	—	Boolean	Indicates whether the object is currently in a state in which effective properties are applicable (true). Otherwise, the value is false .
Notes: 1. This property is only provided when the associated SE is at version 2.13.1 or later. 2. On a Get request, this property is returned only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the associated SE version is 2.14.0 with the suitable MCL bundle, or a later SE version.			

List Group Profiles

The List Group Profiles operation lists the Group Profiles for the associated CPC object.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/group-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
group-profiles	Array of group-actprof-info objects	Array of nested objects (described in the next table).

Each group-actprof-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Group Profile object.
name	String	The name of the Group Profile.

Description

This operation lists the Group Profiles for the associated CPC object.

If the name query parameter is specified, the returned list is limited to those Group Profiles that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 934](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

HTTP error status code	Reason code	Description
500 (Server Error)	281	An unexpected error occurred during the collection of the list of group profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb61cl538wuyebdyzu4
```

Figure 516. List Group Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json;charset=UTF-8
content-length: 182
{
  "group-profiles": [
    {
      "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/
        DEFAULT",
      "name": "DEFAULT"
    }
  ]
}
```

Figure 517. List Group Profiles: Response

Get Group Profile Properties

The Get Group Profile Properties operation retrieves the properties of a single Group Profile designated by *{group-profile-name}*.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}
```

URI variables

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{group-profile-name}</i>	Group Profile name.

Response body contents

On successful completion, the response body provides the current values of the properties for the Group Profile as defined in the [“Data model”](#) on page 932.

Description

The URI path must designate an existing Group Profile and the API user must have object-access permission to the associated CPC object. If either of these conditions is not met, HTTP status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Group Profile.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 935](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The group profile name in the URI (<i>{group-profile-name}</i>) does not designate an existing group profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/DEFAULT HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lc1538wuyebdyzu4
```

Figure 518. Get Group Profile Properties: Request

```

200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:20 GMT
content-type: application/json; charset=UTF-8
content-length: 250
{
  "absolute-cbp-capping": {"type": "none"},
  "absolute-icf-capping": {"type": "none"},
  "absolute-ifl-capping": {"value": 98.23, "type": "processors"},
  "absolute-general-purpose-capping": {"value": 1.05, "type": "processors"},
  "absolute-ziip-capping": {"type": "none"},
  "capacity": 0,
  "class": "group-profile",
  "description": "This is the default Group profile.",
  "effective-absolute-cbp-capping": {"type": "none"},
  "effective-absolute-icf-capping": {"value": 55.86, "type": "processors"},
  "effective-absolute-ifl-capping": {"value": 98.23, "type": "processors"},
  "effective-absolute-general-purpose-capping": {"type": "none"},
  "effective-absolute-ziip-capping": {"type": "none"},
  "effective-capacity": 1,
  "effective-properties-apply": true,
  "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/DEFAULT",
  "name": "DEFAULT",
  "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340"
}

```

Figure 519. Get Group Profile Properties: Response

Update Group Profile Properties

The Update Group Profile Properties operation updates one or more writable properties of the Group Profile object designated by *{group-profile-name}*.

HTTP method and URI

POST /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}

URI variables

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{group-profile-name}</i>	Group Profile name.

Request body contents

The request body is expected to contain one or more field names representing writable Group Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Group Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Group Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Customize/Delete Activation Profiles** task

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The group profile name in the URI (<i>{group-profile-name}</i>) does not designate an existing group profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	9	The operation cannot be completed because it is attempting to update an effective property when the object is not in a state in which effective properties are applicable. More specifically, the request body contains one or more fields which correspond to a property marked with the (e) qualifier in the data model, and the object's effective-properties-apply property value is false .
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Inventory service data

Information about group profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Group Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class "**cpc**" are to be included. An entry for a particular group profile is included

only if the API user has access permission to that object as described in the [Get Group Profile Properties](#) operation.

For each Group Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Group Profile Properties”](#) on page 935. That is, the data provided is the same as would be provided if a [Get Group Profile Properties](#) operation were requested targeting this object.

Capacity records

A capacity record represents a temporary upgrade that can be applied to a CPC.

These upgrades are provided through the following offerings:

- **On/Off Capacity on Demand (On/Off CoD)** - This offering allows you to temporarily add additional capacity or specialty engines due to seasonal activities, period-end requirements, peaks in workload, or application testing.
- **Capacity Backup (CBU)** - This offering allows you to replace model capacity or specialty engines to a backup server in the event of an unforeseen loss of server capacity because of an emergency.
- **Capacity for Planned Events (CPE)** - This offering allows you to replace model capacity or specialty engines due to a relocation of workload during system migrations or a data center move.

Data model

For definitions of the qualifier abbreviations in the following tables, see [“Property characteristics”](#) on page 68.

This element includes the following class-specific properties.

Table 385. Capacity records: class-specific properties		
Name	Type	Description
element-uri	String/ URI	The canonical URI path of the capacity record object, of the form <code>/api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}</code> where <code>{cpc-id}</code> is the value of the object-id property of the CPC object and <code>{capacity-record-id}</code> is the value of the record-identifier property of the Capacity Record object.
parent	String/ URI	The canonical URI path for the associated CPC object
class	String	The class of a capacity record object is "capacity-record" .
record-identifier	String (1-8)	The identifier for the capacity record.
record-type	String Enum	The type of capacity record. One of: • "unknown" - the record does not specify a record-type • "cbu" - a Capacity Backup Upgrade record • "oocod" - an On/Off Capacity on Demand record • "planned-event" - a Capacity for Planned Events record • "loaner" - resources loaned to the installation • "boost" - System Recovery Boost record

Table 385. Capacity records: class-specific properties (continued)

Name	Type	Description
activation-status	String Enum	An indication if any of the resources defined for the record are currently activated. One of: • "unknown" - the activation status of the record is not known • "not-activated" - the record is not currently active • "real" - the record is either active or pending activation, via an Add Temporary Capacity operation with a test=false option • "test" - the record is either active or pending activation via an Add Temporary Capacity operation with a test=true option • "can-be-activated" - the record is available for activation, but not currently active.
activation-date	Timestamp	Defines the time stamp for when additional capacity for the record was activated.
record-expiration-date	Timestamp	Defines the time stamp for when the capacity record will expire.
activation-expiration-date	Timestamp	Defines the time stamp for when the additional capacity activated for the record will expire and no longer be active.
maximum-real-days	Integer	Defines the maximum days that real additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
maximum-test-days	Integer	Defines the maximum days that test additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-real-days	Integer	Defines the remaining number of days that additional real capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-test-days	Integer	Defines the remaining number of days that additional test capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
maximum-real-hours	Integer	Defines the maximum hours that real additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
maximum-test-hours	Integer	Defines the maximum hours that test additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-real-hours	Integer	Defines the remaining number of hours that additional real capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-test-hours	Integer	Defines the remaining number of hours that additional test capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-number-of-real-activations	Integer	Defines the number of times that real additional capacity can be activated for the record. A value of -1 indicates that activation count is unlimited.

Table 385. Capacity records: class-specific properties (continued)

Name	Type	Description
remaining-number-of-test-activations	Integer	Defines the number of times that test additional capacity can be activated for the record. A value of -1 indicates that activation count is unlimited.
processor-info	Array of caprec-proc-info objects	A nested object describing the processor capacities available with this capacity record.
available-targets	Array of caprec-target objects	A nested object describing the set of possible activation and deactivation targets contained within this capacity record. One of these targets is chosen via the software-model request body field on the Add Temporary Capacity or Remove Temporary Capacity operations.

Table 386. caprec-proc-info object

Name	Type	Description
type	String Enum	Identifies the type of specialty processor represented. One of: • "cp" - central processor • "aap" - Application Assist Processor • "ifl" - Integrated Facility for Linux processor • "icf" - Internal Coupling Facility processor • "iip" - z Integrated Information Processors processor • "sap" - System Assist Processor • "cbp" - Container Based Processor.
processor-step	Integer	The number of processors steps available
speed-step	Integer	The CP processor speed activation step. A null object is returned for all other processor types.
max-number-processors	Integer	The maximum number of processors available for this processor type. A value of -1 indicates an unlimited number of processors.
remaining-processor-days	Integer	The remaining processor days for this processor type. A value of -1 indicates an unlimited number of days.
remaining-msu-days	Integer	The remaining MSU days for this processor type. A value of -1 indicates an unlimited number of days. A null object is returned for processor types where this field is not meaningful.

Table 387. caprec-target object

Name	Type	Description
processor-step	Integer	The CPU processor step. This is the incremental delta CPUs compared to the current activation level. The returned value may be negative.
speed-step	Integer	The CPU processor speed activation step. This is the incremental delta speed steps compared to current activation level. The returned value may be negative.

Table 387. caprec-target object (continued)

Name	Type	Description
software-model	String (1-3)	The software model that this target represents
billable-msu-cost	Integer	The overall billable MSU cost for this target
billable-msu-delta	Integer	The change in billable MSU cost by activating this target. The value may be negative.

List Capacity Records

The List Capacity Records operation lists the capacity record for a given CPC that are managed by this HMC.

HTTP method and URI

GET /api/cpcs/{cpc-id}/capacity-records

In this request, the URI variable {cpc-id} is the object ID of the target CPC object.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
capacity-record	Array of objects	Array of nested objects (described in the next table).

Each nested object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Capacity Record object.
record-identifier	String	The record identifier of the Capacity Record

Description

This operation lists the capacity record for a given CPC that are managed by this HMC.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by {cpc-id}.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 942](#).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Get Capacity Record Properties

The `Get Capacity Record Properties` operation retrieves the properties of a single Capacity Record designated by *{capacity-record-id}* from the CPC object designated by *{cpc-id}*.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}
```

URI variables

Variable	Description
<i>{cpc-id}</i>	Object ID of the target CPC object.
<i>{capacity-record-id}</i>	Capacity Record identifier, returned by a previous List Capacity Records operation

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body provides the current values of the properties for the Capacity Record as defined in [“Data model”](#) on page 939.

Description

The URI path must designate an existing Capacity Record and the API user must have object-access permission to the associated CPC object. If either of these conditions is not met, status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Capacity Record object.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	276	The capacity record has expired, it can be deleted from the SE.
	302	The capacity record identifier in the URI (<i>{capacity-record-id}</i>) must be 1 to 8 characters.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The capacity record identifier in the URI (<i>{capacity-record-id}</i>) does not designate an existing capacity record.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Inventory service data

Information about capacity records can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Capacity Record objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly via a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular capacity record is included only if the API user has access permission to that object as described in the Get Capacity Record Properties operation.

For each Capacity Record object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for [“Get Capacity Record Properties” on page 943](#). That is, the data provided is the same as would be provided if a Get Capacity Record Properties operation were requested targeting this object.

Chapter 12. Energy management

Energy Management is a management task that is pervasive and spread across several components in the systems management stack. Each layer in the stack needs to implement two key functions:

- A set of management functions appropriate for this level at the stack. Energy management functions provided by lower layers can be used to implement these functions.
- Management interfaces are provided that allows management layers above to configure and control the energy management functions.

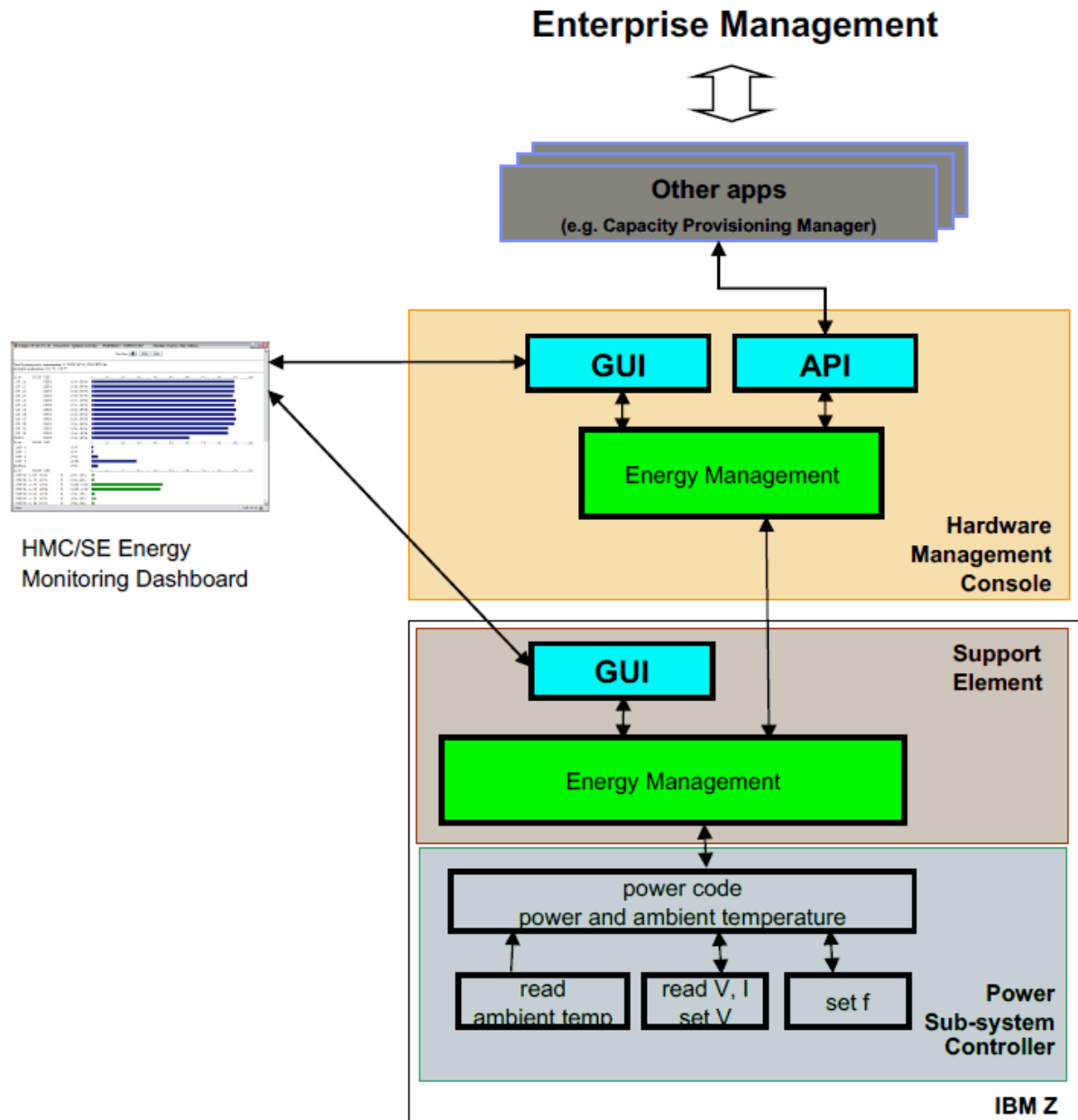


Figure 520. Energy management as applied throughout layers of enterprise management

To achieve this several pieces are needed:

Power and thermal monitoring

"You can't improve what you don't measure" is a trivial engineering paradigm. Measuring energy consumption and the thermal environment is key for management. Energy Monitoring for Z was initially introduced with z9®.

Energy control

Based on the measurement data - either for an individual system or aggregated for a group of servers or even a complete data center - analytics can be implemented. These can keep a watch on given limits or can identify optimizing potentials. At a system level energy control mechanisms will be provided to allow for changing energy consumption of a system. These energy controls can be categorized into two groups:

- **Power saving** - Power saving mechanisms are used to reduce the average energy consumption of a system. Through powering off components or reducing performance the power saving is typically achieved. For older servers, such as the zEnterprise 196 the Static Power Savings Mode is implemented that reduces processor frequency and voltage for power saving purposes. Power saving capabilities are not supported on the IBM z15™.
- **Power capping** - Power capping is a means to limit peak power consumption of a system. This is especially important in constrained data center environments. Today power and cooling allocation in data centers is usually done via the label power. This typically leads to a significant over-provisioning. Through power capping the power allocation for a system can be adjusted better to the real power consumption of a system and therefore more servers can be deployed within the same physical limits of their data center. Power capping is not supported on the IBM z14®, or the IBM z15.

Groups

A group is composed of an object that contains groups or another object and the object or objects it contains. For example, [Figure 521 on page 947](#) represents a CPC that contains a zCPC.

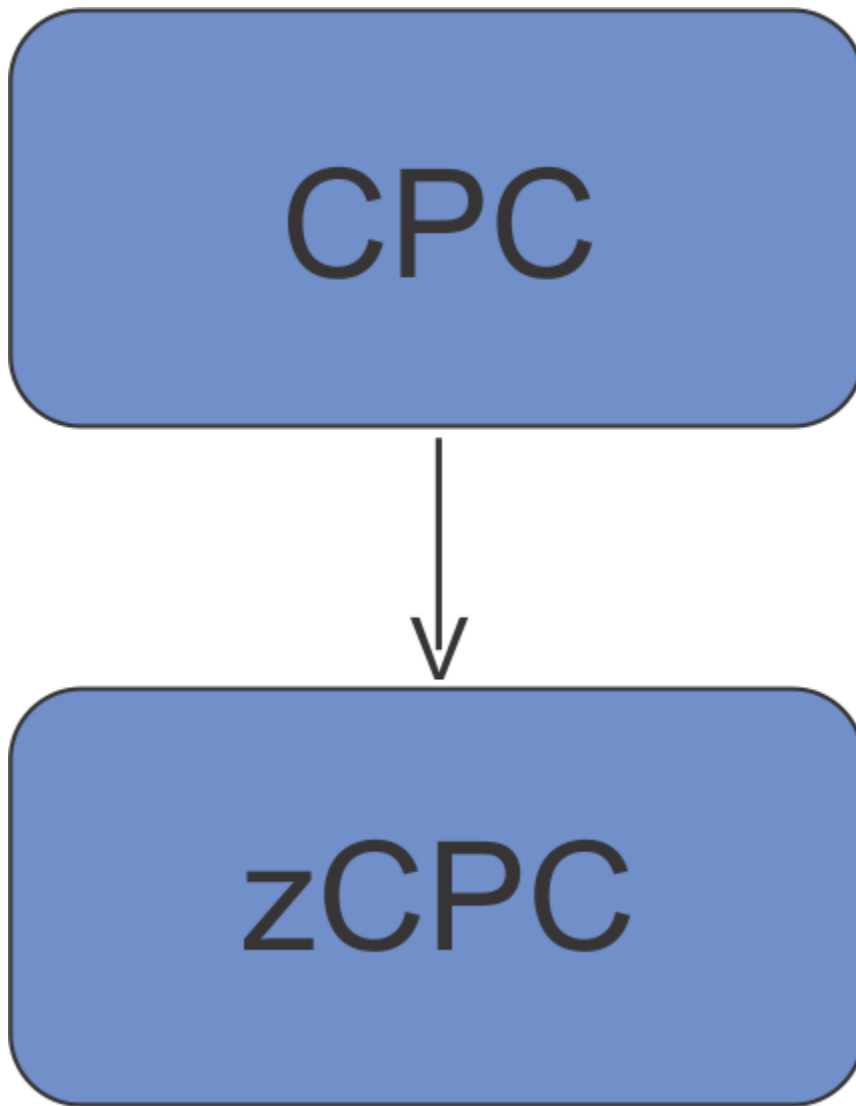


Figure 521. Example of a CPC group that contains a zCPC

Special states

In this chapter, the following states are used but the reasoning behind the states isn't always clear. So they are explained here in more detail:

"custom"

Occurs only on groups and indicates that the group does not control the children. Clients are able to alter the children of a group individually.

"under-group-control"

Occurs only on children and indicates that a group controls the state. When clients want to alter the state, the group must be set to **"custom"** first.

"not-supported"

Indicates that the feature (either power saving or power capping) is currently not supported, possible reasons can be:

- Hardware does not support it → permanent
- Firmware level does not support it → can change after a firmware update
- The hardware is not powered on → can change after the device is powered on.

"not-available"

Couldn't read the state of the underlying hardware.

"not-entitled"

Indicates that the automate feature is not installed and so power saving and power capping is not allowed.

Power saving

Power saving is a function that reduces the energy consumption of a system. Please note that power saving is only available if the Automate management enablement feature is installed. Power saving capabilities are not supported on the IBM z15. The possible settings include:

High performance

The power consumption and performance of the object are not reduced. This is the default setting.

Low power

The performance of the object is reduced to allow for low power consumption. When this setting is selected for CPC objects, all components of the object enabled for power saving have reduced performance to allow for low power consumption. Use this setting to enable group power saving.

Note: You can only set the power saving setting of the zCPC to Low power one time per calendar day in an air cooled system. This power save property is set to Not Supported if the current zCPC power saving setting is High performance but the zCPC has already entered Low power once within the calendar day.

Custom

Use Custom to disable group power saving and individually configure the components of the object for power saving.

Note: This setting is available only for CPC objects.

Group power saving

The following are important concepts regarding group power saving:

- Group power saving settings replace individual object settings--that is, the Power Saving setting of a CPC supersedes the Power Saving setting of any object contained within the CPC.
- You can enable group power saving by setting the Power Saving setting of the CPC to Low power or High performance.
- You can change individual Power Saving settings only if the object is not under group power saving control.
- To disable group power saving without changing the individual Power Saving settings of the group members, change the Power Saving setting of the CPC to Custom.

Power capping

Please note that power capping is only available if the Automate management enablement feature is installed. Power capping is not supported on the IBM z14, or the IBM z15.

Group capping

The following are important concepts regarding group power capping:

- Group caps replace individual object caps—that is, the Cap Value of a CPC supersedes the power cap of any object contained within the CPC.
- You can enable group capping by setting the Power Capping setting of the CPC to Enabled.
- You can change individual Cap Values if the object is not under group capping control.

- If a CPC contains an object that does not support power capping, the Power Rating is used in calculating the minimum power cap value for the group. The Power Rating can be found on the details window for an object.
- The maximum Cap Value for a group is the sum of the Power Rating of all Group objects.
- When a group component is powered off or removed, the group cap is redistributed to the remaining group components.
- To disable group capping without changing the individual power caps of the group members, change the Power Capping setting of the CPC to Custom.

Energy management operations summary

The following tables provide an overview of the operations provided. All POST operation are executed asynchronously and provide a job URI which can be used to obtain the current status of the asynchronous portion of the operation.

Note: The zCPC is not modeled as a full entity like a CPC, because energy management needs the zCPC to represent only Z hardware without hardware extensions. That is the reason why all zCPC related operations are tied to the CPC.

Table 388. Energy management: operations summary	
Operation name	HTTP method and URI path
“Set CPC Power Save” on page 950	POST /api/cpcs/{cpc-id}/operations/set-cpc-power-save
“Set CPC Power Capping” on page 952	POST /api/cpcs/{cpc-id}/operations/set-cpc-power-capping
“Get CPC Energy Management Data” on page 959	GET /api/cpcs/{cpc-id}/energy-management-data
“Set zCPC Power Save” on page 955	POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-save
“Set zCPC Power Capping” on page 957	POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-capping
“Get Energy Optimization Advice Summary” on page 960	GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-summary
“Get Energy Optimization Advice Details” on page 966	GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-details

Table 389. Energy management: URI variables	
Variable	Description
{cpc-id}	Object ID of a CPC

Energy Management for CPC object

The energy management for the CPC object represents all energy management for the CPC.

Data model

The data model for a CPC object includes some properties related to energy management. These properties are described in [“Energy management related additional properties” on page 753](#).

Operations

Set CPC Power Save

Use the Set CPC Power Save operation to set the power save setting of a CPC.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-cpc-power-save
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
power-saving	String Enum	Required	The possible settings are: • "high-performance" - The power consumption and performance of the CPC are not reduced. This is the default setting. • "low-power" - Low power consumption for all components of the CPC enabled for power saving. • "custom" - Components may have their own settings changed individually. No component settings are actually changed when this mode is entered.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
job-uri	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to control the average energy consumption of a CPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated CPC is under group control (see [“Group capping” on page 948](#)) or the **cpc-power-saving** property of the CPC is set to **"not-supported"** or **"not-entitled"**. (See [“Energy management related additional properties” on page 753](#) for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-saving settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the Query Job Status operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described

in “HTTP status and reason codes” on page 951. After completion, the Query Job Status operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to all CPC and zCPC objects
- Action/task permission to the **Power Save** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in “Response body contents” on page 950.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the Z hardware
	162	Communication error occurred while trying to access the Z hardware
	163	An error occurred at one or more children

If the job reason code is 163, the **job-results** field provided by the Query Job Status operation will contain an object with the following fields:

Field name	Type	Description
errors	Object array	A list of error objects, containing detailed error information about errors occurred on children
at-least-one-operation-succeed	Boolean	True indicates that the operation was successful for at least one child.

Each error object has this structure:

Job status codes	Job reason code	Description
object-uri	String URI	The canonical URI path for a specific object where the error occurred
reason-code	Integer	Specify the specific error type, possible values are: • 160 - A firmware error occurred while executing the operation • 161 - A hardware error occurred while performing the energy management operation • 162 - Communication error occurred while trying to access the hardware
message	String	A non-localized message provided for development purposes only. Client applications should not display this message directly to the user.

Set CPC Power Capping

Use the Set CPC Power Capping operation to set the power capping settings of a CPC.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/set-cpc-power-capping

In this request, the URI variable {cpc-id} is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
power-capping-state	String Enum	Required	The possible settings are: • "disabled" - The power cap of the CPC is not set and the peak power consumption is not limited. This is the default setting. • "enabled" - The peak power consumption of the CPC is limited to the current cap value. • "custom" - Individually configure the components of the Z hardware for power capping. No component settings are actually changed when this mode is entered.

Field name	Type	Rqd/Opt	Description
power-cap-current	Integer	Optional	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the CPC. This field is only required if the power-capping-state field is set to "enabled" . The power-cap-current must be between cpc-power-cap-minimum and cpc-power-cap-maximum : cpc-power-cap-minimum <= value <= cpc-power-cap-maximum

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
job-uri	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to limit the peak power consumption of a CPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated CPC is under group control (see [“Group capping”](#) on page 948) or the **cpc-power-capping-state** property of the CPC is set to **"not-supported"** or **"not-entitled"**. (See [“Energy management related additional properties”](#) on page 753 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-capping settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described in [“HTTP status and reason codes”](#) on page 953. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to all CPC and zCPC objects
- Action/task permission to the **Power Capping** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 953.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The power-cap-current field contains a value that is not in the range cpc-power-cap-minimum ... cpc-power-cap-maximum
	5	The power-cap-current field is not set, but power-capping-state field is set to "enabled" .
403 (Forbidden)	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the Z hardware
	162	Communication error occurred while trying to access the Z hardware
	163	An error occurred at one or more children

If the job reason code is 163, the **job-results** field provided by the Query Job Status operation will contain an object with the following fields:

Field name	Type	Description
errors	Object array	A list of error objects, containing detailed error information about errors occurred on children
at-least-one-operation-succeed	Boolean	True indicates that the operation was successful for at least one child.

Each error object has this structure:

Job status codes	Job reason code	Description
object-uri	String URI	The canonical URI path for a specific object where the error occurred
reason-code	Integer	Specify the specific error type, possible values are: • 160 - A firmware error occurred while executing the operation • 161 - A hardware error occurred while performing the energy management operation • 162 - Communication error occurred while trying to access the hardware
message	String	A non localized message provided for development purposes only. Client applications should not display this message directly to the user.

Set zCPC Power Save

Use the Set zCPC Power Save operation to set the power save settings of the zCPC portion of a CPC.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-save

In this request, the URI variable {cpc-id} is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
power-saving	String Enum	Required	The possible settings are: • "high-performance" - The power consumption and performance of the zCPC are not reduced. This is the default setting. • "low-power" - Low power consumption for all components of the zCPC enabled for power saving.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
job-uri	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to control the average energy consumption of a zCPC portion of the CPC {cpc-id}, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated zCPC is under group control (see [“Group capping”](#) on page 948) or the **zcpc-power-saving** property of the zCPC is set to **"not-supported"** or **"not-entitled"**. (See

“Energy management related additional properties” on page 753 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-saving settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the Query Job Status operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described in “HTTP status and reason codes” on page 956. After completion, the Query Job Status operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to all CPC and zCPC objects
- Action/task permission to the **Power Save** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in “Response body contents” on page 955.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully

Job status codes	Job reason code	Description
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the Z hardware
	162	Communication error occurred while trying to access the Z hardware

Set zCPC Power Capping

Use the Set zCPC Power Capping operation to set the power capping settings of the zCPC portion of a CPC.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-capping
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
power-capping-state	String Enum	Required	The possible settings are: "disabled" - The power cap of the zCPC is not set and the peak power consumption is not limited. This is the default setting. "enabled" - The peak power consumption of the zCPC is limited to the current cap value.
power-cap-current	Integer	Optional	Specifies the current cap value for the zCPC in watts (W). The current cap value indicates the power budget for the zCPC. This field is only required if the power-capping-state field is set to "enabled" . The power-cap-current must be between zcpc-power-cap-minimum and zcpc-power-cap-maximum : zcpc-power-cap-minimum <= value <= zcpc-power-cap-maximum

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
job-uri	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to limit the peak power consumption of a zCPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated zCPC is under group control (see [“Group capping”](#) on page 948) or the **zcpc-power-capping-state** property of the zCPC is set to **"not-supported"** or **"not-entitled"**. (See [“Energy management related additional properties”](#) on page 753 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-capping settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described below. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to all CPC and zCPC objects
- Action/task permission to the **Power Capping** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in [“Response body contents”](#) on page 957.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
	7	The power-cap-current field contains a value that is not in the range zcpc-power-cap-minimum ... zcpc-power-cap-maximum
	5	The power-cap-current field is not set, but power-capping-state field is set to "enabled" .
403 (Forbidden)	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the Z hardware
	162	Communication error occurred while trying to access the Z hardware

Get CPC Energy Management Data

Use the Get CPC Energy Management Data operation to retrieve all energy management related data in one single call.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/energy-management-data
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
objects	Array of objects	An array of nested em-data objects containing the energy management data. The format of each nested object is given in the next table.

Each nested em-data object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The canonical URI path of the specific object to which this em-data object pertains.
object-id	String	Object-id property of the specific object to which this em-data object pertains.
class	String	The type of the specific object to which this em-data object pertains.
properties	Object	Nested object containing the energy management properties for the object identified by the object-uri field, as described in the data model section for objects of the type indicated by the class field.
error-occurred	Boolean	If true, indicates that an error occurred while querying the data for the object specified by the object-uri. As a consequence the property could be null or incomplete.

Description

The Get CPC Energy Management Data is a convenience operation to allow a client to retrieve all energy management related data for a CPC in a single request rather than invoking several requests to retrieve this data.

Note that this operation returns data for a child object of the designated CPC only if the API user has object-access permission to that object. Children objects for which the API user does not have access are omitted from the response and no error is indicated.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by the request, and for any children objects for which data is to be returned.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 959](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI (<i>{cpc-id}</i>) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,” on page 43](#).

Get Energy Optimization Advice Summary

The Get Energy Optimization Advice Summary operation provides a summary of all currently available energy optimization advice for a CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-summary
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC whose energy optimization advice summary is to be returned.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
advice	Array of energy-optimization-advice-summary objects	A summary of all energy optimization advice available for the CPC. It is an array of nested energy-optimization-advice-summary objects as described in the next table.

Each nested energy-optimization-advice-summary object contains the following fields:

Field name	Type	Description
type	String Enum	Identifies the advice type. Valid values are: • "environmental" - Advice about environmental aspects of energy management such as air temperature. • "processor-utilization" - Advice related to the amount of processor utilization in the CPC.
timestamp	Timestamp	The time when this advice was created or updated. This field and the last-energy-advice-time property of the CPC are updated each time the status field is updated.
summary	String (0-153)	Human readable description of the status field.
status	String Enum	The current status of the parameters factored into this advice. The valid values for this field depend on the value of the type field. When type is "environmental" : • "optimal" - Parameters related to this advice are currently at optimal values; no changes are recommended at this time. • "above-threshold" - One or more of the parameters is above its threshold; changes are recommended. When type is "processor-utilization" : • "low-utilization" - System utilization is low. If static power save is enabled, power consumption can be reduced. • "low-utilization-power-save" - System utilization is low, but static power save is enabled. • "high-utilization" - System utilization is high and the system is configured optimally. • "high-utilization-power-save" - System utilization is high, but static power save is enabled. Performance can be increased by disabling static power save on the system.

Description

This operation returns summary information about the current energy optimization advice for the CPC object specified by *{cpc-id}*.

On successful execution, the energy optimization advice summary for each type of advice for which there is available data for the CPC is provided in the response body, and HTTP status code 200 (OK) is returned.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object identified in the request URI.
- Action/task permission to the **Energy Optimization Advisor** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 960](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

<i>Table 390. Get Energy Optimization Advice Summary: HTTP status and reason codes</i>		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing CPC, or it designates a CPC for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
POST /api/cpcs/a44d8ab8-e68e-3a07-9bff-e484a62ca00d/operations/get-energy-optimization-
advice-details HTTP/1.1
x-api-session: 2szwxczuje9cttj01145ozlab65x9osmpvvdibob5w0nvgwm80
content-type: application/json
content-length: 52
{
  "timescale": "four-hours",
  "type": "environmental"
}
```

Figure 522. Get Energy Optimization Advice Summary: Request


```

200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 19:32:27 GMT
content-type: application/json; charset=UTF-8
content-length: 2683
{
  "ambient-temperature":24.200000762939453,
  "power-saving":126,
  "status":"above-threshold",
  "summary":"The ambient temperature is above the current threshold of 23.0\u00b0C
    (73.4\u00b0F). If the ambient temperature is lowered below the identified threshold,
    the system power consumption will be reduced by approximately 126 W.",
  "temperature-graph":[
    {
      "data":23.80500030517578,
      "timestamp":1457451614605
    },
    {
      "data":23.577499389648438,
      "timestamp":1457452214606
    },
    {
      "data":23.645000457763672,
      "timestamp":1457452814605
    },
    {
      "data":24.00749969482422,
      "timestamp":1457453414605
    },
    {
      "data":23.850000381469727,
      "timestamp":1457454014605
    },
    {
      "data":23.697500228881836,
      "timestamp":1457454614605
    },
    {
      "data":23.704999923706055,
      "timestamp":1457455214605
    },
    {
      "data":23.825000762939453,
      "timestamp":1457455814605
    },
    {
      "data":23.704999923706055,
      "timestamp":1457456414605
    },
    {
      "data":23.75,
      "timestamp":1457457014605
    }
  ],
}

```

Figure 523. Get Energy Optimization Advice Summary: Response (Part 1)

```
{
  "data":23.704999923706055,
  "timestamp":1457457614605
},
{
  "data":23.7450008392334,
  "timestamp":1457458214605
},
{
  "data":23.78499984741211,
  "timestamp":1457458814606
},
{
  "data":23.834999084472656,
  "timestamp":1457459414607
},
{
  "data":23.795000076293945,
  "timestamp":1457460014606
},
{
  "data":23.912500381469727,
  "timestamp":1457460614604
},
{
  "data":23.80500030517578,
  "timestamp":1457461214606
},
{
  "data":23.6875,
  "timestamp":1457461814606
},
{
  "data":23.799999237060547,
  "timestamp":1457462414604
},
{
  "data":23.889999389648438,
  "timestamp":1457463014605
},
{
  "data":23.907499313354492,
  "timestamp":1457463614604
},
{
  "data":23.877500534057617,
  "timestamp":1457464214614
},
{
  "data":24.012500762939453,
  "timestamp":1457464814605
},
}
```

Figure 524. Get Energy Optimization Advice Summary: Response (Part 2)

```

    {
      "data":24.200000762939453,
      "timestamp":1457465414604
    }
  ],
  "temperature-threshold":23.0,
  "timestamp":1457031003347,
  "type":"environmental",
  "wattage":6952,
  "wattage-graph":[
    {
      "data":6957,
      "timestamp":1457451614605
    },
    {
      "data":6970,
      "timestamp":1457452214606
    },
    {
      "data":6968,
      "timestamp":1457452814605
    },
    {
      "data":6970,
      "timestamp":1457453414605
    },
    {
      "data":6965,
      "timestamp":1457454014605
    },
    {
      "data":6970,
      "timestamp":1457454614605
    },
    {
      "data":6965,
      "timestamp":1457455214605
    },
    {
      "data":6963,
      "timestamp":1457455814605
    },
    {
      "data":6958,
      "timestamp":1457456414605
    },
    {
      "data":6969,
      "timestamp":1457457014605
    },
    {
      "data":6963,
      "timestamp":1457457614605
    }
  ],

```

Figure 525. Get Energy Optimization Advice Summary: Response (Part 3)

```

    {
      "data":6963,
      "timestamp":1457458214605
    },
    {
      "data":6967,
      "timestamp":1457458814606
    },
    {
      "data":6961,
      "timestamp":1457459414607
    },
    {
      "data":6967,
      "timestamp":1457460014606
    },
    {
      "data":6963,
      "timestamp":1457460614604
    },
    {
      "data":6968,
      "timestamp":1457461214606
    },
    {
      "data":6969,
      "timestamp":1457461814606
    },
    {
      "data":6964,
      "timestamp":1457462414604
    },
    {
      "data":6963,
      "timestamp":1457463014605
    },
    {
      "data":6951,
      "timestamp":1457463614604
    },
    {
      "data":6941,
      "timestamp":1457464214614
    },
    {
      "data":6947,
      "timestamp":1457464814605
    },
    {
      "data":6955,
      "timestamp":1457465414604
    }
  ]
}

```

Figure 526. Get Energy Optimization Advice Summary: Response (Part 4)

Get Energy Optimization Advice Details

The Get Energy Optimization Advice Details operation returns detailed information about currently available energy optimization advice for a CPC.

HTTP method and URI

POST /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-details

In this request, the URI variable *{cpc-id}* is the object ID of the CPC whose energy optimization advice details are to be returned.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
timescale	String Enum	Optional	Timescale for the advice graph data points. This is the amount of time to be covered by the data points in the graph. Valid values are: • "four-hours" • "one-day" • "three-days" • "one-week" If not specified, the default is "one-day" .
type	String Enum	Required	The type of advice details to return. Valid values are: • "environmental" - Advice about environmental aspects of energy management such as air temperature. • "processor-utilization" - Advice related to the processor utilization in the CPC.

Response body contents

On successful completion, the response body contains a JSON object that provides the values of the specific energy optimization advice object requested from the CPC. The advice object contains a set of fields common to all types of advice, and a set of additional fields that differ based on the advice type.

The common fields provided for all types of advice are defined in the following table:

Field name	Type	Description
type	String Enum	Identifies the advice type. Valid values are: • "environmental" - Advice about environmental aspects of energy management such as air temperature. • "processor-utilization" - Advice related to the processor utilization in the CPC.
timestamp	Timestamp	The time when this advice was created or updated. This field and the last-energy-advice-time property of the CPC are updated each time the status field is updated.
summary	String (0-153)	Human readable description of the status field.

Field name	Type	Description
status	String Enum	The current status of the parameters factored into this advice. The valid values for this field depend on the value of the type field. When type is "environmental": • "optimal" - Parameters related to this advice are currently at optimal values; no changes are recommended at this time. • "above-threshold" - One or more of the parameters is above its threshold; changes are recommended. When type is "processor-utilization": • "low-utilization" - System utilization is low. If static power save is enabled, power consumption can be reduced by the amount specified in the power-saving field. • "low-utilization-power-save" - System utilization is low, but static power save is enabled. • "high-utilization" - System utilization is high and the system is configured optimally. • "high-utilization-power-save" - System utilization is high, but static power save is enabled. Performance can be increased by disabling static power save on the system.

When the **type** field contains **"environmental"**, the response body also contains the following additional type-specific fields:

Field name	Type	Description
ambient-temperature	Float	The ambient temperature when this advice was generated, in degrees Celsius.
wattage	Integer	The electrical power usage when this advice was generated, in Watts.
temperature-threshold	Float	The temperature threshold that causes an increase of the system power consumption, in degrees Celsius.
power-saving	Integer	The amount of electrical power that can be saved when following this advice, in Watts.
temperature-graph	Array of float-data-point objects	An ordered collection of temperature data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is a float-data-point object representing the ambient environment temperature in degrees Celsius at a specific point in time. The interval is implicit given by the timestamps of the individual data points.
wattage-graph	Array of integer-data-point objects	An ordered collection of electrical power data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is an integer-data-point object representing the total system power, in Watts, at a specific point in time. The interval is implicit, given by the timestamps of the individual data points.

When the **type** field contains **"processor-utilization"**, the response body also contains the following additional type-specific fields:

Field name	Type	Description
wattage	Integer	The electrical power usage when this advice was generated, in Watts.

Field name	Type	Description
power-saving	Integer	The amount of electrical power that will be saved by following this advice, in Watts.
utilization-threshold	Integer (0-100)	The threshold in percentage. The threshold is used to calculate the current system status. If system utilization is below the threshold, enabling static power save should be considered.
utilization	Array of type-utilization objects	List of all processor types installed in the CPC and their percent utilization at the time this advice was created.
static-power-save-recommended	Boolean	Indicates whether enabling static power save is recommended. True if recommended; false otherwise.
utilization-graphs	Array of type-utilization-graph objects	A collection of processor utilization graphs. Each element of this array contains the utilization graph of a processor type available in the system.

Each nested float-data-point object contains the following fields:

Field name	Type	Description
data	Float	Floating point data value.
timestamp	Timestamp	The time when this data point was taken

Each nested integer-data-point object contains the following fields.

Field name	Type	Description
data	Integer	Integer data value.
timestamp	Timestamp	The time when this data point was taken.

Each type-utilization-graph object contains the following fields:

Field name	Type	Description
type	String Enum	The processor type. Valid values are: • "cp" - Central (general purpose) processor. • "icf" - Internal Coupling Facility processor. • "ifl" - Integrated Facility for Linux processor. • "sap" - Service assist processor. • "sys" - Total system utilization. (Used for threshold monitoring).
utilization-graph	Array of integer-data-point objects	An ordered collection of utilization data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is an integer-data-point object representing the utilization of the processor type in percentage at a specific point in time. The interval is implicit, given by the timestamps of the individual data points.

Each nested type-utilization object contains the following fields:

Field name	Type	Description
type	String Enum	The processor type. Valid values are: • "cp" - Central (general purpose) processor. • "icf" - Internal Coupling Facility processor. • "ifl" - Integrated Facility for Linux processor. • "sap" - Service assist processor. • "sys" - Total system utilization. (Used for threshold monitoring).
utilization	Integer (0-100)	The current utilization in percentage for the given processor type.

Description

This operation returns detailed information about the requested type of energy optimization advice for the CPC object specified by *{cpc-id}*.

On successful execution, the energy optimization advice details for the requested type of advice for the CPC are provided in the response body, and HTTP status code 200 (OK) is returned. The details cover as much of the requested time period as is available.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object identified in the request URI.
- Action/task permission to the **Energy Optimization Advisor** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in [“Response body contents” on page 967](#).

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 391. Get Energy Optimization Advice Details: HTTP status and reason codes		
HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 50 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing CPC, or it designates a CPC for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Table 391. Get Energy Optimization Advice Details: HTTP status and reason codes (continued)

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in [Chapter 3, “Invoking API operations,”](#) on page 43.

Example HTTP interaction

```
GET /api/cpcs/a44d8ab8-e68e-3a07-9bff-e484a62ca00d/operations/get-energy-optimization-
advice-summary HTTP/1.1
x-api-session: 2z7xak1rm55fqu2o48h5btjsu5rwtexyuxrmvcn51rzbsxi6w
```

Figure 527. Get Energy Optimization Advice Details: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 19:30:21 GMT
content-type: application/json; charset=UTF-8
content-length: 511
{
  "advice": [
    {
      "status": "above-threshold",
      "summary": "The ambient temperature is above the current threshold of 23.0\u00b0C
(73.4\u00b0F). If the ambient temperature is lowered below the identified
threshold, the system power consumption will be reduced by approximately 126 W.",
      "timestamp": 1457031003347,
      "type": "environmental"
    },
    {
      "status": "high-utilization",
      "summary": "Based on processor utilization there are no system power consumption
recommendations.",
      "timestamp": 1448991133600,
      "type": "processor-utilization"
    }
  ]
}
```

Figure 528. Get Energy Optimization Advice Details: Response

Appendix A. Enum values for a type of managed objects within User Roles

The valid Enum values that can be used to specify a class of managed objects within User Role objects are listed in the following table:

Table 392. Enum values for a class of managed objects	
Enum for type name	Type
accelerator-adapter	zEnterprise Data Compression (zEDC) adapter
chpid	Channel path identifier used to map to a specific channel and device number
cloud-network-adapter	Cloud Network adapter
cp	Physical processor in the CPC
cpc	CPC defined to the console via Add Object Definition
crypto	Physical Crypto adapter providing support for cryptographic operations
crypto-adapter	Cryptographic coprocessor adapter
fc-storage-group	Fibre Connection (FICON) storage group
fcp-storage-group	FCP storage group
ficon-adapter	Fibre Connection (FICON) adapter
fid	Logical identifier associated with virtual function on a PCIe native adapter
hipersockets-adapter	Memory-based network adapter
lpar-image	Partition where an operating system is run
osa-adapter	Open Systems Adapter (OSA) network adapter
partition	Dynamic Partition Manager (DPM) partition
pattern-match-group	Group containing objects whose name matches a specified pattern
pchid	Physical channel identifier associated with channel hardware (excludes Cryptos)
roce-adapter	RDMA over Converged Ethernet adapter
storage-template	Storage resource definition template
system-manual-definition	Template used to manually define a system object to the Hardware Management Console
undefined-cpc	CPC that is discovered in the console domain but not defined to the console
user-defined-group	A group of objects defined by a user for organizing objects with a similar purpose or to assign different activation profiles

Appendix B. Enum values for the User Role object

The valid Enum values for the **name** property of User Role objects with a **type** of "**system-defined**" are listed in the following table:

<i>Table 393. Enum values for the name property of User Role objects with a type of "system-defined"</i>		
Enum for system-defined User Role name	System-defined User Role	Valid as an associated User Role
hmc-access-administrator-tasks	Access Administrator Tasks	Yes
hmc-advanced-operator-tasks	Advanced Operator Tasks	Yes
hmc-all-resources	All Resources	No
hmc-all-system-managed-objects	All System Managed Objects	No
hmc-defined-system-managed-objects	Defined System Managed Objects	No
hmc-energy-administrator-tasks	Energy Administrator Tasks	Yes
hmc-operator-tasks	Operator Tasks	Yes
hmc-service-representative-tasks	Service Representative Tasks	Yes
hmc-storage-administrator-objects	Storage Administrator Objects	No
hmc-storage-administrator-tasks	Storage Administrator Tasks	Yes
hmc-system-programmer-tasks	System Programmer Tasks	Yes

Appendix C. Enum values for the Task object

The valid Enum values for the **name** property of Task objects are listed in the following table:

Table 394. Enum values for the name property of Task objects		
Enum for task name	Task	view-only-mode supported
access-removable-media	Access Removable Media	false
activate-base	Activate	false
adapter-details	Adapter Details	false
add-object-definition	Add Object Definition	false
add-systems-to-ctn	Add Systems to CTN	false
adjust-leap-second-offset	Adjust Leap Second Offset	false
adjust-time	Adjust Time	false
adjust-time-zone-offset	Adjust Time Zone Offset	false
alternate-support-element	Alternate Support Element	false
alternate-support-element-engineering-changes	Alternate Support Element Engineering Changes (ECs)	false
analyze-console-internal-code	Analyze Console Internal Code	false
archive-security-logs-base	Archive Security Logs	false
archive-security-logs-z	Archive Security Logs	false
audit-and-log-management	Audit and Log Management	false
authorize-internal-code-changes	Authorize Internal Code Changes	false
automatic-activation	Automatic Activation	false
backup-critical-console-data	Backup Critical Console Data	false
backup-critical-data	Backup Critical Data	false
block-automatic-licensed-internal-code-change-installation	Block Automatic Licensed Internal Code Change Installation	false
certificate-management	Certificate Management	false
change-console-internal-code	Change Console Internal Code	false
change-internal-code	Change Internal Code	false
change-lpar-controls	Change LPAR Controls	false
change-lpar-group-controls	Change LPAR Group Controls	false
change-lpar-i-o-priority-queuing	Change LPAR I/O Priority Queuing	false
change-lpar-security	Change LPAR Security	false
change-object-definition	Change Object Definition	false
change-object-options	Change Object Options	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
concurrent-upgrade-engineering-changes	Concurrent Upgrade Engineering Changes (ECs)	false
configure-3270-emulators	Configure 3270 Emulators	false
configure-backup-settings	Configure Backup Settings	false
configure-channel-path-on-off	Configure Channel Path On/Off	true
configure-data-replication	Configure Data Replication	false
configure-external-time-source	Configure External Time Source	false
configure-storage-storageadmin	Configure Storage – Storage Administrator	false
configure-storage-sysprog	Configure Storage - System Programmer	false
configure-system-policies	Configure System Policies	false
connect-systems-to-key-managers	Connect Systems to Key Managers	false
console-default-user-settings	Console Default User Settings	false
console-messenger	Console Messenger	false
control-pulse-per-second-signal	Control Pulse Per Second Signal	false
copy-console-logs-to-media	Copy Console Logs to Media	false
create-certificate-signing-request	Create Certificate Signing Request	false
create-hipersockets-adapter	Create HiperSockets Adapter	false
create-self-signed-certificate	Create Self-Signed Certificate	false
create-welcome-text	Create Welcome Text	false
customer-information	Customer Information	false
customize-activity-profiles	Customize Activity Profiles	false
customize-api-settings	Customize API Settings	false
customize-automatic-logon	Customize Automatic Logon	false
customize-console-date-time	Customize Console Date/Time	false
customize-console-services	Customize Console Services	false
customize-customer-information	Customize Customer Information	false
customize-delete-activation-profiles	Customize/Delete Activation Profiles	false
customize-network-settings-base	Customize Network Settings	false
customize-outbound-connectivity	Customize Outbound Connectivity	false
customize-product-engineering-access	Customize Product Engineering Access	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
customize-remote-service	Customize Remote Service	false
customize-scheduled-operations	Customize Scheduled Operations	false
customize-scheduled-operations-c	Customize Scheduled Operations	false
customize-support-element-date-time	Customize Support Element Date/Time	false
deactivate-base	Deactivate	false
deconfigure-ctn	Deconfigure CTN	false
delete-hipersockets-adapter	Delete HiperSockets Adapter	false
delete-partition	Delete Partition	false
domain-security	Domain Security	false
dump-partition	Dump	false
edit-certificates	Edit Certificates	false
edit-frame-layout	Edit Frame Layout	false
enable-dynamic-partition-manager	Enable Dynamic Partition Manager	false
enable-ftp-access-to-mass-storage-media	Enable FTP Access to Mass Storage Media	false
enable-i-o-priority-queuing	Enable I/O Priority Queuing	false
energy-optimization-advisor	Energy Optimization Advisor	false
engineering-changes	Engineering Changes (ECs)	false
environmental-efficiency-statistics	Environmental Efficiency Statistics	false
export-certificates-directly-to-key-managers	Export Certificates Directly to Key Managers	false
export-certificates-to-email	Export to Email	false
export-certificates-to-filesystem	Export to File System	false
export-certificates-to-ftp	Export to FTP	false
export-certificates-to-key-managers	Export Certificates to Key Managers	false
export-certificates-to-usb	Export to USB	false
export-ctn-data	Export CTN data (.xls)	false
export-import-profile-data	Export/Import Profile Data (API only)	false
export-wwpns	Export WWPNS	false
fibre-channel-analyzer	Fibre Channel Analyzer	false
file-dialog	File Dialog	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
firmware-details	Firmware Details	false
format-media	Format Media	false
getting-started-with-dynamic-partition-manager	Getting Started with Dynamic Partition Manager	false
grouping	Grouping	false
hardware-messages	Hardware Messages	true
idaa-callhome-config	Configure IDAA Call-Home	false
image-details	Image Details	false
import-access-control-file	Manage Product Engineering Access Control File	false
import-key-manager-certificate	Import Key Manager Certificate	false
import-signed-certificate	Import Signed Certificate	false
input-output-configuration-save-and-restore	Input/Output (I/O) Configuration Save and Restore	false
installation-complete-report	Installation Complete Report	false
integrated-3270-console	Integrated 3270 Console	false
integrated-ascii-console	Integrated ASCII Console	false
join-existing-ctn	Join Existing CTN	false
lic-security-mode	Change Licensed Internal Code Security Mode	false
limitedlic-security-mode	Change Limited Licensed Internal Code Security Mode (SE task)	false
load	Load	false
load-from-removable-media-or-server	Load from Removable Media or Server	false
logical-processor-add	Logical Processor Add	false
manage-adapters	Manager Adapters	false
manage-coupling-facility-port-enablement	Manage Coupling Facility Port Enablement	true
manage-flash-allocation	Manage Flash allocation	false
manage-ldap-server-definitions	Manage LDAP Server Definitions	false
manage-password-rules	Manage Password Rules	false
manage-power-service-state	Manage Power Service State	false
manage-print-screen-files	Manage Print Screen Files	false
manage-processor-sharing	Manage Processor Sharing	false
manage-remote-connections	Manage Remote Connections	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
manage-remote-support-requests	Manage Remote Support Requests	false
manage-ssh-keys	Manage SSH Keys	false
manage-syslog-servers	Manage Syslog Servers	false
manage-system-time	Manage System Time	false
manage-user-patterns	Manage User Patterns	false
manage-user-roles	Manage User Roles	false
manage-user-templates	Manage User Templates	false
manage-users	Manage Users	false
manage-web-services-api-logs	Manage Web Services API Logs	false
mobile-app-preferences	HMC Mobile Settings	false
modify-assigned-server-roles	Modify Assigned Server Roles	false
monitor-system-events	Monitor System Events	false
monitors-dashboard	Monitors Dashboard	false
multi-factor-auth	Manage Multi-factor Authentication	false
network-diagnostic-information	Network Diagnostic Information	false
new-partition	New Partition	false
object-locking-settings	Object Locking Settings	false
offload-virtual-retain-data-to-removable-media	Offload Problem Analysis Data to Removable Media	false
operating-system-messages	Operating System Messages	true
osa-advanced-facilities	OSA Advanced Facilities	true
partition-details	Partition Details	false
partition-details-controls	Partition Details - Controls	false
pchid-details	Adapter Details	false
perform-console-repair-action	Perform a Console Repair Action	false
perform-model-conversion	Perform Model Conversion	false
perform-problem-analysis	Perform Problem Analysis	false
perform-transfer-rate-test	Perform Transfer Rate Test	false
product-engineering-directed-changes	Product Support Directed Changes	false
psw-restart	PSW Restart	false
reassign-channel-path-ids	Reassign Channel Path IDs	false
reassign-devices	Reassign Devices	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
reassign-hardware-management-console	Reassign Hardware Management Console	false
reassign-i-o-path	Reassign I/O Path	false
reboot-support-element	Reboot Support Element	false
rebuild-vital-product-data	Rebuild Vital Product Data	false
remote-hardware-management-console	Remote Hardware Management Console	false
remote-service	Remote Service	false
remove-key-manager-connections	Remove Key Manager Connections	false
remove-object-definition	Remove Object Definition	false
remove-systems-from-ctn	Remove Systems from CTN	false
rename-ctn	Rename CTN	false
report-problem-base	Report a Problem	false
report-problem-z	Report a Problem	false
reset-clear	Reset Clear	false
reset-normal	Reset Normal	false
restore-config	Import Dynamic Partition Manager Configuration	false
retrieve-backup-file-from-ftp	Retrieve Backup or Upgrade Data	false
retrieve-internal-code	Retrieve Internal Code	false
save-restore-customizable-console-data	Save/Restore Customizable Console Data	false
save-stp-debug-data	Save STP Debug Data	false
save-upgrade-data	Save Upgrade Data	false
save-upgrade-ftp	Save Upgrade Data	false
se-advanced-facilities	Advanced Facilities	true
se-change-lpar-cryptographic-controls	Change LPAR Cryptographic Controls	false
se-change-mirror-time	Change Mirror Time (SE task)	false
se-channel-interface-tests	Channel Interface Tests	false
se-channel-pchid-assignment	Channel PCHID Assignment	false
se-channel-problem-determination	Channel Problem Determination	false
se-check-internal-code-change-dependencies	Check Dependencies (SE task)	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
se-checkout-tests	Checkout Tests (SE task)	false
se-chpid-details	CHPID Details (SE task)	false
se-cleanup-discontinuance	Cleanup Discontinuance (SE task)	false
se-configure-on-off	Configure On/Off	true
se-cp-details	CP Details (SE task)	false
se-crypto-details	Crypto Details (SE task)	false
se-cryptographic-configuration	Cryptographic Configuration	true
se-cryptographic-management	Cryptographic Management	true
se-define-clonable-internal-code-levels	Define Clonable Internal Code Levels (SE task)	false
se-delete-lpar-dump-data	Delete LPAR Dump Data (SE task)	false
se-display-adapter-id	Display Adapter ID	false
se-display-or-alter	Display or Alter (SE task)	false
se-dump-lpar-data	Dump LPAR Data (SE task)	false
se-dump-machine-loader-data	Dump Machine Loader Data (SE task)	false
se-edit-lpar-internal-code-change	Edit LPAR Internal Code Change (SE task)	false
se-enable-disable-dynamic-channel-subsystem	Enable/Disable Dynamic Channel Subsystem	false
se-energy-optimization-advisor	Energy Optimization Advisor (SE task)	false
se-fcp-configuration	FCP Configuration	false
se-fcp-npiv-mode-on-off	FCP NPIV Mode On/Off	false
se-fid-details	FID Details (SE task)	false
se-force-channel-internal-code-change	Force Channel Internal Code Change (SE task)	false
se-ifcc-and-other-errors	IFCC and Other Errors (SE task)	false
se-input-output-configuration	Input/Output (I/O) Configuration	false
se-interrupt	Interrupt (SE task)	false
se-load-from-removable-media-or-server	Load from Removable Media or Server (SE task)	false
se-load-processor-from-file	Load Processor From File (SE task)	false
se-lpar-internal-code-change-utility	LPAR Internal Code Change Utility (SE task)	false
se-manage-pci-system-services	Manage PCI System Services	false
se-migrate-channel-configuration-files	Migrate Channel Configuration Files (SE task)	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
se-msq-processor-test	MSQ Processor Test (SE task)	false
se-network-traffic-analyzer-authorization	Network Traffic Analyzer Authorization	false
se-nnhb-monitor	NNhb Monitor (SE task)	false
se-non-disruptive-hardware-change	Nondisruptive Hardware Change (SE task)	false
se-offload-pa-data-to-hmc-removable-media	Offload Problem Analysis Data to HMC removable Media (SE task)	false
se-perform-model-conversion	Perform Model Conversion	false
se-power-off	Power Off (SE task)	false
se-power-on	Power On (SE task)	false
se-power-on-reset	Power-on Reset (SE task)	false
se-prepare-system-for-discontinuance	Prepare System For Discontinuance (SE task)	false
se-query-channel-crypto-configure-on-off-pending	Query Channel/Crypto Configure On/Off Pending	false
se-query-coupling-facility-reactivations	Query Coupling Facility Reactivations	false
se-query-internal-code-changes-pending-power-on-reset	Query Internal Code Changes Pending Power-on Reset (SE task)	false
se-reassign-io-path	Reassign I/O Path (SE task)	false
se-redundant-io-interconnect-status-and-control	Redundant I/O Interconnect® Status and Control	false
se-release-io-path	Release I/O Path	false
se-reset-error-thresholds	Reset Error Thresholds	false
se-reset-swap-channel-path	Reset Swap Channel Path (SE task)	false
se-selective-channel-patch-controls	Selective Channel Patch Controls (SE task)	false
se-service-on-off	Service On/Off	false
se-service-required-state-query	Service Required State Query	false
se-show-led	Show LED	false
se-specify-concurrent-upgrade-sync-point	Specify Concurrent Upgrade Sync Point (SE task)	false
se-start-processor	Start Processor (SE task)	false
se-stop-processor	Stop Processor (SE task)	false
se-stop-processor-on-cp-address-match	Stop Processor on CP Address Match (SE task)	false
se-storage-information	Storage Information	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
se-store-status	Store Status (SE task)	false
se-swap-channel-patch	Swap Channel Path (SE task)	false
se-transmit-vital-product-data	Transmit Vital Product Data (SE task)	false
se-update-hom-and-vpd	Update HOM and VPD (SE task)	false
se-update-io-world-wide-port-number	Update I/O World Wide Port Number (SE task)	false
se-update-pci-adapter-internal-code	Update PCI Adapter Internal Code	false
se-view-hardware-configuration	View Hardware Configuration (SE task)	false
se-view-internal-code-changes-summary	View Internal Code Changes Summary	false
se-view-lpar-cryptographic-controls	View LPAR Cryptographic Controls	false
service-status	Service Status	false
set-ctn-member-restriction	Set CTN Member Restriction	false
set-power-cap	Set Power Cap	false
set-power-saving	Set Power Saving	false
setup-new-ctn	Setup New CTN	false
shutdown-or-restart	Power Off or Restart	false
single-object-operations	Single Object Operations	false
single-step-console-internal-code	Single Step Console Internal Code	false
single-step-internal-code-changes	Single Step Internal Code Changes	false
split-to-new-ctn	Split to New CTN	false
start-all	Start All Processors	false
start-partitions	Start	false
start-systems	Start	false
stop-all	Stop All Processors	false
stop-partitions	Stop	false
stop-systems	Stop	false
system-activity-display	Activity	false
system-details	System Details	false
system-information	System Information	false
system-input-output-configuration-analyzer	System Input/Output Configuration Analyzer	false

Table 394. Enum values for the **name** property of Task objects (continued)

Enum for task name	Task	view-only-mode supported
tip-of-the-day	Tip of the Day	false
toggle-lock	Toggle Lock	false
transmit-console-service-data	Transmit Console Service Data	false
transmit-service-data	Transmit Service Data	false
transmit-vital-product-data-base	Transmit Vital Product Data	false
transmit-vital-product-data-z	Transmit Vital Product Data	false
user-settings	User Settings	false
users-and-tasks	Users and Tasks	false
view-activation-profiles	View Activation Profiles	false
view-adapter-security	View Adapter Security	false
view-certificate-details	View Certificate Details	false
view-console-events	View Console Events	false
view-console-information	View Console Information	false
view-console-service-history	View Console Service History	false
view-console-tasks-performed	View Console Tasks Performed	false
view-external-time-source	View External Time Source	false
view-frame-layout	View Frame Layout	false
view-licenses	View Licenses	false
view-partition-resource-assignments	View Partition Resource Assignments	false
view-pulse-per-second-signal	View Pulse Per Second Signal	false
view-security-logs	View Security Logs	false
view-service-history	View Service History	false
virtual-media-load	Manage Console Recovery	false
virtual-se-management	Virtual Support Element Management	false
whats-new	What's New	false
zeroize-crypto-domain	Zeroize Crypto Domain (API only)	false

Appendix D. Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other

companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Unix is a trademark of The Open Group in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.

Class A Notices

The following Class A statements apply to this IBM product. The statement for other IBM products intended for use with this product will appear in their accompanying manuals.

Federal Communications Commission (FCC) Statement

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his own expense.

Properly shielded and grounded cables and connectors must be used in order to meet FCC emission limits. IBM is not responsible for any radio or television interference caused by using other than recommended cables and connectors or by unauthorized changes or modifications to this equipment. Unauthorized changes or modifications could void the user's authority to operate the equipment.

This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Industry Canada Compliance Statement

This Class A digital apparatus complies with Canadian ICES-003.

Avis de conformité à la réglementation d'Industrie Canada

Cet appareil numérique de la classe A est conforme à la norme NMB-003 du Canada.

European Community Compliance Statement

This product is in conformity with the protection requirements of EU Council Directive 2014/30/EU on the approximation of the laws of the Member States relating to electromagnetic compatibility. IBM cannot accept responsibility for any failure to satisfy the protection requirements resulting from a non-recommended modification of the product, including the fitting of non-IBM option cards.

This product has been tested and found to comply with the limits for Class A Information Technology Equipment according to European Standard EN 55032. The limits for Class A equipment were derived for commercial and industrial environments to provide reasonable protection against interference with licensed communication equipment.

European Community contact:
IBM Deutschland GmbH

Technical Regulations, Department M372
IBM-Allee 1, 71139 Ehningen, Germany
Tele: +49 (0) 800 225 5423 or +49 (0) 180 331 3233
email: halloibm@de.ibm.com

Warning: This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user may be required to take adequate measures.

VCCI Statement - Japan

この装置は、クラス A 情報技術装置です。この装置を家庭環境で使用する
と電波妨害を引き起こすことがあります。この場合には使用者が適切な対策
を講ずるよう要求されることがあります。
VCCI-A

The following is a summary of the Japanese VCCI statement above:

This is a Class A product based on the standard of the VCCI Council. If this equipment is used in a domestic environment, radio interference may occur, in which case the user may be required to take corrective actions.

Japan JIS C 61000-3-2 Compliance

(一社) 電子情報技術産業協会 高調波電流抑制対策実施
要領に基づく定格入力電力値： Knowledge Centerの各製品の
仕様ページ参照

For products less than or equal to 20 A per phase, the following statement applies:

高調波電流規格 JIS C 61000-3-2 適合品

For products greater than 20 A, single-phase, the following statements apply:

高調波電流規格 JIS C 61000-3-2 準用品

本装置は、「高圧又は特別高圧で受電する需要家の高調波抑制対
策ガイドライン」対象機器（高調波発生機器）です。

回路分類：6（単相、P F C回路付）

換算係数：0

For products greater than 20 A per phase, three-phase, the following statements apply:

高調波電流規格 JIS C 61000-3-2 準用品

本装置は、「高圧又は特別高圧で受電する需要家の高調波抑制対
策ガイドライン」対象機器（高調波発生機器）です。

回路分類：5（3相、P F C回路付）

換算係数：0

Electromagnetic Interference (EMI) Statement - People's Republic of China

声 明

此为 A 级产品,在生活环境中,
该产品可能会造成无线电干扰。
在这种情况下,可能需要用户对其
干扰采取切实可行的措施。

Declaration: This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user may need to perform practical action.

Electromagnetic Interference (EMI) Statement - Taiwan

警告使用者：
這是甲類的資訊產品，在
居住的環境中使用時，可
能會造成射頻干擾，在這
種情況下，使用者會被要
求採取某些適當的對策。

The following is a summary of the Taiwan EMI statement above:

Warning: This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user will be required to take adequate measures.

IBM Taiwan Contact Information:

台灣IBM 產品服務聯絡方式：
台灣國際商業機器股份有限公司
台北市松仁路7號3樓
電話：0800-016-888

Electromagnetic Interference (EMI) Statement - Korea

이 기기는 업무용(A급)으로 전자파적합등록을 한 기기이오니
판매자 또는 사용자는 이 점을 주의하시기 바라며, 가정외의
지역에서 사용하는 것을 목적으로 합니다.

Germany Compliance Statement

**Deutschsprachiger EU Hinweis: Hinweis für Geräte der Klasse A EU-Richtlinie zur
Elektromagnetischen Verträglichkeit**

Dieses Produkt entspricht den Schutzanforderungen der EU-Richtlinie 2014/30/EU zur Angleichung der Rechtsvorschriften über die elektromagnetische Verträglichkeit in den EU-Mitgliedsstaaten und hält die Grenzwerte der EN 55032 Klasse A ein.

Um dieses sicherzustellen, sind die Geräte wie in den Handbüchern beschrieben zu installieren und zu betreiben. Des Weiteren dürfen auch nur von der IBM empfohlene Kabel angeschlossen werden. IBM übernimmt keine Verantwortung für die Einhaltung der Schutzanforderungen, wenn das Produkt ohne Zustimmung von IBM verändert bzw. wenn Erweiterungskomponenten von Fremdherstellern ohne Empfehlung von IBM gesteckt/eingebaut werden.

EN 55032 Klasse A Geräte müssen mit folgendem Warnhinweis versehen werden:

"Warnung: Dieses ist eine Einrichtung der Klasse A. Diese Einrichtung kann im Wohnbereich Funk-Störungen verursachen; in diesem Fall kann vom Betreiber verlangt werden, angemessene Maßnahmen zu ergreifen und dafür aufzukommen."

Deutschland: Einhaltung des Gesetzes über die elektromagnetische Verträglichkeit von Geräten

Dieses Produkt entspricht dem "Gesetz über die elektromagnetische Verträglichkeit von Geräten (EMVG)". Dies ist die Umsetzung der EU-Richtlinie 2014/30/EU in der Bundesrepublik Deutschland.

Zulassungsbescheinigung laut dem Deutschen Gesetz über die elektromagnetische Verträglichkeit von Geräten (EMVG) (bzw. der EMC EG Richtlinie 2014/30/EU) für Geräte der Klasse A

Dieses Gerät ist berechtigt, in Übereinstimmung mit dem Deutschen EMVG das EG-Konformitätszeichen - CE - zu führen.

Verantwortlich für die Einhaltung der EMV Vorschriften ist der Hersteller:
International Business Machines Corp.
New Orchard Road
Armonk, New York 10504
Tel: 914-499-1900

Der verantwortliche Ansprechpartner des Herstellers in der EU ist:
IBM Deutschland GmbH
Technical Regulations, Abteilung M372
IBM-Allee 1, 71139 Ehningen, Germany
Tel: +49 (0) 800 225 5423 or +49 (0) 180 331 3233
email: halloibm@de.ibm.com

Generelle Informationen:

Das Gerät erfüllt die Schutzanforderungen nach EN 55024 und EN 55032 Klasse A.

Electromagnetic Interference (EMI) Statement - Russia

ВНИМАНИЕ! Настоящее изделие относится к классу A.
В жилых помещениях оно может создавать радиопомехи, для
снижения которых необходимы дополнительные меры

Index

A

Accept Mismatched FCP Storage Volumes [490](#)
accessibility
 contact IBM [liii](#)
 features [liv](#)
Activate CPC [775](#)
Activate Logical Partition [861](#)
Add Candidate Adapter Ports to an FCP Storage Group [472](#)
Add Connection Endpoint [399](#)
Add Member to Custom Group [736](#)
Add Partition to Capacity Group [338](#)
Add Permission to User Role [665](#)
Add Temporary Capacity [785](#)
Add User Role to User [644](#)
Add Volume Range [416](#)
API version number, function included in [7](#)
assistive technologies [liii](#)
Attach Storage Group to Partition [256](#)

C

Cancel Job [108](#)
Change Adapter Type [316](#)
Change Crypto Domain Configuration [245](#)
Change Crypto Type [302](#)
Change Logon Password [91](#)
Change STP-only Coordinated Timing Network [793](#)
Create Capacity Group [332](#)
Create Custom Group [733](#)
Create HBA [260](#)
Create Hipersocket [304](#)
Create LDAP Server Definition [714](#)
Create Metrics Context [121](#)
Create MFA Server Definition [724](#)
Create NIC [230](#)
Create Partition [185](#)
Create Password Rule [701](#)
Create Storage Fabric [359](#)
Create Storage Group [450](#)
Create Storage Path [420](#)
Create Storage Site [347](#)
Create Storage Template [543](#)
Create User [648](#)
Create User Pattern [687](#)
Create User Role [671](#)
Create Virtual Function [223](#)

D

Deactivate CPC [778](#)
Deactivate Logical Partition [863](#)
Decline Console Service [610](#)
Decline CPC Service [813](#)
Decrease Crypto Configuration [247](#)
Define Storage Control Unit [409](#)
Define Storage Subsystem [389](#)

Define Storage Switch [373](#)
Delete Capacity Group [334](#)
Delete Completed Job Status [106](#)
Delete Console Hardware Message [605](#)
Delete CPC Hardware Message [807](#)
Delete Custom Group [735](#)
Delete HBA [263](#)
Delete Hipersocket [306](#)
Delete LDAP Server Definition [716](#)
Delete Logical Partition OS Message [883](#)
Delete Metrics Context [127](#)
Delete MFA Server Definition [726](#)
Delete NIC [235](#)
Delete Partition [191](#)
Delete Partition Asynchronously [193](#)
Delete Partition OS Message [278](#)
Delete Password Rule [703](#)
Delete Storage Fabric [362](#)
Delete Storage Group [456](#)
Delete Storage Path [423](#)
Delete Storage Site [350](#)
Delete Storage Template [546](#)
Delete User [652](#)
Delete User Pattern [690](#)
Delete User Role [673](#)
Delete Virtual Function [225](#)
Detach Storage Group from Partition [258](#)
Dump Partition [213](#)

E

Enum values for a class of managed objects [973](#)
Enum values for Task objects [977](#)
Enum values for User Role objects [975](#)
Establish Shared Secret Key [86](#)
Export Profiles [781](#)
Export WWPN List [814](#)

F

FICON storage configuration [147](#)
Flash memory adapters [138](#)
Fulfill FCP Storage Volume [488](#)
Fulfill FICON Storage Volume [481](#)
Fulfill FICON Storage Volumes [484](#)

G

Get Adapter Properties [298](#)
Get ASCII Console WebSocket URI [279](#)
Get Capacity Group Properties [336](#)
Get Capacity Record Properties [943](#)
Get Connected VNICs of a Virtual Switch [324](#)
Get Connection Report [509](#)
Get Console Audit Log [593](#)
Get Console Hardware Message Properties [603](#)

- Get Console Properties [582](#)
- Get Console Security Log [598](#)
- Get Console Service Request Information [608](#)
- Get CPC Audit Log [797](#)
- Get CPC Energy Management Data [959](#)
- Get CPC Hardware Message Properties [805](#)
- Get CPC Notification Preferences for Device [618](#)
- Get CPC Properties [761](#)
- Get CPC Security Log [799](#)
- Get CPC Service Request Information [811](#)
- Get Custom Group Properties [731](#)
- Get Energy Optimization Advice Details [966](#)
- Get Energy Optimization Advice Summary [960](#)
- Get Group Profile Properties [935](#)
- Get HBA Properties [267](#)
- Get Image Activation Profile Properties [917](#)
- Get Inventory [112](#)
- Get LDAP Server Definition Properties [710](#)
- Get Load Activation Profile Properties [927](#)
- Get Logical Partition Properties [855](#)
- Get Metrics [124](#)
- Get MFA Server Definition Properties [721](#)
- Get Mobile App Preferences [613](#)
- Get Network Port Properties [309](#)
- Get NIC Properties [237](#)
- Get Notification Topics [94](#)
- Get Partition Properties [195](#)
- Get Partitions Assigned to Adapter [307](#)
- Get Partitions for a Storage Group [502](#)
- Get Password Rule Properties [697](#)
- Get Reset Activation Profile Properties [896](#)
- Get Storage Control Unit Properties [413](#)
- Get Storage Fabric Properties [363](#)
- Get Storage Group Histories [517](#)
- Get Storage Group Properties [459](#)
- Get Storage Path Properties [424](#)
- Get Storage Port Properties [313](#)
- Get Storage Site Properties [352](#)
- Get Storage Subsystem Properties [393](#)
- Get Storage Switch Properties [377](#)
- Get Storage Template Properties [548](#)
- Get Storage Template Volume Properties [558](#)
- Get Storage Volume Properties [479](#)
- Get Task Properties [678](#)
- Get User Pattern Properties [684](#)
- Get User Properties [639](#)
- Get User Role Properties [661](#)
- Get Virtual Function Properties [227](#)
- Get Virtual Storage Resource Properties [498](#)
- Get Virtual Switch Properties [323](#)

I

- IBM z Unified Resource Manager [3](#)
- Import DPM Configuration [817](#)
- Import Profiles [780](#)
- Increase Crypto Configuration [242](#)

J

- Join STP-only Coordinated Timing Network [794](#)

K

- keyboard
- navigation [liii](#)

L

- Leave STP-only Coordinated Timing Network [796](#)
- List Adapters of a CPC [295](#)
- List Capacity Groups of a CPC [330](#)
- List Capacity Records [942](#)
- List Console Hardware Messages [601](#)
- List CPC Hardware Messages [802](#)
- List CPC Objects [759](#)
- List Custom Group Members [739](#)
- List Custom Groups [730](#)
- List Group Profiles [933](#)
- List Image Activation Profiles [915](#)
- List LDAP Server Definitions [708](#)
- List Load Activation Profiles [925](#)
- List Logical Partitions of CPC [850](#)
- List Managed Virtual Machines of a Logical Partition [890](#)
- List MFA Server Definitions [719](#)
- List OS Messages of a Logical Partition [880](#)
- List OS Messages of a Partition [274](#)
- List Partitions of a CPC [180](#)
- List Password Rules [695](#)
- List Permitted Logical Partitions [852](#)
- List Permitted Partitions [182](#)
- List Reset Activation Profiles [894](#)
- List Storage Control Units of a Storage Subsystem [407](#)
- List Storage Fabrics [357](#)
- List Storage Groups [447](#)
- List Storage Sites [345](#)
- List Storage Subsystems of a Storage Site [387](#)
- List Storage Switches of a Storage Fabric [371](#)
- List Storage Switches of a Storage Site [369](#)
- List Storage Template Volumes of a Storage Template [555](#)
- List Storage Templates [541](#)
- List Storage Volumes of a Storage Group [476](#)
- List Tasks [676](#)
- List Unmanaged CPCs [611](#)
- List User Patterns [682](#)
- List User Roles [659](#)
- List Users [637](#)
- List Virtual Storage Resources of a Storage Group [495](#)
- List Virtual Switches of a CPC [321](#)
- Load Logical Partition [869](#)
- Logoff [93](#)
- Logon [80](#)

M

- managed objects, valid enum values [973](#)
- Modify Storage Group Properties [461](#)
- Modify Storage Template Properties [549](#)
- Mount ISO Image [252](#)
- Move Storage Subsystem to Storage Site [397](#)
- Move Storage Switch to Storage Fabric [383](#)
- Move Storage Switch to Storage Site [381](#)

N

navigation
 keyboard [liii](#)

O

Open OS Message Channel [272](#), [878](#)

P

Perform PSW Restart [220](#)
Provide Requested MFA Information [88](#)
PSW Restart [871](#)

Q

Query API Version [78](#)
Query Job Status [103](#)

R

Reassign Storage Adapter Port [268](#)
Reject Mismatched FCP Storage Volumes [492](#)
Remove Candidate Adapter Ports from an FCP Storage Group [474](#)
Remove Connection Endpoint [401](#)
Remove Member from Custom Group [738](#)
Remove Partition from Capacity Group [340](#)
Remove Permission from User Role [668](#)
Remove Temporary Capacity [788](#)
Remove User Role from User [646](#)
Remove Volume Range [418](#)
Reorder User Patterns [591](#)
Request Console Service [606](#)
Request CPC Service [809](#)
Resend Request [469](#)
Reset Clear [867](#)
Reset Normal [865](#)
Restart Console [588](#)
RoCE adapters [139](#)

S

SCSI Dump [887](#)
SCSI Load [885](#)
Send OS Command [271](#), [876](#)
Set Auto-Start List [783](#)
Set CPC Power Capping [952](#)
Set CPC Power Save [950](#)
Set Mobile App Preferences [615](#)
Set STP Configuration [791](#)
Set zCPC Power Capping [957](#)
Set zCPC Power Save [955](#)
shortcut keys [liii](#)
Shutdown Console [589](#)
Start CPC [771](#)
Start Dump Program [216](#)
Start FCP Storage Discovery [506](#)
Start Logical Partition [873](#)
Start Partition [206](#)
Stop CPC [773](#)
Stop Logical Partition [874](#)

Stop Partition [210](#)
Submit Requests [96](#)
Summary of updates by API version number [7](#)
Swap Current Time Server [789](#)

T

Task objects, valid enum values [977](#)

U

Undefine Storage Control Unit [411](#)
Undefine Storage Subsystem [392](#)
Undefine Storage Switch [376](#)
Unified Resource Manager [3](#)
Unmount ISO Image [254](#)
Update Adapter Properties [299](#)
Update Capacity Group Properties [342](#)
Update CPC Notification Preferences for Device [622](#)
Update CPC Properties [769](#)
Update Group Profile Properties [937](#)
Update HBA Properties [265](#)
Update Image Activation Profile Properties [920](#)
Update LDAP Server Definition Properties [712](#)
Update Load Activation Profile Properties [930](#)
Update Logical Partition Properties [860](#)
Update MFA Server Definition Properties [723](#)
Update Network Port Properties [311](#)
Update NIC Properties [238](#)
Update Partition Properties [198](#)
Update Partition Properties Asynchronously [202](#)
Update Password Rule Properties [699](#)
Update Reset Activation Profile Properties [898](#)
Update Storage Control Unit Properties [414](#)
Update Storage Fabric Properties [365](#)
Update Storage Path Properties [426](#)
Update Storage Port Properties [314](#)
Update Storage Site Properties [353](#)
Update Storage Subsystem Properties [395](#)
Update Storage Switch Properties [379](#)
Update User Pattern Properties [685](#)
Update User Properties [641](#)
Update User Role Properties [663](#)
Update Virtual Function Properties [228](#)
Update Virtual Storage Resource Properties [499](#)
Update Virtual Switch Properties [326](#)
User Role objects, valid enum values [975](#)

V

Validate LUN Path [504](#)

W

Web Services API [3](#)

Z

Zeroize Crypto Domain [250](#)
zManager [3](#)



SC27-2638-02

