

IBM Security Web Gateway Appliance
バージョン 7.0

Web リバース・プロキシの 構成ガイド



IBM Security Web Gateway Appliance
バージョン 7.0

Web リバース・プロキシの 構成ガイド



お願い

本書および本書で紹介する製品をご使用になる前に、775 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM Security Access Manager (製品番号 5724-C87) バージョン 7、リリース 0、モディフィケーション 0、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC22-5433-01
IBM Security Web Gateway Appliance
Version 7.0
Configuration Guide for Web Reverse
Proxy

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2013.4

© Copyright IBM Corporation 2002, 2013.

目次

図	xiii

表	xv

本書について	xvii

対象読者	xvii

資料および用語集へのアクセス	xviii

関連資料	xxi

アクセシビリティ	xxiii

技術研修	xxiii

サポート情報	xxiii

第 1 部 管理 1

第 1 章 IBM Security Access Manager for Web WebSEAL の概要 3

概要	3

WebSEAL の概要	4

IBM Security Web Gateway Appliance	5

アプライアンスでの WebSEAL の機能	5

セキュリティ・モデル	7

セキュリティ・モデルの概念	7

保護オブジェクト・スペース	8

アクセス・コントロール・リスト (ACL) および保	

護オブジェクト・ポリシー (POP)	9

アクセス・コントロール・リスト (ACL) ポリシ	

ー	10

保護オブジェクト・ポリシー (POP)	10

明示ポリシーと継承ポリシー	11

ポリシー管理: Web Portal Manager	11

Web スペースの保護	12

セキュリティ・ポリシー計画およびインプリメン	

テーション	13

コンテンツ・タイプおよび保護レベル	14

WebSEAL 認証	15

標準 WebSEAL Junction	16

Web スペース・スケーラビリティ	18

複製フロントエンド WebSEAL サーバー	18

Junction されたバックエンド・サーバー	18

複製バックエンド・サーバー	20

第 2 章 サーバー管理 21

WebSEAL インスタンス管理	21

複数のサーバー間での WebSEAL データの同期化	22

同期の自動化	23

WebSEAL 用リソースの監査およびロギング	26

エラー・メッセージ・ロギング	26

WebSEAL サーバー・アクティビティの監査	26

従来の HTTP イベントの監査およびロギング	27

WebSEAL の問題判別リソース	28

構成データ・ログ・ファイル	28

統計	30

Trace ユーティリティ	30

第 2 部 構成 33

第 3 章 Web サーバー構成 35

WebSEAL サーバーおよびホスト名指定	35

構成ファイル内の WebSEAL サーバー名	35

「pdadmin サーバー・リスト」内の WebSEAL サ	

ーバー名	36

保護オブジェクト・スペース内の WebSEAL サー	

バー名	36

WebSEAL ホスト (マシン) 名の指定	37

WebSEAL 構成ファイル	37

構成ファイルの編成	38

構成ファイルの名前とロケーション	38

構成ファイルの設定の変更	38

ディレクトリーの索引付け	39

ディレクトリーの索引付けの構成	39

ファイル・タイプごとのグラフィカル・アイコン	

の構成	40

コンテンツのキャッシング	40

コンテンツ・キャッシングの概念	41

コンテンツ・キャッシングの構成	41

WebSEAL コンテンツ・キャッシングへの HTTP	

ヘッダーの与える影響	42

すべてのキャッシュのフラッシュ	45

特定文書に関するキャッシュ・コントロール	45

通信プロトコル構成	46

HTTP 要求用の WebSEAL の構成	46

HTTPS 要求用の WebSEAL の構成	47

特定の SSL バージョンからの接続の制限	47

HTTP 持続接続	48

HTTPOnly Cookie を処理するための WebSEAL	

構成	48

HTTP および HTTPS 通信用のタイムアウト設定	49

追加の WebSEAL サーバー・タイムアウト設定	51

WebDAV のサポート	52

Microsoft RPC over HTTP のサポート	54

チャンク転送コーディングのサポート	54

インターネット・プロトコル・バージョン 6 (IPv6)	

のサポート	54

IPv4 および IPv6 の概要	54

IPv6 および IPv4 のサポートの構成	55

IPv6: 互換性のサポート	56

IPv6: アップグレードの注意	56

資格情報属性の IP レベル	57

LDAP ディレクトリー・サーバー構成	57

ワーカー・スレッドの割り振り	58

WebSEAL ワーカー・スレッド構成	58
junction 用のワーカー・スレッドの割り振り (Junction フェアネス)	59
HTTP データ圧縮	62
MIME タイプに基づく圧縮	62
ユーザー・エージェント・タイプに基づく圧縮	63
POP 内の圧縮ポリシー	64
データ圧縮の制限	64
データ圧縮ポリシーの構成	65
UTF-8 でのマルチロケール・サポート	65
マルチロケール・サポートの概念	65
マルチロケール・サポートの構成	70
要求データ内の文字エンコード方式の検証	76
サポートされているワイルドカード・パターン・マ ッチング文字	77
システム環境変数の設定	78
第 4 章 Web サーバー応答の構成	79
静的 HTML サーバー応答ページ	79
HTML サーバー応答ページのロケーション	84
管理ルート	85
アカウント管理ページのロケーション	85
エラー・メッセージ・ページのロケーション	85
junction 固有の静的サーバー応答ページ	86
HTML サーバー応答ページの変更	86
HTML 応答ページのカスタマイズのためのガイド ライン	87
HTML 応答ページのカスタマイズのためのマク ロ・リソース	87
テンプレートの組み込みマクロ	90
カスタム・ログイン・フォームへのイメージの追 加	93
アカウント管理ページの構成	94
構成ファイル・スタンザ・エントリーおよび値	94
アカウント期限切れエラー・メッセージの構成	94
パスワード・ポリシー・オプションの構成	95
エラー・メッセージ・ページの構成	96
時刻エラー・ページの使用可能化	97
新規 HTML エラー・メッセージ・ページの作成	97
WebSEAL の前のバージョンとの互換性	98
サーバー応答でのマルチロケール・サポート	98
Accept-Language HTTP ヘッダー	98
マルチロケール・サポート用プロセス・フロー	99
WebSEAL におけるマルチロケール・サポートに 影響する条件	100
Mozilla Firefox での favicon.ico ファイルの処理	100
サーバー応答ページへのカスタム・ヘッダーの追加 リダイレクト応答内のロケーション URL フォーマ ットの構成	102
ローカル応答リダイレクト	103
ローカル応答リダイレクトの概要	104
ローカル応答リダイレクトのプロセス・フロー	104
ローカル応答リダイレクトの使用可能化と使用不 可化	105
リダイレクトされた応答の内容	105
ローカル応答リダイレクト用 URI	105

ローカル応答リダイレクトの操作	106
ローカル応答リダイレクト用マクロ・サポート	108
ローカル応答リダイレクト構成の例	112
ローカル応答リダイレクトの技術上の注意点	113
ローカル認証によるリモート応答処理	113
HTML リダイレクト	115
HTML リダイレクトの使用可能化	115
リダイレクト時の HTML フラグメントの保持	115

第 5 章 Web サーバー・セキュリティ ー構成 119

Suite B 暗号のみをサポートするための WebSEAL の構成	119
クロスサイト・スクリプトがもたらすぜい弱性の回 避	120
クロスサイト・リクエスト・フォージェリー (CSRF) アタックの防止	121
秘密トークンの検証	121
リファラー検証	122
非送信請求認証要求の拒否	123
WebSEAL およびバックエンド・サーバー識別の抑 止	124
WebSEAL サーバー識別の抑止	124
バックエンド・アプリケーション・サーバー識別 の抑止	125
HTTP メソッドの使用不可化	125
Platform for Privacy Preferences (P3P)	126
コンパクト・ポリシーの概要	126
コンパクト・ポリシーの宣言	128
Junction ヘッダーの保存	129
P3P ヘッダー内のデフォルト・コンパクト・ポ リシー	129
P3P ヘッダーの構成	130
カスタム P3P コンパクト・ポリシーの指定	137
P3P 構成のトラブルシューティング	137

第 6 章 ランタイム・セキュリティー・ サービス外部許可サービス 139

ランタイム・セキュリティー・サービス外部許可サ ービスについて	139
WebSEAL におけるランタイム・セキュリティー・ サービス外部許可サービスの構成	140
ランタイム・セキュリティー・サービス外部許可サ ービスのサンプル構成データ	143

第 3 部 認証 147

第 7 章 認証の概要 149

認証の定義および目的	149
ユーザー要求内の情報	149
クライアント識別および資格情報	150
認証プロセス・フロー	151
リソースへの認証アクセスと非認証アクセス	152
認証済みユーザーの要求プロセス	152
非認証ユーザーの要求プロセス	152

SSL を介したアクセス条件	153
ユーザーの強制ログイン	153
非認証 HTTPS の使用	154
サポートされる認証方式	154
ユーザー・エージェントに基づく認証要求	154

第 8 章 認証方式 157

認証に関する用語	157
ログアウトおよびパスワード変更の操作	158
ログアウト: pkmslogout	158
pkmslogout でのカスタム応答ページのコントロ ール	158
パスワードの変更: pkmspasswd	159
Windows での Active Directory のパスワード変 更の問題	159
基本認証	160
基本認証の使用可能化および使用不可	160
レルム名の設定	161
フォーム認証	161
フォーム認証の使用可能および使用不可	161
HTML 応答フォームのカスタマイズ	162
WebSEAL へのログイン・フォーム・データの直 接送付	162
クライアント・サイド証明書認証	164
クライアント・サイド証明書認証モード	164
証明書認証の構成タスクの要約	167
証明書認証の使用可能化	167
証明書認証メカニズムの構成	168
証明書ログイン・エラー・ページ	170
証明書ログイン・フォーム	171
セッション・トラッキング用の SSL セッション ID の使用不可化	171
証明書 SSL ID キャッシュの使用可能化と構成 証明書 SSL ID キャッシュのタイムアウトの設 定	172
誤ったプロトコル用のエラー・ページ	173
証明書認証の使用不可化	173
証明書 SSL ID キャッシュの使用不可化	174
証明書認証に関する技術上の注意点	174
Kerberos 認証	174
Kerberos 認証の構成	175
制限	176
LTPA 認証	177
LTPA 認証の概要	177
LTPA 認証の使用可能化	178
鍵ファイル情報	178
クライアントの Cookie 名の指定	179
Junction の Cookie 名の指定	179
LTPA トークンの存続時間の制御	180
LTPA 認証の使用不可化	180

第 9 章 拡張認証方式 181

多重プロキシ・エージェント (MPA)	181
多重プロキシ・エージェント (MPA) の概要	181
有効なセッション・データ・タイプと認証方式	182

MPA および複数クライアントの認証プロセス・ フロー	183
MPA 認証の使用可能および使用不可	184
MPA 用ユーザー・アカウントの作成	184
webseal-mpa-servers グループへの MPA アカ ントの追加	184
MPA 認証に関する制限	184
ユーザー切り替え認証	185
ユーザー切り替え機能の概要	185
ユーザー切り替え認証の構成	188
ユーザー切り替えの使用	191
ユーザー切り替えの追加機能のサポート	192
再認証	193
再認証の概念	193
セキュリティ・ポリシーに基づく再認証	195
再認証 POP: 作成と適用	195
セッション非アクティブに基づく再認証	195
セッション非アクティブに基づく再認証の使用可 能化	196
セッション・キャッシュ・エントリーの存続時間 値のリセット	196
セッション・キャッシュ・エントリーの存続時間 値の延長	197
セッション存続時間満了に伴うセッション除去の 回避	198
ログイン失敗ポリシー制限でのユーザー・セッシ ョンの削除	199
再認証用のログイン・フォームのカスタマイズ	200
認証強度ポリシー (ステップアップ)	200
認証強度の概念	201
認証強度構成タスクの要約	202
認証強度ポリシーの確立	203
認証レベルの指定	203
認証強度ログイン・フォームの指定	205
保護オブジェクト・ポリシーの作成	206
ネットワーク・ベースのアクセス制限の指定	207
保護リソースへの保護オブジェクト・ポリシーの 付加	210
複数の認証レベル間でのユーザー ID マッチン グの実行	211
非認証ユーザーへのログイン応答の制御	212
より高いレベルへの認証のステップアップ	212
外部認証インターフェース	213
クライアント証明書ユーザー・マッピング	213
概要	213
ユーザー・マッピング・ルール評価プログラム	217
CDAS の管理方法	221
証明書マッピング・モジュールを使用するための WebSEAL の構成	223

第 10 章 認証後の処理 229

認証後の自動リダイレクト	229
自動リダイレクトの概要	229
自動リダイレクトの使用可能化	230
自動リダイレクトの使用不可	230
制限	231

自動リダイレクト用マクロ・サポート	231
サーバー・サイド要求キャッシング	233
サーバー・サイド要求キャッシングの概念	233
サーバー・サイド要求キャッシングのプロセス・フロー	234
サーバー・サイド・キャッシングの構成	235

第 11 章 パスワード処理 239

ログイン失敗ポリシー (「スリー・ストライク」ログイン・ポリシー)	239
ログイン失敗ポリシーの概念	239
ログイン失敗ポリシーの設定	240
アカウント使用不可時間間隔の設定	241
アカウント使用不可通知応答の構成	242
複製された WebSEAL サーバーにおけるログイン失敗ポリシー	242
パスワード・ストレングス・ポリシー	244
パスワード・ストレングス・ポリシーの概念	244
パスワード・ストレングス・ポリシー	244
パスワード・ストレングス・ポリシー・コマンドの構文	244
デフォルトのパスワード・ストレングス・ポリシーの値	246
有効なパスワードと無効なパスワードの例	246
ユーザーおよびグローバル設定の指定	247

第 12 章 資格情報の処理 249

資格情報用の拡張属性	249
資格情報にレジストリー属性を追加するためのメカニズム	249
レジストリー属性資格サービスの構成	250
拡張資格情報属性の junction の取り扱い	252
資格情報リフレッシュ	254
資格情報リフレッシュの概念	254
資格情報リフレッシュの構成	258
資格情報リフレッシュの使用法	260

第 13 章 外部認証インターフェース 263

外部認証インターフェースの概要	263
外部認証インターフェースのプロセス・フロー	263
外部認証インターフェースの構成	267
外部認証インターフェースの使用可能化	267
認証処理の開始	268
外部認証インターフェース・トリガー URL の構成	268
認証データの HTTP ヘッダー名	269
特殊 HTTP ヘッダーからの認証データの抽出	270
資格情報の生成方法	271
外部認証インターフェースの資格情報交換	271
ユーザー ID の検証	273
外部認証アプリケーションの作成方法	273
外部認証インターフェース HTTP ヘッダーのリファレンス	275
既存の WebSEAL 機能を持つ外部認証インターフェースの使用	276

外部認証インターフェースによる要求キャッシング	276
外部認証インターフェースによる認証後リダイレクト	277
外部認証インターフェースによるセッション処理	277
外部認証インターフェースによる認証強度レベル	277
外部認証インターフェースによる再認証	278
外部認証インターフェースによるログイン・ページおよびマクロのサポート	279
クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定	279
クライアント固有のセッション・キャッシュ・エントリー非アクティブ・タイムアウト値の設定	281

第 4 部 セッション状態 283

第 14 章 セッション状態の概要 285

セッション状態の概念	285
サポートされるセッション ID データ・タイプ	285
クライアント要求から取得される情報	286
WebSEAL セッション・キャッシュ構造	287
クラスター環境におけるデプロイメントの考慮事項	288
すべての WebSEAL レプリカ・サーバーでの整合性のとれた構成	288
ロード・バランサーでのクライアントとサーバー間のセッション・アフィニティー	288
新規マスターへのフェイルオーバー	288
ある WebSEAL サーバーから別のサーバーへのフェイルオーバー	289
クラスター環境におけるフェイルオーバー処理のオプション	289
オプション 1: WebSEAL がフェイルオーバー・イベントを処理しない	289
オプション 2: 認証データが各要求に含まれる	289
オプション 3: フェイルオーバー Cookie	290
オプション 4: Session Management Server	290
オプション 5: LTPA Cookie	291

第 15 章 セッション・キャッシュ構成 293

セッション・キャッシュ構成の概要	293
SSL セッション ID キャッシュ構成	294
キャッシュ・エントリーのタイムアウト値	294
最大同時 SSL セッション値	294
WebSEAL セッション・キャッシュの構成	295
最大セッション・キャッシュ・エントリー値	295
キャッシュ・エントリーの存続時間タイムアウト値	296
クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定	297
キャッシュ・エントリーの非アクティブ・タイムアウト値	299
並行セッションの制限	300
セッション・キャッシュの制限	301

第 16 章 フェイルオーバー・ソリューション 303

フェイルオーバー認証の概念	303
フェイルオーバー環境	303
フェイルオーバー Cookie	305
フェイルオーバー認証のプロセス・フロー	305
フェイルオーバー構成の例	306
フェイルオーバー Cookie へのデータの追加	306
フェイルオーバー Cookie からのデータの抽出	309
ドメイン全体にわたるフェイルオーバー認証	310
フェイルオーバー認証構成	311
フェイルオーバー認証の構成	312
フェイルオーバー Cookie のプロトコル	313
Cookie データを暗号化および暗号化解除する鍵 ペアの生成	313
フェイルオーバー Cookie 存続時間の指定	314
Cookie スtringで使用する UTF-8 エンコー ドの指定	314
認証強度レベルの追加	315
欠落したフェイルオーバー Cookie の再発行	315
セッション存続時間タイム・スタンプの追加	316
セッション・アクティビティ・タイム・スタン プの追加	316
アクティビティ・タイム・スタンプの更新間隔 の追加	317
追加の拡張属性	318
抽出のための属性	318
ドメイン全体のフェイルオーバー Cookie の使用 可能化	319
存続時間タイム・スタンプの妥当性検査	320
アクティビティ・タイム・スタンプの妥当性検 査	320
非スティッキー・フェイルオーバー環境のフェイル オーバー	321
非スティッキー・フェイルオーバーの概念	321
非スティッキー・フェイルオーバー・ソリューシ ョンの構成	323
既存の WebSEAL 機能と連動したフェイルオー バー Cookie の使用	324
フェイルオーバー環境におけるパスワード変更操作	324
第 17 章 非クラスター環境でのセッシ ョン状態	327
非クラスター環境でのセッション状態の維持	327
SSL を介したセッション状態情報のコントロー ル	327
異なるトランスポートでの同一セッション鍵の使 用	328
有効なセッション鍵データ・タイプ	329
有効なセッションのタイムアウト値	330
use-same-session に対する Netscape 4.7x の制限	330
セッション Cookie	331
セッション Cookie の概念	331
セッション Cookie の使用に関する条件	332
セッション Cookie 名のカスタマイズ	332
各要求を使用したセッション Cookie の送信	333
古いセッション Cookie に対するカスタマイズ応答	333

セッションの除去および古いセッション Cookie の概念	333
古いセッション Cookie に対するカスタマイズ応 答の使用可能化	335
HTTP ヘッダーを使用したセッション状態の維持	336
HTTP ヘッダー・セッション鍵の概念	336
セッション状態を維持するための HTTP ヘッダ ーの構成	337
MPA からの要求を求めるためのセットアップ	338
Microsoft Office アプリケーションとのセッション の共用	339
Microsoft Office アプリケーションとのセッショ ン共用の概要	339
一時セッション・キャッシュの構成	340
Microsoft Office アプリケーションとの共用セッ ションの構成	342

第 5 部 Session Management Server 347

第 18 章 Session Management Server (SMS) の概要 349

フェイルオーバー環境	349
Session Management Server (SMS)	350
サーバー・クラスター、レプリカ・セット、および セッション・レルム	351
SMS プロセス・フロー	351
複数の DNS ドメインでのセッション共有	353

第 19 章 SMS を使用した WebSEAL のクイック・スタート・ガイド 357

SMS を使用した WebSEAL の構成の要約	357
1. 情報の収集	357
2. WebSEAL 構成ファイル設定	358
3. Security Access Manager CA 証明書のインポー ート	358
4. WebSEAL サーバーを再始動する	359
5. 仮想ホストの junction の作成	359
6. Session Management Server の junction	359
7. 最大並行セッション・ポリシーの設定	360
8. 構成のテスト	360

第 20 章 SMS を使用した WebSEAL の構成 363

WebSEAL の SMS の構成	363
Session Management Server (SMS) の構成	363
WebSEAL の SMS の使用可能化と使用不可化	363
Session Management Server のクラスターとロー ケーションの指定	364
最大並行セッション・ポリシー値の検索	365
レプリカ・セット構成	365
複数のレプリカ・セットに参加する WebSEAL の構成	365
標準 junction のレプリカ・セットへの割り当て	366
レプリカ・セットに割り当てられる仮想ホスト	366

レプリカ・セット構成の例	367
SMS の最終アクセス時間更新頻度の調整	370
SMS 通信タイムアウトの構成	370
SMS 応答タイムアウトの構成	370
ブロードキャスト・イベントの接続タイムアウトの構成	371
SMS パフォーマンスの構成	372
事前割り振りセッション ID の最大数	372
ハンドル・プール・サイズの構成	372
SMS 認証	372
WebSEAL および SMS の SSL 構成	373
WebSEAL 鍵データベースの構成	373
SSL 証明書の識別名 (DN) の指定	374
SMS 接続用の GSKit 構成	375
最大並行セッション・ポリシー	375
最大並行セッション・ポリシーの設定	375
最大並行セッション・ポリシーの実行	379
ユーザーおよび最大並行セッション・ポリシーの切り替え	380
セッション・レルム内でのシングル・サインオン	381
セッション・レルムおよびセッション共用の概念	381
セッション共用の構成	382
ログイン・ヒストリーの構成	385
ログイン失敗の通知の使用可能化	385
Session Management Server への junction の作成	386
ログイン・ヒストリー JSP へのアクセスの許可	386
ログイン・ヒストリーを表示するための JSP のカスタマイズ	387

第 6 部 許可 389

第 21 章 許可の構成 391

WebSEAL 固有の ACL ポリシー	391
/WebSEAL/host-instance_name	391
/WebSEAL/host-instance_name/file	391
WebSEAL ACL 許可	392
デフォルトの /WebSEAL ACL ポリシー	392
ACL 名に使用できる文字	392
保護品質 POP	393
許可データベースの更新とポーリングの構成	393
保護品質レベルの構成	394
許可決定情報	397
OAuth 許可決定のサポート	397

第 22 章 鍵管理 403

鍵管理の概要	403
ローカル管理インターフェースの鍵管理	404
クライアント・サイドおよびサーバー・サイドの証明書の概念	404
WebSEAL 鍵データベース・ファイルの構成	405
WebSEAL 鍵データベース・ファイル	405
鍵データベース・ファイル・パスワード	406
WebSEAL テスト証明書	407
サーバー名表示	407
Security Access Manager でのサーバー間 SSL 通信	408

WebSEAL での証明書の失効	408
証明書取り消しリスト (CRL)	409
CRL 検査の構成	409
証明書の分散ポイント	410
CRL キャッシュの構成	410
キャッシュ・エントリーの最大数の設定	410
GSKit キャッシュ存続時間タイムアウト値の設定	411
CRL キャッシュの使用可能化	411
SSL 接続用 WebSEAL テスト証明書の使用	411

第 7 部 標準 WebSEAL junction 415

第 23 章 標準 WebSEAL junction . . 417

WebSEAL junction の概要	417
junction タイプ	418
大まかなアクセス・コントロールの適用: 要約	418
きめの細かいアクセス・コントロールの適用: 要約	418
WebSEAL junction に関する他の参照情報	419
Web Portal Manager を使用した Junction の管理	419
Web Portal Manager を使用した junction の作成	419
Web Portal Manager を使用した junction のリスト	420
Web Portal Manager を使用した junction の削除	420
ローカル管理インターフェースでの junction 管理	420
pdadmin ユーティリティを使用した junction の管理	421
junction データベースのインポートとエクスポート	422
標準 WebSEAL junction の構成	422
pdadmin server task create コマンド	422
TCP タイプの標準 junction の作成	423
SSL タイプの標準 junction の作成	423
mutual junction の作成	424
SSL ベース標準 junction	425
標準 junction への複数のバックエンド・サーバーの追加	426
ローカル・タイプの標準 Junction	426
ローカル junction の無効化	426
透過パス Junction	427
標準 WebSEAL junction のフィルター操作の概念	427
透過パス Junction の概念	428
透過パス Junction の構成	429
透過パス Junction の例	429
WebSEAL junction を使用するときの技術上の注意点	430
WebSEAL junction を作成するためのガイドライン	430
同一 junction への複数のバックエンド・サーバーの追加	431
Junction を介して許可を実施する場合の例外条件	432
junction を介した証明書認証	432
ドメイン Cookie の取り扱い	433

要求および応答でサポートされる HTTP バージョン	433
Web Portal Manager を持った junction 先アプリケーション	434
バックエンドのサーバー Web スペースの生成方法 (query_contents)	434
query_contents の概要	434
query_contents コンポーネント	436
UNIX ベースの Web サーバーでの query_contents のインストールおよび構成	437
Windows ベースの Web サーバーでの query_contents のインストールおよび構成	438
query_contents の一般プロセス・フロー	440
query_contents プログラムの保護	440
第 24 章 拡張 junction の構成	443
相互認証 SSL junction	443
相互認証 SSL junction プロセスの要約	443
バックエンド・サーバー証明書の妥当性検査	444
識別名 (DN) の突き合わせ	445
クライアント証明書による認証	445
BA ヘッダーによる認証	446
TCP および SSL のプロキシ junction の作成	446
SSL を介した WebSEAL から WebSEAL への junction	447
ステートフル junction	448
ステートフル junction の概念	449
ステートフル junction の構成	449
ステートフル junction のためのバックエンド・サーバー UUID の指定	450
使用不可なステートフル・サーバーの処理	453
新規 junction の強制適用	453
仮想ホスト Junction での /pkmslogout の使用	454
Junction スロットル	455
junction スロットルの概念	455
スロットル状態への junction サーバーの遷移	456
オフライン状態の Junction サーバー	458
オンライン状態の Junction サーバー	460
junction スロットル・メッセージ	462
junction スロットルと既存の WebSEAL 機能の併用	463
Cookie の管理	464
Junction ポータル・サーバーへのセッション Cookie の引き渡し	466
大文字小文字を区別しない URL のサポート	467
Windows ファイル・システムへの junction	468
例	469
小文字オブジェクト名への ACL および POP の付加	470
仮想ホストへの標準 Junction	470
HTTP ヘッダー・データ用の UTF-8 エンコード	471
リソース単位でのバッファリングのバイパス	472
junction を介したシングル・サインオン・ソリューション	473

第 25 章 Junction リソースへの URL の変更	475
URL の変更の概念	475
URL で使用されるパスのタイプ	476
URL の特殊文字	477
応答の中の URL の変更	478
タグ・ベースの静的 URL のフィルター操作	478
スクリプトのフィルター操作による絶対 URL の変更	487
rewrite-absolute-with-absolute オプションの構成	488
フィルター操作による Content-Length ヘッダーの変更	489
フィルター掛けされていないサーバー相対リンクでの制限	489
要求の中の URL の変更	490
junction マッピングによるサーバー相対 URL の変更	491
junction Cookie を使用したサーバー相対 URL の変更	493
junction Cookie JavaScript ブロックのコントロール	495
HTTP Referer ヘッダーを使用したサーバー相対 URL の変更	498
要求の中のサーバー相対 URL 処理のコントロール	499
複数の -j junction にわたるサーバーからの Cookie の処理	501
Cookie の処理: -j による Set-Cookie パス属性の変更	501
Cookie の処理: -j による Set-Cookie 名前属性の変更	502
Cookie 名の保存	503
Cookie の処理: -I による固有の Set-Cookie 名前属性の確認	504
第 26 章 HTTP 変換	507
HTTP 変換ルール	507
Extensible Stylesheet Language Transformation (XSLT)	508
HTTP 要求オブジェクト	508
HTTP 応答オブジェクト	509
HTTP 応答の置換	509
XSL Transformation ルール	510
再処理に関する考慮事項	511
XSLT テンプレート	511
構成	512
構成ファイルの更新	512
保護オブジェクト・ポリシー (POP)	513
HTTP 変換のシナリオ例	513
シナリオ 1: URI、ヘッダー、および Cookie の変更 (HTTPRequest)	513
シナリオ 2: ヘッダーのみの変更 (HTTPResponse)	516
シナリオ 3: ResponseLine/StatusCode のみの変更 (HTTPResponse)	518
シナリオ 4: Cookie のみの変更 (HTTPResponse)	519

シナリオ 5: 既知の HTTP 要求に対する応答の生成	522
変換エラー	523

第 27 章 HTTP を介した Microsoft RPC 525

WebSEAL での RPC over HTTP のサポート	525
junction 構成	526
POP 構成	527
認証の制限	527
タイムアウトの考慮事項	528
WebSEAL サーバー・ログのエラー	528
ワーカー・スレッドの考慮事項	529

第 28 章 コマンド・オプションの要約: 標準 junction 531

pdadmin サーバー・タスクによる junction の作成	531
junction のサーバー・タスク・コマンド	532
初期サーバーの junction の作成	533
既存 junction へのサーバーの追加	541

第 8 部 仮想ホスト機能 (Virtual Hosting) 545

第 29 章 仮想ホスト junction 547

仮想ホスト junction の概念	547
標準 WebSEAL junction	547
URL フィルターの課題	548
仮想ホスティング	548
仮想ホスト Junction のソリューション	548
仮想ホスト junction が無視するスタンザおよびスタンザ・エントリ	550
オブジェクト・スペース内で表される仮想ホスト	551
仮想ホスト junction の構成	551
リモート・タイプ仮想ホスト junction の作成	552
ローカル・タイプ仮想ホスト junction の作成	554
シナリオ 1: リモート仮想ホスト junction	556
仮想ホスト junction のインターフェースの定義	558
デフォルトのインターフェース仕様	558
追加インターフェースの定義	558
シナリオ 2: インターフェースを持つ仮想ホスト junction	560
既存の WebSEAL 機能での仮想ホストの使用	564
仮想ホストを使用した e-community シングル・サインオン	564
仮想ホストを使用したクロスドメイン・シングル・サインオン	565
仮想ホスト junction を使用した動的 URL	566
仮想ホスト・シングル・サインオンのためのドメイン・セッション Cookie の使用	566
Junction スロットル	568
シナリオ 3: 仮想ホストの拡張構成	568
仮想ホスト junction の制限	570
仮想ホストでの SSL セッション ID は使用不可	571

第 30 章 コマンド・オプションの要約: 仮想ホスト junction 573

pdadmin サーバー・タスクによる仮想ホスト junction の作成	573
仮想ホスト junction のサーバー・タスク・コマンド	574
仮想ホスト junction の作成	576
仮想ホスト junction へのサーバーの追加	582

第 9 部 シングル・サインオン・ソリューション 585

第 31 章 junction を介したシングル・サインオン・ソリューション 587

Tivoli Federated Identity Manager を使用したシングル・サインオン	587
Tivoli Federated Identity Manager との接続用の GSKit 構成	589
Kerberos 資格情報の使用	589
HTTP BA ヘッダーを使用したシングル・サインオン	590
シングル・サインオン (SSO) の概念	591
HTTP BA ヘッダー内のクライアント識別	591
クライアント識別と汎用パスワード	592
元のクライアント BA ヘッダー情報の転送	593
クライアント BA ヘッダー情報の除去	594
GSO からのユーザー名とパスワード	595
junction を介したクライアント識別情報	595
HTTP ヘッダーで提供される ID 情報	596
HTTP ヘッダー内のクライアント識別 (-c)	596
HTTP ヘッダーのクライアント IP アドレス (-r)	599
WebSEAL 生成の HTTP ヘッダーのサイズの制限	600
グローバル・サインオン (GSO)	600
グローバル・サインオンの概要	600
認証情報のマッピング	601
GSO 使用可能 WebSEAL junction の構成	602
GSO キャッシュの構成	603
IBM WebSphere へのシングル・サインオン (LTPA)	604
LTPA の概要	604
LTPA junction の構成	605
LTPA キャッシュの構成	606
LTPA シングル・サインオンに関する技術上の注意点	607
フォーム・シングル・サインオン認証	607
フォーム・シングル・サインオンの概念	608
フォーム・シングル・サインオン・プロセスのフロー	608
アプリケーション・サポートのための要件	610
フォーム・シングル・サインオン用の構成ファイルの作成	611
フォーム・シングル・サインオンを使用可能にする方法	615
フォーム・シングル・サインオンの例	615

第 32 章 クロスドメイン・シングル・サインオン 617

クロスドメイン・シングル・サインオンの概念	617
クロスドメイン・シングル・サインオンの概要	617
デフォルト認証トークンとカスタム認証トークン	618
拡張ユーザー属性および ID のマッピング	618
属性転送およびユーザー・マッピングの CDSO プロセス・フロー	618
クロスドメイン・シングル・サインオンの構成	621
CDSO 構成の要約	621
CDSO 条件と要件	622
CDSO 認証を使用可能および使用不可にする	623
認証トークン・データの暗号化	623
トークン・タイム・スタンプの構成	624
トークン・ラベル名の構成	625
CDSO HTML リンクの作成	625
トークン作成中の CDMF エラーの処理	626
認証トークンの保護	626
仮想ホストを使用したクロスドメイン・シングル・サインオンの使用	627
CDSO 用の拡張属性	627
トークンに追加する拡張属性	627
トークンから抽出する拡張属性	628
クロスドメイン・シングル・サインオン用のトークンの UTF-8 エンコード	629

第 33 章 LTPA シングル・サインオン 631

LTPA シングル・サインオンの概要	631
LTPA シングル・サインオンの構成	632
LTPA シングル・サインオンに関する技術上の注意点	632

第 34 章 e-community シングル・サインオン 635

e-community シングル・サインオンの概念	635
e-community の概要	635
e-community の機能と要件	637
e-community のプロセス・フロー	637
e-community Cookie	643
vouch-for 要求と応答	643
vouch-for トークン	644
e-community シングル・サインオンの構成	645
e-community 構成の要約	645
e-community の条件と要件	646
e-community 認証を使用可能および使用不可にする	647
e-community 名の指定	648
「保証」トークンの暗号化	648
vouch-for トークン・ラベル名の構成	649
マスター認証サーバー (MAS) の指定	650
vouch-for URL の指定	651
トークンおよび ec-Cookie の存続時間値の構成	652
トークン作成中の CDMF エラーの処理	652
非認証アクセスの使用可能化	653
vouch-for トークンの生成機能の制限	653

認証失敗時の動作の構成	654
pkmslogout-nomas を使用したログアウト	654
仮想ホストでの e-community の使用	654
ECSO 用の拡張属性	655
トークンに追加する拡張属性	655
トークンから抽出する拡張属性	656
e-community シングル・サインオン用のトークンの UTF-8 エンコード	657

第 35 章 シングル・サインオフ 659

シングル・サインオフ機能の概要	659
シングル・サインオフの構成	660
シングル・サインオフ要求および応答の仕様	660

第 10 部 デプロイメント 661

第 36 章 WebSEAL インスタンスのデプロイメント 663

WebSEAL インスタンスの構成の概要	663
WebSEAL インスタンスの構成計画	663
WebSEAL インスタンス構成値の例	665
各 WebSEAL インスタンス用の固有の構成ファイル	666
WebSEAL インスタンスの構成タスク	666
WebSEAL インスタンスの追加	666
WebSEAL インスタンスの除去	667
ロード・バランシング環境	668
フロントエンド WebSEAL サーバーの複製	668
login_success 応答のコントロール	669

第 37 章 アプリケーションの統合 671

バックエンド・サーバー・サイド・アプリケーションのサポート	671
標準 junction の推奨構成例	672
-v を使用した完全な Host ヘッダー情報	672
絶対 URL の標準フィルター操作	672
カスタム・パーソナライゼーション・サービス	673
パーソナライゼーション・サービスの概念	673
パーソナライゼーション・サービスのための WebSEAL の構成	674
パーソナライゼーション・サービスの例	674
バックエンド・サーバーのユーザー・セッション管理	675
ユーザー・セッション管理の概念	675
ユーザー・セッション ID 管理の使用可能化	676
HTTP ヘッダーへのユーザー・セッション・データの挿入	677
ユーザー・セッションの終了	679
バックエンド・サーバー用のユーザー・イベント	682

第 38 章 動的 URL 685

動的 URL に対するアクセス・コントロール	685
動的 URL のコンポーネント	685

動的 URL に対するアクセス・コントロール:	
dynurl.conf	685
照会ストリング・フォーマットへの POST 本体	
動的データの変換	686
動的 URL への ACL および POP オブジェクト	
のマッピング	687
文字エンコードと照会ストリング検証	688
動的 URL 用の WebSEAL の更新	689
オブジェクト・スペース内の動的 URL の解決	689
POST 要求に対する制限の構成	690
動的 URL の要約および技術上の注意点	691
動的 URL の例: Travel Kingdom 社の場合	692
アプリケーション	693
インターフェース	693
セキュリティー・ポリシー	694
セキュア・クライアント	694
アクセス制御	695
結論	695

第 39 章 Internet Content Adaptation Protocol (ICAP) のサポート 697

WebSEAL との ICAP の統合 - ワークフロー	698
機能の有効範囲	699
WebSEAL 内での ICAP サポートの構成	699

第 11 部 付録 701

付録 A. 構成ファイルを変更するためのガイドライン 703

一般ガイドライン	703
デフォルト値	703
ストリング	704
定義済みストリング	704
ファイル名	704
整数	705
ブール値	705

付録 B. コマンド・リファレンス 707

構文ステートメントの読み方	708
help.	709
server list	710
server task add	711
server task cache flush all	714
server task cluster restart	716
server task create	717
server task delete	726
server task dynurl update	727
server task help	728
server task jmt	730
server task list	731
server task offline	733
server task online	735
server task refresh all_sessions	737
server task reload	738
server task remove	739
server task server restart	741
server task show	742
server task server sync	743
server task terminate all_sessions	744
server task terminate session	746
server task throttle	747
server task virtualhost add	749
server task virtualhost create	752
server task virtualhost delete	760
server task virtualhost list.	761
server task virtualhost offline	763
server task virtualhost online.	765
server task virtualhost remove	767
server task virtualhost show	770
server task virtualhost throttle	772

特記事項. 775

索引 779



1. WebSEAL によるリソースの保護	5	30. セキュア SSL (HTTPS) junction	424
2. 保護オブジェクト・スペース	9	31. プロキシ junction の例	447
3. ACL ポリシー	10	32. WebSEAL-to-WebSEAL junction シナリオ	447
4. 明示的および継承されたポリシー	11	33. ステートフル junction でのバックエンド・サーバー UUID の使用	450
5. Web スペースの保護	13	34. 異なる UUID	451
6. junction による WebSEAL とバックエンド・リソースの接続	16	35. ステートフル junction のためのバックエンド・サーバー UUID の指定	452
7. WebSEAL junction による統一された Web スペースの実現	17	36. 仮想ホストの構成	471
8. Junction されたバックエンド・サーバー	19	37. 要約: バックエンド・リソースへの URL の変更	476
9. 統一された Web スペース	19	38. 絶対 URL のフィルター操作	488
10. 複製バックエンド・サーバー	20	39. junction Cookie を使用したサーバー相対 URL の処理	494
11. クラスタ・サポート	25	40. HTTP を介した WebSEAL RPC	525
12. HTTP および HTTPS 通信用のタイムアウト設定	51	41. 仮想ホスト junction シナリオ 1	557
13. 認証プロセス・フロー	151	42. 仮想ホスト junction シナリオ 2	562
14. 外部 Kerberos 認証	175	43. 仮想ホスト junction シナリオ 3	569
15. MPA ゲートウェイを介した通信	182	44. 複数回のログイン	591
16. ユーザー切り替え中のアドミニストレーターおよびユーザー・キャッシュ・データのスワッピング	187	45. バックエンド・アプリケーション・サーバーへの認証情報の提供	592
17. WebSEAL 要求キャッシング・プロセス・フローの例	235	46. 識別と「ダミー」パスワードが含まれる BA ヘッダー	593
18. 外部認証インターフェースのプロセス・フロー	264	47. WebSEAL による元のクライアント識別情報の転送	594
19. WebSEAL セッション・キャッシュ	287	48. クライアント BA ヘッダー情報の除去	594
20. セッション・キャッシュ構成ファイル・エントリー	294	49. グローバル・サインオン・メカニズム	601
21. 複製 WebSEAL サーバーのフェイルオーバー	304	50. フォーム・シングル・サインオン・プロセスのフロー	609
22. Microsoft SharePoint サーバーとの WebSEAL セッションの共用	343	51. CDMF を使用したクロスドメイン・シングル・サインオン・プロセス	620
23. 複製 WebSEAL サーバーのフェイルオーバー	350	52. e-community のモデル	636
24. WebSEAL/SMS のプロセス・フロー	352	53. e-community のプロセス・フローの構成の例	639
25. 単一 WebSEAL サーバーの Junction 構成	368	54. セッション管理	676
26. レプリカ・セット構成	369	55. すべての userA セッションの終了	682
27. OAuth EAS の論理フロー	398	56. 要求 URL の照会ストリングでのデータの引き渡し	685
28. 鍵格納ファイルの管理構成	403	57. 動的 URL に対する許可	688
29. 非セキュアの TCP (HTTP) junction	423		

表

1. アプライアンスがサポートしない WebSEAL の機能	6	22. 認証強度用にサポートされている認証方式	203
2. WebSEAL インスタンス管理	21	23. 認証強度レベルの整数値の例	207
3. サポートされているワイルドカード・マッチング文字	77	24. ネットワーク範囲を指定するネットマスクの使用 (IPv4)	208
4. URL マクロおよび 非 URL マクロでエンコードされる文字	91	25. ネットワーク範囲を指定するネットマスクの使用 (IPv6)	209
5. カスタム・ヘッダーの定義用マクロ	101	26. 外部認証インターフェースの構成	267
6. P3P デフォルト・ヘッダーの値	130	27. 外部認証アプリケーションに対する認証要求の例:	269
7. access エントリー用にサポートされている値	131	28. WebSEAL が提供する補足の資格情報データ	271
8. categories エントリー用にサポートされている値	132	29. PAC ヘッダー	275
9. disputes エントリー用にサポートされている値	133	30. ユーザー ID ヘッダー	275
10. remedies エントリー用にサポートされている値	133	31. セッション ID ヘッダー	276
11. non-identifiable エントリー用にサポートされている値	133	32. 共通ヘッダー	276
12. purpose エントリー用にサポートされている値	134	33. フェイルオーバー Cookie 用にサポートされているプロトコル	313
13. opt-in または opt-out ポリシー用にサポートされている値	135	34. ローカル・タイプの junction オプション	426
14. recipient エントリー用にサポートされている値	135	35. 戻りコード	436
15. Opt-in ポリシー値	136	36. フィルター対象のエンコード・タイプ	481
16. retention エントリー用にサポートされている値	136	37. 基本エレメント	510
17. ランタイム・セキュリティ・サービス EAS アクセス決定	139	38. XSLT テンプレート・ファイル	512
18. 基本認証の構成	160	39. リモート・タイプの仮想ホスト junction のオプション	552
19. フォーム認証の構成	162	40. ローカル・タイプの仮想ホスト junction のオプション	554
20. 証明書認証の構成	168	41. 追加インターフェース定義に有効なプロパティおよび値	559
21. LTPA 認証の構成	178	42. Tivoli Federated Identity Manager trust チェーンの構成要件	588
		43. WebSEAL インスタンスを追加するためのワークシート	666

本書について

「*IBM Security Web Gateway Appliance: Web リバース・プロキシの構成ガイド*」へようこそ。

IBM Security Access Manager for Web (旧名: IBM Tivoli Access Manager for e-business) は、広範な Web リソースおよびアプリケーション・リソースにセキュリティ・ポリシーを適用するための、ユーザー認証、許可、および Web シングル・サインオンのソリューションです。

IBM Security Web Gateway Appliance には Security Access Manager が含まれます。このアプライアンスでは Web リバース・プロキシ を使用して、Web アプリケーション・セッションのユーザー・アクセスおよび認証管理を行います。本書では、このプロキシを示す場合に *WebSEAL* という用語を使用します。

IBM Security Access Manager for Web WebSEAL は、Security Access Manager セキュア・ドメイン内の Web ベースのリソースのためのリソース・マネージャーです。WebSEAL は、ハイパフォーマンスかつマルチスレッド化された Web サーバーであり、で保護された Web オブジェクト・スペースに対して、きめ細かいセキュリティ・ポリシーを適用します。WebSEAL は、シングル・サインオン・ソリューションを提供し、バックエンド Web アプリケーション・サーバー・リソースをそのセキュリティ・ポリシーに取り込むことができます。

この構成ガイドでは、ご使用のセキュア Web ドメインのリソースを管理するための包括的な一連の手順と参照情報について説明します。また、広範囲にわたる WebSEAL 機能の背景と概念についても説明します。WebSEAL の構成に使用するすべてのスタンザの参照情報については、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

対象読者

本書は、Security Access Manager WebSEAL 環境の構成および保守を担当するシステム・アドミニストレーターを対象としています。

本書の読者は以下の事項についてよく理解しておく必要があります。

- PC および UNIX または Linux のオペレーティング・システム
- データベースのアーキテクチャーと概念
- セキュリティ管理
- HTTP、TCP/IP、ファイル転送プロトコル (FTP)、Telnet などのインターネット・プロトコル
- Lightweight Directory Access Protocol (LDAP) とディレクトリー・サービス
- サポートされるユーザー・レジストリー
- WebSphere® Application Server の管理
- 認証と許可

SSL (Secure Sockets Layer) 通信を使用可能にしようとしている場合は、SSL プロトコル、鍵交換 (公開鍵と秘密鍵)、デジタル署名、暗号アルゴリズム、および認証局 (CA) についての知識も必要です。

資料および用語集へのアクセス

このセクションには、以下が含まれています。

- 『IBM Security Access Manager for Web ライブラリー』の資料のリスト。
- xx ページの『オンライン資料』へのリンク。
- xxi ページの『IBM Terminology Web サイト』へのリンク。

IBM Security Access Manager for Web ライブラリー

以下の資料は IBM Security Access Manager for Web ライブラリーにあります。

- *IBM Security Access Manager for Web Quick Start Guide*、GI11-9333-01

主なインストールおよび構成タスクを要約した手順を説明します。

- *IBM Security Web Gateway Appliance - ハードウェア・オフファリング・クイック・スタート・ガイド*、SC22-5434-00

WebSEAL Hardware Appliance に接続し、初期構成を実行するプロセスの手順を示しています。

- *IBM Security Web Gateway Appliance - 仮想オフファリング・クイック・スタート・ガイド*

WebSEAL 仮想アプライアンスの接続および初期構成の実行のプロセスをユーザーに示します。

- *IBM Security Access Manager for Web インストール・ガイド*、GC23-6502-02

Security Access Manager のインストールおよび構成方法の説明があります。

- *IBM Security Access Manager for Web アップグレード・ガイド*、SC23-6503-02

バージョン 6.0 または 6.1.x からバージョン 7.0 にアップグレードするための情報をユーザーに提供します。

- *IBM Security Access Manager for Web 管理ガイド*、SC23-6504-02

Security Access Manager の概念と使用手順についての説明があります。Web Portal Manager インターフェースから、および **pdadmin** ユーティリティーを使用したタスクの実行方法が記載されています。

- *IBM Security Access Manager for Web WebSEAL 管理ガイド*、SC23-6505-02

WebSEAL を使用してユーザーのセキュア Web ドメインのリソースを管理する場合の背景資料、管理手順、および参照情報が記載されています。

- *IBM Security Access Manager for Web Plug-in for Web Servers 管理ガイド*、SC23-6507-02

Web サーバー・プラグインを使用して Web ドメインを保護する際の手順および参照情報が記載されています。

- *IBM Security Access Manager for Web 共用セッション管理 管理ガイド*、SC23-6509-02

Session Management Server の管理時の考慮事項および操作手順について説明しています。

- *IBM Security Access Manager for Web 共用セッション管理 デプロイメント・ガイド*、SC22-5431-00

Session Management Server のデプロイメント時の考慮事項について説明しています。

- *IBM Security Web Gateway Appliance 管理ガイド*、SC22-5432-00

WebSEAL Appliance の管理手順およびテクニカル・リファレンス情報を提供します。

- *IBM Security Web Gateway Appliance Web リバース・プロキシの構成ガイド*、SA88-5104-00

WebSEAL Appliance の構成手順およびテクニカル・リファレンス情報を提供します。

- *IBM Security Web Gateway Appliance Web リバース・プロキシ・スタンザ・リファレンス*、SC27-4442-00

IBM® Security Web Gateway Appliance Web リバース・プロキシのすべてのスタンザの参照情報が記載されています。

- *IBM Security Access Manager for Web WebSEAL 構成スタンザ・リファレンス*、SC27-4443-00

WebSEAL Appliance のすべてのスタンザの参照情報を提供します。

- *IBM Global Security Kit: CapiCmd Users Guide*、SC22-5459-00

鍵データベース、公開鍵と秘密鍵のペア、および認証要求の作成についての説明を示します。

- *IBM Security Access Manager for Web Auditing Guide*、SC23-6511-02

ネイティブ Security Access Manager アプローチおよび Common Auditing and Reporting Service (共通監査報告サービス) を使用した監査イベントの構成および管理に関する情報があります。また、Common Auditing and Reporting Service (共通監査報告サービス) のインストールおよび構成についての情報もあります。このサービスは、運用レポートの生成および表示に使用します。

- *IBM Security Access Manager for Web Command Reference*、SC23-6512-02

Security Access Manager で提供されるコマンド、ユーティリティ、およびスクリプトに関する参照情報が記載されています。

- *IBM Security Access Manager for Web Administration C API Developer Reference*、SC23-6513-02

管理 API の C 言語インプリメンテーションを使用して、アプリケーションが Security Access Manager の管理タスクを実行できるようにするための参照情報が記載されています。

- *IBM Security Access Manager for Web Administration Java Classes Developer Reference*、SC23-6514-02

管理 API の Java™ 言語インプリメンテーションを使用して、アプリケーションが Security Access Manager の管理タスクを実行できるようにするための参照情報が記載されています。

- *IBM Security Access Manager for Web Authorization C API Developer Reference*、SC23-6515-02

許可 API の C 言語インプリメンテーションを使用して、アプリケーションが Security Access Manager セキュリティーを使用できるようにするための参照情報が記載されています。

- *IBM Security Access Manager for Web Authorization Java Classes Developer Reference*、SC23-6516-02

許可 API の Java 言語インプリメンテーションを使用して、アプリケーションが Security Access Manager セキュリティーを使用できるようにするための参照情報が記載されています。

- *IBM Security Access Manager for Web Web Security Developer Reference*、SC23-6517-02

認証モジュールを開発するためのプログラミングおよび参照の情報が記載されています。

- *IBM Security Access Manager for Web Error Message Reference*、GI11-8157-02

メッセージおよび戻りコードについての説明および修正アクションが記載されています。

- *IBM Security Access Manager for Web Troubleshooting Guide*、GC27-2717-01

問題判別についての説明が記載されています。

- *IBM Security Access Manager for Web Performance Tuning Guide*、SC23-6518-02

ユーザー・レジストリーとして IBM Tivoli Directory Server を使用する Security Access Manager からなる環境の、パフォーマンス・チューニングについての説明があります。

オンライン資料

IBM では、製品のリリース時および資料の更新時に、以下の場所に製品資料を掲載しています。

IBM Security Access Manager for Web インフォメーション・センター

http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome.html サイトでは、この製品のインフォメーション・センターのウェルカム・ページが表示されます。

IBM Publications Center

このサイト (<http://www-05.ibm.com/e-business/linkweb/publications/servlet/pbi.wss>) には、必要なすべての IBM 資料を見つけるのに役立つカスタマイズ検索機能が用意されています。

IBM Terminology Web サイト

IBM Terminology Web サイトは、製品ライブラリーの用語を 1 つのロケーションに統合したものです。Terminology Web サイト (<http://www.ibm.com/software/globalization/terminology>) にアクセスできます。

関連資料

このセクションでは、Security Access Manager ソリューションに関連および含まれる IBM 製品のリストを示します。

注: 以下のミドルウェア製品は、IBM Security Web Gateway Appliance のパッケージに含まれていません。

IBM Global Security Kit

Security Access Manager では、Global Security Kit (GSKit) バージョン 8.0.x を使用してデータ暗号化が行われます。GSKit は、*IBM Security Access Manager for Web* バージョン 7.0 の特定プラットフォーム向けの製品イメージまたは DVD に収録されています。

GSKit バージョン 8 には、鍵管理のためのコマンド行ツールである GSKCapiCmd (`gsk8capiCmd_64`) が同梱されています。

GSKit バージョン 8 では、鍵管理ユーティリティの iKeyman (`gskikm.jar`) が同梱されなくなりました。iKeyman は、IBM Java バージョン 6 以降と共にパッケージされており、現在はネイティブ GSKit ランタイムに依存しない Pure Java アプリケーションです。製品に組み込まれている `java/jre/lib/gskikm.jar` ライブラリーは、移動または削除しないでください。

「*IBM Developer Kit and Runtime Environment, Java Technology Edition, Version 6 and 7, iKeyman User's Guide for version 8.0*」は、Security Access Manager インフォメーション・センターで入手可能です。この資料は、以下から直接入手することもできます。

<http://download.boulder.ibm.com/ibmdl/pub/software/dw/jdk/security/60/iKeyman.8.User.Guide.pdf>

注:

GSKit バージョン 8 には、セキュリティ問題を修正するために必要なトランスポート層セキュリティの実装に対する重要な変更が含まれています。

GSKit バージョン 8 の変更は、Internet Engineering Task Force (IETF) Request for Comments (RFC) の要件に準拠しています。ただし、以前のバージョンの GSKit とは互換性がありません。GSKit を使用する Security Access Manager と通信するコンポーネントは、GSKit バージョン 7.0.4.42、8.0.14.26、またはそれ以降を使用するようにアップグレードする必要があります。それ以外の場合は、通信の問題が発生することがあります。

IBM Tivoli Directory Server

IBM Tivoli Directory Server バージョン 6.3 FP17 (6.3.0.17-ISS-ITDS-FP0017) は、*IBM Security Access Manager for Web* バージョン 7.0 の特定プラットフォーム向けの製品イメージまたは DVD に収録されています。

Tivoli Directory Server について詳しくは、以下のサイトをご覧ください。

<http://www.ibm.com/software/tivoli/products/directory-server/>

IBM Tivoli Directory Integrator

IBM Tivoli Directory Integrator バージョン 7.1.1 は、*IBM Tivoli Directory Integrator Identity Edition V 7.1.1 for Multiplatform* の特定プラットフォーム向けの製品イメージまたは DVD に収録されています。

IBM Tivoli Directory Integrator について詳しくは、以下のサイトをご覧ください。

<http://www.ibm.com/software/tivoli/products/directory-integrator/>

IBM DB2 Universal Database™

IBM DB2 Universal Database Enterprise Server Edition バージョン 9.7 FP4 は、*IBM Security Access Manager for Web* バージョン 7.0 の特定プラットフォーム向けの製品イメージまたは DVD に収録されています。DB2® は、Tivoli Directory Server ソフトウェアと共にインストールすることも、スタンドアロン製品としてインストールすることもできます。DB2 は、Security Access Manager 用のユーザー・レジストリーとして、Tivoli Directory Server または z/OS® LDAP サーバーを使用する場合に必要です。z/OS LDAP サーバーでは、DB2 を別途ご購入いただく必要があります。

DB2 について詳しくは、以下のサイトをご覧ください。

<http://www.ibm.com/software/data/db2>

IBM WebSphere 製品

WebSphere Application Server Network Deployment バージョン 8.0 および WebSphere eXtreme Scale バージョン 8.5.0.1 のインストール・パッケージは、Security Access Manager バージョン 7.0 に同梱されています。WebSphere eXtreme Scale は、Session Management Server (SMS) コンポーネントを使用する場合にのみ必要です。

WebSphere Application Server は、以下のアプリケーションのサポートを可能にします。

- Web Portal Manager インターフェース (Security Access Manager を管理します)。
- Web Administration Tool (Tivoli Directory Server を管理します)。
- Common Auditing and Reporting Service (共通監査報告サービス) (監査イベントの処理およびレポート作成を行います)。
- Session Management Server (Web セキュリティー・サーバー環境で共有セッションを管理します)。

- 属性検索サービス

WebSphere Application Server について詳しくは、以下のサイトをご覧ください。

<http://www.ibm.com/software/webservers/appserv/was/library/>

アクセシビリティ

アクセシビリティ機能は、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に使用できるようにサポートします。これらの製品を使用することにより、インターフェースを耳で聴いて確認したり、ナビゲートしたりするための支援テクノロジーをご利用いただけます。また、グラフィカル・ユーザー・インターフェースのすべての機能は、マウスを使用しなくてもキーボードから操作できるようになっています。

IBM のアクセシビリティに対する取り組みについて詳しくは、IBM Accessibility Center を参照してください。

技術研修

以下は英語のみの対応となります。技術研修の情報については、IBM Education Web サイト (<http://www.ibm.com/software/tivoli/education>) を参照してください。

サポート情報

IBM サポートは、コード関連の問題、およびインストールまたは使用方法に関する短時間の定型質問に対する支援を提供します。IBM ソフトウェア・サポート・サイトには、<http://www.ibm.com/software/support/probsub.html> から直接アクセスできます。

「*IBM Security Access Manager for Web Troubleshooting Guide*」には、以下についての詳細が記載されています。

- IBM サポートに連絡する前に収集する情報。
- IBM サポートに連絡するためのさまざまな方法。
- IBM Support Assistant の使用方法。
- 問題をお客様自身で特定し、修正するために必要な説明および問題判別リソース。

注: 製品インフォメーション・センターの「コミュニティおよびサポート」タブに、追加のサポート・リソースが示されています。

第 1 部 管理

第 1 章 IBM Security Access Manager for Web WebSEAL の概要

IBM Security Access Manager for Web (Security Access Manager) は、分散アプリケーション用の、堅固でセキュアな集中ポリシー管理ソリューションです。

IBM Security Access Manager for Web WebSEAL は、ハイパフォーマンスかつマルチスレッド化された Web サーバーであり、Security Access Manager で保護された Web オブジェクト・スペースに対して、きめ細かいセキュリティ・ポリシーを適用します。WebSEAL は、シングル・サインオン・ソリューションを提供し、バックエンド Web アプリケーション・サーバー・リソースをそのセキュリティ・ポリシーに取り込むことができます。

この章では、WebSEAL サーバーの主な機能を紹介します。

この章で扱うトピックは以下のとおりです。

- 『概要』
- 4 ページの『WebSEAL の概要』
- 5 ページの『IBM Security Web Gateway Appliance』
- 5 ページの『アプライアンスでの WebSEAL の機能』
- 7 ページの『セキュリティ・モデル』
- 12 ページの『Web スペースの保護』
- 13 ページの『セキュリティ・ポリシー計画およびインプリメンテーション』
- 15 ページの『WebSEAL 認証』
- 16 ページの『標準 WebSEAL Junction』
- 18 ページの『Web スペース・スケーラビリティ』

概要

IBM Security Access Manager for Web は、地理的に分散したイントラネットおよびエクストラネット上でリソースのエンドツーエンド保護を提供する、完全とも言える許可/ネットワーク・セキュリティ・ポリシー管理ソリューションです。

IBM Security Access Manager for Web は、その最新のセキュリティ・ポリシー管理機能に加えて、認証、許可、データ・セキュリティ、および集中リソース管理機能も提供します。Security Access Manager を標準のインターネット・ベースのアプリケーションと組み合わせて使用することで、非常にセキュアで管理の行き届いたイントラネットを構築できます。

Security Access Manager のコア機能には次のものがあります。

- 認証フレームワーク

Security Access Manager には、広範囲をカバーするオーセンティケーターが組み込まれているほか、外部オーセンティケーターもサポートしています。

- 許可フレームワーク

Security Access Manager 許可サービスは、Security Access Manager 許可 API を介してアクセスされ、セキュア・ドメイン内にある保護リソースに対する要求を許可するか拒否するかを決定する機能を提供します。

Security Access Manager を使用することで、私設の社内ネットワーク・ベースのリソースへのアクセスを安全に管理できると同時に、公開のインターネットの広範な接続性と使いやすさをさらに高めることができます。Security Access Manager は、企業のファイアウォール・システムと組み合わせて使用することで、エンタープライズ・イントラネットを無許可アクセスおよび侵入から完全に保護することができます。

WebSEAL の概要

IBM Security Access Manager for Web WebSEAL は、Web ベースの情報とリソースを管理および保護するためのリソース・マネージャーです。

WebSEAL はハイパフォーマンスかつマルチスレッド化された Web サーバーであり、Security Access Manager の保護 Web オブジェクト・スペース内のリソースに対してきめ細かいセキュリティ・ポリシーを適用します。WebSEAL は、シングル・サインオン・ソリューションを提供し、バックエンド Web アプリケーション・サーバー・リソースをそのセキュリティ・ポリシーに取り込むことができます。

WebSEAL は、通常は逆 Web プロキシとして働き、Web ブラウザーから HTTP/HTTPS 要求を受信し、自身が所有する Web サーバーまたは junction (接合) されたバックエンド Web アプリケーション・サーバーからコンテンツを配布します。WebSEAL を通過した要求は、Security Access Manager 許可サービスによって評価され、要求されたリソースにアクセスする権限がそのユーザーにあるかどうか が判別されます。

WebSEAL は次のような機能を提供します。

- 複数の認証方式をサポートします。

組み込みアーキテクチャーとプラグイン・アーキテクチャーによって、各種認証メカニズムを柔軟にサポートできます。

- Security Access Manager 許可サービスを統合します。
- HTTP および HTTPS 要求を受け入れます。
- WebSEAL junction テクノロジーを介してバックエンド・サーバー・リソースを統合、保護します。

結合された保護オブジェクト・スペースの統一されたビューを提供します。

- ローカルおよびバックエンド・サーバー・リソースのためのきめ細かいアクセス・コントロールを管理します。

サポートされているリソースには、URL、URL ベースの正規表現、CGI プログラム、HTML ファイル、Java サーブレットおよび Java クラス・ファイルがあります。

- 逆 Web プロキシとして動作します。

WebSEAL は、クライアントに対しては Web サーバーとしての役を果し、保護している junction 先バックエンド・サーバーに対しては Web ブラウザーとしての役を果します。

- シングル・サインオン機能を提供します。

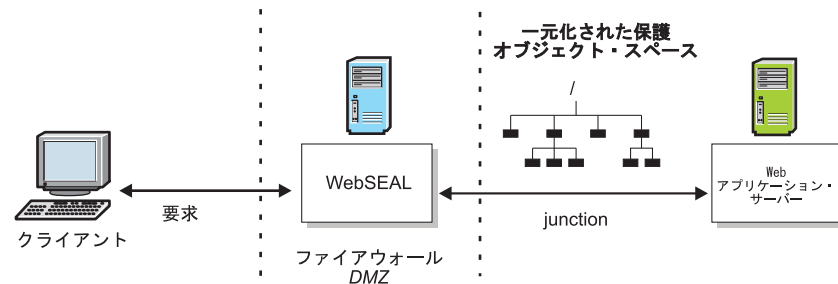


図 1. WebSEAL によるリソースの保護

IBM Security Web Gateway Appliance

IBM Security Web Gateway Appliance を使用することにより、ユーザーと Web アプリケーションのセッションのアクセス管理と認証管理を行い、脅威からアプリケーションを保護することができます。

このアプライアンスは、WebSEAL を使用することで、スケーラビリティを有効にし、無効なアクセスや脅威からアプリケーションを保護します。WebSEAL は、ユーザーとバックエンド・アプリケーション・サーバーの間に配置されます。

IBM Security Web Gateway Appliance には、次のオファリングがあります。

- 仮想オファリング
- ハードウェア・オファリング

アプライアンスは、管理目的のローカル管理インターフェース (LMI) を装備しています。LMI を使用してご使用の環境の WebSEAL インスタンスを管理できます。LMI の使用方法について詳しくは、「IBM Security Web Gateway Appliance 管理ガイド」を参照してください。

アプライアンスでの WebSEAL の機能

IBM Security Web Gateway Appliance Web リバース・プロキシには、WebSEAL の標準ソフトウェア・インストールによって提供される機能の大部分が組み込まれています。ただし、このセクションで詳しく説明するように、違いもいくつかあります。

表 1. アプライアンスがサポートしない WebSEAL の機能

機能	説明
CDAS や EAS を含むカスタム・ライブラリー	このアプライアンスはカスタム CDAS モジュールをサポートしません。結果としてアプライアンスは、以下の認証メカニズムをサポートしません。 • IP アドレス • HTTP ヘッダー • パスワード変更後 WebSEAL には、これらのメカニズムのための CDAS モジュールが組み込まれていません。 注: このアプライアンスは、IBM Security Identity Manager Password Synchronization Plug-in をサポートします。詳しくは、「<i>IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス</i>」の [itim] スタンザを参照してください。
RSA トークン	このアプライアンスはデフォルトで、RSA トークン認証をサポートしません。ただし、トークン認証用の外部認証インターフェース (EAI) をインプリメントすることができます。詳細については、263 ページの『第 13 章 外部認証インターフェース』を参照してください。
Kerberos (Windows デスクトップ・シングル・サインオン)	このアプライアンスは、内部で Kerberos 認証をサポートしません。ただし、Kerberos 認証を処理するように EAI を構成することができます。詳細については、174 ページの『Kerberos 認証』を参照してください。
ローカル junction	以下の制限がアプライアンス上のローカル junction サポートに適用されます。 • このアプライアンスは、WebSEAL インスタンスのローカル junction で単一の固定ファイル・システム・パスをサポートすることができます。 • アプライアンスのローカル junction では、CGI スクリプトを一切実行できません。
ハードウェア・ベースの暗号方式	このアプライアンスは、ハードウェア・ベースの暗号方式をサポートしません。ただし、ハードウェア・アプライアンスの i7-2600 プロセッサでは AES-NI をサポートしており、暗号操作を処理できます。
アプリケーション応答測定 (ARM)	WebSEAL ソフトウェアには、要求と応答の処理ストリーム全体でトランザクションをモニターする ARM が組み込まれています。このアプライアンスに ARM は組み込まれていません。

表 1. アプライアンスがサポートしない WebSEAL の機能 (続き)

機能	説明
Tivoli® Common Directory ログイン	Tivoli Common Directory ログイン機能では、IBM Security ソフトウェア・アプリケーションのすべてのログ・ファイルを共通のファイル・システム・ディレクトリーに格納します。アプライアンスはこの共通ログインをサポートしません。アプライアンスのログインは LMI を介して管理されます。
パイプまたは CARS への監査の送信	このアプライアンスは、パイプまたは CARS サーバーに監査レコードを直接に送信することができません。ただし、中間の ISAM Authorization Server を使用して、間接的に監査レコードを宛先まで送信できます。このアプライアンスは、リモート syslog に監査データを送信することもできます。
ARS (Web サービス)	IBM Security Access Manager for Web ARS Web サービスは、要求情報を外部の ARS サーバーに送信して許可を得ることができます。ARS はアプライアンスでは使用できません。

セキュリティ・モデル

このセクションは、以下のトピックで構成されています。

- 『セキュリティ・モデルの概念』
- 8 ページの『保護オブジェクト・スペース』
- 9 ページの『アクセス・コントロール・リスト (ACL) および保護オブジェクト・ポリシー (POP)』
- 10 ページの『アクセス・コントロール・リスト (ACL) ポリシー』
- 10 ページの『保護オブジェクト・ポリシー (POP)』
- 11 ページの『明示ポリシーと継承ポリシー』
- 11 ページの『ポリシー管理: Web Portal Manager』

セキュリティ・モデルの概念

Security Access Manager セキュア・ドメイン用のセキュリティ・ポリシーを管理および保持する、2 つの重要なセキュリティ構造があります。

- ユーザー・レジストリー

ユーザー・レジストリー (IBM TivoliDirectory Server や Microsoft Active Directory など) には、Security Access Manager 環境に参加できるすべてのユーザーおよびグループが含まれています。この環境をセキュア・ドメイン といいます。

- マスター許可 (ポリシー) データベース

許可データベースには、ドメイン (**保護オブジェクト・スペース**) 内のすべてのリソースの表記が含まれています。セキュリティー・アドミニストレーターは、保護を必要とするリソースにルールを適用することによって、任意のレベルのセキュリティーを指定することができます。これらのルールは、アクセス・コントロール・リスト (**ACL**) ポリシーおよび保護オブジェクト・ポリシー (**POP**) と呼ばれています。

認証のプロセスでは、ユーザーの識別を WebSEAL に対して証明します。ユーザーは、認証済み (Authenticated) または非認証 (Unauthenticated) としてセキュア・ドメインに参加できます。認証済みユーザーは、ユーザー・レジストリー内にアカウントを持っている必要があります。セキュリティー・アドミニストレーターは、ACL および POP を使用して、以下を行うことができます。

- 特定のリソースを非認証ユーザーがパブリックに使用できるようにする。
- その他のリソースを特定の認証済みユーザーのみが使用できるようにする。

ユーザーが正常に認証されると、WebSEAL は、資格情報 と呼ばれる一連の識別情報を作成します。資格情報には、ユーザー識別、グループ・メンバーシップ、および特殊な (「拡張」) セキュリティー属性が含まれます。

ユーザーがセキュア・ドメインに完全に参加するには、資格情報が必要です。Security Access Manager 許可サービスは、ユーザーの認証資格情報を、要求されたリソースに割り当てられているポリシー許可と比較することにより、セキュリティー・ポリシーを実施します。許可サービスは、結果として得られた勧告をリソース・マネージャー (例えば WebSEAL) に渡し、リソース・マネージャーが元の要求に対する応答を完了します。

保護オブジェクト・スペース

保護オブジェクト・スペースは、Security Access Manager セキュア・ドメインに属するリソースの階層的表現です。オブジェクト・スペースに表示される仮想オブジェクトは、以下に示したとおり、実際の物理ネットワーク・リソースを表しています。

- システム・リソース - 実際の物理ファイルまたはアプリケーション。
- 保護オブジェクト - 許可サービス、Web Portal Manager、およびその他の Security Access Manager 管理ユーティリティーが使用する、実際のシステム・リソースの論理表現。

ポリシーは、オブジェクト・スペース内のオブジェクトに付加してリソースに対する保護機能を提供するものです。許可サービスは、これらのポリシーに基づいて許可を決定します。

Security Access Manager ベースおよび Security Access Manager WebSEAL の結合インストールによって、以下のオブジェクト・スペース・カテゴリーが提供されます。

- **Web オブジェクト**

Web オブジェクトは、HTTP URL によりアドレッシングできるリソースです。これには、静的 Web ページ、およびデータベース照会またはその他の何らかの

タイプのアプリケーションに変換される動的 URL が含まれます。WebSEAL サーバーは、Web オブジェクトの保護を担当します。

- **Security Access Manager 管理オブジェクト**

管理オブジェクトは、Web Portal Manager を介して実行できる管理アクティビティーを表します。管理オブジェクトは、ユーザーを定義し、セキュリティー・ポリシーを設定するために必要なタスクを表しています。Security Access Manager は、管理アクティビティーの代行をサポートしており、セキュリティー・ポリシーを設定するアドミニストレーター権限をオブジェクト・スペースのサブセットにのみ限定することができます。

- **ユーザー定義オブジェクト**

ユーザー定義オブジェクトは、ユーザーが定義するタスクまたはネットワーク・リソースのうち、Security Access Manager の許可 API を介して許可サービスにアクセスするアプリケーションによって保護されるものを表します。

- **許可ルール**

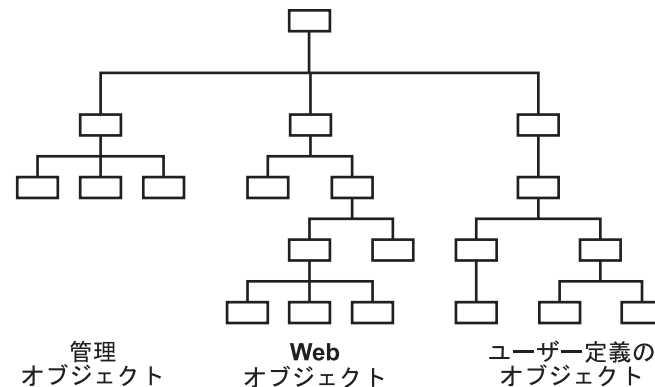


図 2. 保護オブジェクト・スペース

アクセス・コントロール・リスト (ACL) および保護オブジェクト・ポリシー (POP)

セキュリティー・アドミニストレーターは、Security Access Manager システム・リソースを保護するために、ACL ポリシーおよび POP ポリシーとして知られるルールを定義し、これらのポリシーを、保護オブジェクト・スペース内にあるシステム・リソースのオブジェクト表現に適用します。

Security Access Manager 許可サービスは、このようなオブジェクトに適用されるポリシーに基づき、許可の決定を行います。保護オブジェクトについて要求された操作が許可されると、該当リソースを受け持っているアプリケーションがその操作を実行します。

1 つのポリシーが、多数のオブジェクトの保護パラメーターを指示できます。ルールを変更すると、ACL または POP が付加されているすべてのオブジェクトが変更の影響を受けます。

アクセス・コントロール・リスト (ACL) ポリシー

アクセス・コントロール・リスト・ポリシー (ACL ポリシー) は、リソースに特定の操作を実行するために必要な条件を指定する一組のルール (許可) です。ACL ポリシー定義は、セキュア・ドメイン用に設定されるセキュリティ・ポリシーの重要なコンポーネントです。ACL ポリシーは、その他のすべてのポリシーと同様に、保護オブジェクト・スペース内に表示されるリソースに組織のセキュリティ要件を適用するために使用されます。

ACL ポリシーは、具体的には以下の事項を制御します。

1. リソースに対して実行できる操作は何か
2. それらの操作を実行できるのは誰か

1 つの ACL ポリシーは、ユーザー指定およびグループ指定とそれに固有の許可または権限を示す、1 つ以上のエントリーで構成されています。ACL には、非認証ユーザーに適用されるルールも含まれています。

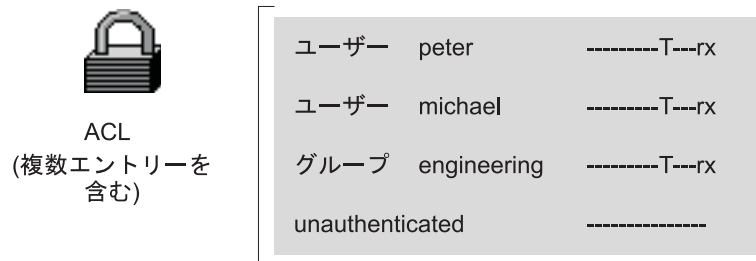


図3. ACL ポリシー

保護オブジェクト・ポリシー (POP)

ACL ポリシーは、保護オブジェクトにアクセスし、そのオブジェクトに対して何らかの操作を実行するための要求に、「はい」または「いいえ」で答えるための情報を伴った許可サービスを提供します。

保護オブジェクト・ポリシー (POP) には、要求に関する追加条件が含まれています。これらの追加条件は、許可サービスからの「はい」の ACL ポリシー決定と共に、Security Access Manager およびリソース・マネージャー (WebSEAL など) に戻されます。POP 条件を実行するのは、Security Access Manager およびリソース・マネージャーの責任です。

次の表は、POP の使用可能属性を示しています。

Security Access Manager が実行	
POP 属性	説明
名前	ポリシーの名前。これは、 pdadmin pop コマンドで <pop_name> 引数になります。
説明	ポリシーの記述テキスト。この属性は、 pop show コマンドの中に現れます。

Security Access Manager が実行	
POP 属性	説明
警告モード	アドミニストレーターに、ACL および POP ポリシーをテストする手段を提供します。
監査レベル	監査のタイプ (すべて、なし、成功したアクセス、拒否されたアクセス、エラー) を指定します。
時刻 (TOD) アクセス	保護オブジェクトに正常にアクセスするための日時の制限。

リソース・マネージャー (WebSEAL など) が実行	
POP 属性	説明
保護品質	データ保護の程度 (なし、保水性、プライバシー) を指定します。
IP Endpoint Authentication Method Policy	外部ネットワークのメンバーがアクセスするための認証要件を指定します。
文書キャッシュ・コントロール	特定文書の取り扱いのためのキャッシング指示を指定します。

明示ポリシーと継承ポリシー

ポリシーは、明示的に適用することも、継承することもできます。Security Access Manager の保護オブジェクト・スペースでは、ACL 属性および POP 属性の継承をサポートしています。継承はセキュリティー・アドミニストレーターにとって重要な管理機能です。アドミニストレーターが明示ポリシーを適用する必要があるのは、階層内で、ルールを変更することが必要になったポイントのみです。

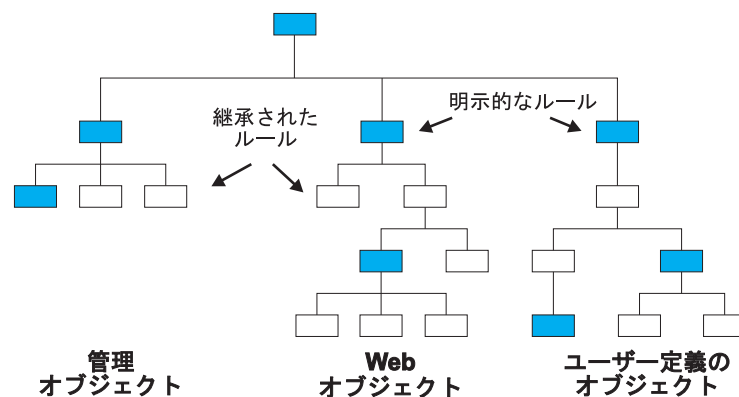


図4. 明示的および継承されたポリシー

ポリシー管理: Web Portal Manager

Web Portal Manager は、Security Access Manager セキュア・ドメイン内のセキュリティー・ポリシーを管理するために使用される Web ベースのグラフィカル・アプリ

リケーションです。**pdadmin** コマンド行ユーティリティーでは、Web Portal Manager と同じ管理機能に加えて、Web Portal Manager ではサポートされていないいくつかのコマンドも使用できます。

Web Portal Manager (または **pdadmin**) から、ユーザー・レジストリー、マスター許可ポリシー・データベース、および Security Access Manager サーバーを管理することができます。また、ユーザーやグループの追加と削除、およびネットワーク・オブジェクトへの ACL と POP の適用もできます。

Web スペースの保護

WebSEAL によってセキュア・ドメイン内のセキュリティを実施している場合は、ユーザーは各自の ID 証明を提示する必要があります。これに対して、Security Access Manager セキュリティー・ポリシーは、要求対象のリソースに対する操作を実行する許可がそのユーザーにあるかどうかを判別します。セキュア・ドメイン内の各 Web リソースへのアクセスは WebSEAL によって制御されるため、WebSEAL が認証および許可を要求することによって、包括的なネットワーク・セキュリティを実現することができます。

セキュリティ・システムでは、許可 (authorization) は認証 (authentication) と区別されます。許可は、認証されたユーザーに、セキュア・ドメイン内の特定リソースに対する操作を実行する権限があるかどうかを判別します。認証は、ユーザーの識別を検証することはできますが、そのユーザーが保護リソースに対する操作を実行する権限を持つかどうかに関しては関知しません。

Security Access Manager 許可モデルでは、許可ポリシーは、ユーザー認証に使用されるメカニズムとは別の、独立したものとしてインプリメントされます。ユーザーは、公開鍵と秘密鍵のペア、秘密鍵、またはユーザー定義のメカニズムのいずれかを使用して、自己の識別を認証することができます。

認証プロセスの一部に、ユーザーの識別を記述した資格情報の作成があります。許可サービスによって行われる認証の判断は、ユーザー資格情報に基づいています。

セキュア・ドメイン内のリソースは、そのドメイン用のセキュリティ・ポリシーから指示されるレベルの保護を受けることになります。セキュリティ・ポリシーは、セキュア・ドメインの正当な参加者と、保護されている各リソースの保護の程度を定義します。

許可プロセスは、以下の基本コンポーネントで構成されます。

- **リソース・マネージャー** は、許可が付与されたときに、要求された操作をインプリメントします。WebSEAL はリソース・マネージャーの 1 つです。

リソース・マネージャーのコンポーネントの 1 つに、要求を処理するために許可サービスに送る**ポリシー・エンフォースャー**があります。

注: 従来型のアプリケーションでは、ポリシー・エンフォースャーとリソース・マネージャーが 1 つのプロセスにまとめられています。この構造を持つ例として、WebSEAL およびサード・パーティーのアプリケーションがあります。

- **許可サービス**は、要求に関する意思決定アクションを実行します。

次の図は、許可プロセス全体を示しています。

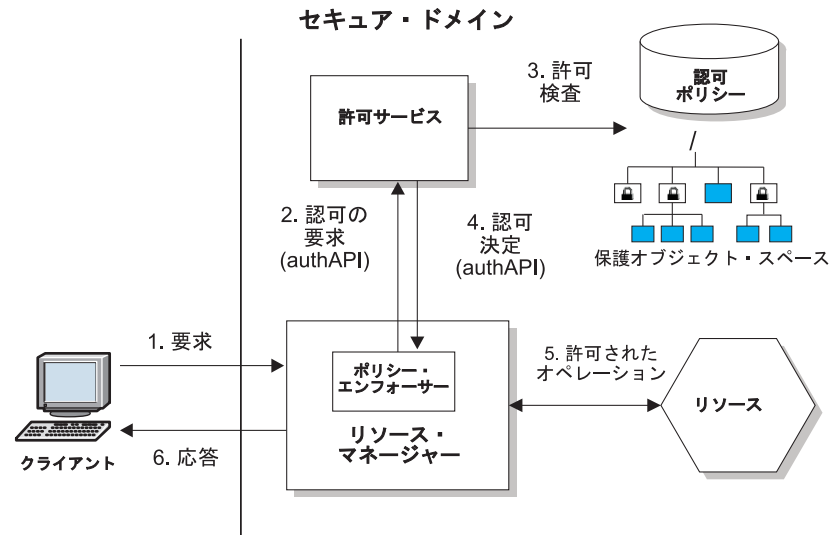


図5. Web スペースの保護

1. 認証済みユーザーからのリソース要求は、リソース・マネージャーに送信され、ポリシー・エンフォースャー・プロセスによってインターセプトされます。

リソース・マネージャーには、WebSEAL (HTTP、HTTPS アクセスの場合) またはサード・パーティーのアプリケーションを使用できます。

2. ポリシー・エンフォースャー・プロセスは、Security Access Manager 許可 API を使用して、許可決定のための許可サービスを呼び出します。
3. 許可サービスは、保護オブジェクト・スペース内のオブジェクトとして表現されているリソースに対して、許可検査を実施します。
 - a. 最初に Security Access Manager POP を検査します。
 - b. 次に、クライアントの資格情報に照らして、オブジェクトに付加された ACL ポリシーを検査します。
 - c. 最終的に、リソース・マネージャーが実行する POP が検査されます。
4. 要求を受け入れるか拒否するか決定が、リソース・マネージャーへの勧告 (ポリシー・エンフォースャーを使用) として戻されます。
5. 最終的に要求が承認されると、リソース・マネージャーはその要求を該当のリソース担当のアプリケーションに渡します。
6. ユーザーは、要求した操作の結果を受け取ります。

セキュリティー・ポリシー計画およびインプリメンテーション

企業の Web リソース・セキュリティー・ポリシーでは、以下を指定します。

- 保護を必要とする Web リソース
- 保護のレベル

Security Access Manager は、これらの Web リソースを、保護オブジェクト・スペースと呼ばれる仮想表現を使用して表します。保護オブジェクト・スペースには、ユーザーのネットワーク内の実際の物理リソースを表すオブジェクトが入ります。

ユーザーは、保護を必要とするオブジェクトに、適切なセキュリティ・メカニズムを適用することによって、セキュリティ・ポリシーをインプリメントします。

セキュリティ・メカニズムには、以下のものがあります。

- **アクセス・コントロール・リスト (ACL) ポリシー**

ACL ポリシーは、アクセスを許可するかどうかを判断するためのユーザー・タイプを識別し、そのオブジェクトに対して許される操作を指定します。

- **保護オブジェクト・ポリシー (POP)**

POP は、保護オブジェクトへのアクセスを制御するその他の条件 (プライバシー、保全性、監査および時刻 (TOD) アクセスなど) を指定します。

- **拡張属性**

拡張属性は、オブジェクト、ACL、または POP に設定され、サード・パーティー・アプリケーション (外部許可サービスなど) によって読み取りまたは解釈が可能な追加の値です。

Security Access Manager のコア・コンポーネントは、Security Access Manager 許可サービスです。このサービスは、ユーザーの資格情報と、オブジェクトに設定されたアクセス・コントロールに基づいて、保護オブジェクト (リソース) へのアクセスを許可または拒否します。

セキュリティ・ポリシーを正常にインプリメントするには、各種のコンテンツ・タイプを論理的に編成し (『コンテンツ・タイプおよび保護レベル』を参照)、適切な ACL ポリシーおよび POP ポリシーを適用する必要があります。アクセス・コントロールの管理は非常に複雑になることがありますが、コンテンツ・タイプを慎重に分類することによって、容易に管理できるようになります。

コンテンツ・タイプおよび保護レベル

ユーザーは、Web スペースのセキュリティ・アドミニストレーターとして、どのタイプのユーザーがどのタイプのコンテンツにアクセスできるかということを正確に把握しておかなければなりません。一部のコンテンツは、厳重に保護する必要がありますが、特定のユーザーのみが使用できるようにしなければなりませんし、あるコンテンツは一般に公開することができます。セキュリティ・シナリオによって、保護要件が異なり、それに伴って異なる WebSEAL 構成が必要になります。

ユーザーは、以下のことを行う責任があります。

- ユーザーがご自分の Web コンテンツを知る。
- このコンテンツにアクセスするユーザーのタイプを識別する。
- このコンテンツを保護するために必要な WebSEAL 構成オプションの長所と弱点を理解する。

Web コンテンツの保護は、以下の大きな 3 つのカテゴリに分類されます。

1. **公開コンテンツ** - アクセスには保護を必要としません。
 - 非認証ユーザーは、HTTP を使用してリソースにアクセスできます。
 - 非認証資格情報は、リソースへのアクセス・コントロールに使用されます。
 - 基本 WebSEAL 構成要件は保護を提供します。
2. **公開コンテンツ** - アクセスにはプライバシー (暗号化) を必要とします。
 - 非認証ユーザーは、HTTPS を使用してリソースにアクセスできます。
 - 暗号化はアプリケーション・サーバーに必須で、これを使用して機密データを保護します (クレジット・カード番号やユーザー・アカウント情報など)。
 - 非認証資格情報は、リソースへのアクセス・コントロールに使用されます。
 - プライバシーを保証する WebSEAL 構成。
3. **非公開コンテンツ** - アクセスには認証を必要とします。
 - 認証クライアントは、HTTP または HTTPS を使用してリソースにアクセスできます。
 - アドミニストレーターによる暗号化の必要性を判別します。
 - 認証資格情報はリソースへのアクセス・コントロールに使用されます。ユーザーごとに Security Access Manager ユーザー・レジストリーに定義されたアカウントが必要です。
 - WebSEAL 構成は複雑なため、すべてのオプションを慎重に検討し、セキュリティ・ポリシーの影響を判別する必要があります。

WebSEAL 認証

認証とは、セキュア・ドメインにログインしようとする個別のプロセスまたはエンティティを識別する方式のことです。WebSEAL は、各ユーザーに対し、識別の証明を要求することによって、セキュア・ドメイン内で高度のセキュリティを実施できます。

WebSEAL 認証プロセスには以下の条件が適用されます。

- WebSEAL は、デフォルトとしていくつかの認証方式をサポートしており、また他の方式を使用するようにカスタマイズすることもできます。
- サーバーとクライアントの両方が認証を必要とする場合、交換は**相互認証**と呼ばれます。
- WebSEAL サーバー・プロセスは認証方式とは別個のものです。
- WebSEAL に対する認証が成功すると、Security Access Manager ユーザー ID が作成されます。
- WebSEAL は、この ID を使用して、そのユーザーの資格情報を作成します。
- 許可サービスは、要求されたりリソースごとのポリシーを管理する ACL 許可および POP 条件を評価した後に、この資格情報を使用して、保護オブジェクトに対するアクセスを許可したり拒否したりします。

認証に対するこの柔軟な方法によって、物理的なネットワーク・トポロジではなく、ビジネス要件に基づいたセキュリティ・ポリシーが可能になります。

WebSEAL 認証の概念に関する全概要については、149 ページの『第 7 章 認証の概要』を参照してください。

標準 WebSEAL Junction

Security Access Manager は、認証サービス、許可サービス、および管理サービスをネットワークに提供します。Web ベースのネットワークでは、バックエンド Web サーバーに配置した Web リソースとアプリケーションを、1 つ以上のフロントエンド WebSEAL サーバーによって統合および保護し、これらのフロントエンド WebSEAL サーバーが上記のようなサービスを提供するのが最善の形です。

WebSEAL サーバーとバックエンド Web アプリケーション・サーバーとの間の接続は、標準 WebSEAL *junction* と呼ばれています。WebSEAL *junction* とは、フロントエンド WebSEAL サーバーとバックエンド・サーバーの間の TCP/IP 接続のことです。

注: また、WebSEAL は仮想ホスト *Junction* と呼ばれる別のフォームの *junction* による仮想ホスティングもサポートします。

バックエンド・サーバーは、別の WebSEAL サーバーでもよいし、あるいはより一般的には、サード・パーティーのアプリケーション・サーバーでも構いません。バックエンド・サーバー Web スペースは、WebSEAL Web スペース内の特別に指定された *junction* (マウント) ポイントで、WebSEAL サーバーに「接続されて」います。

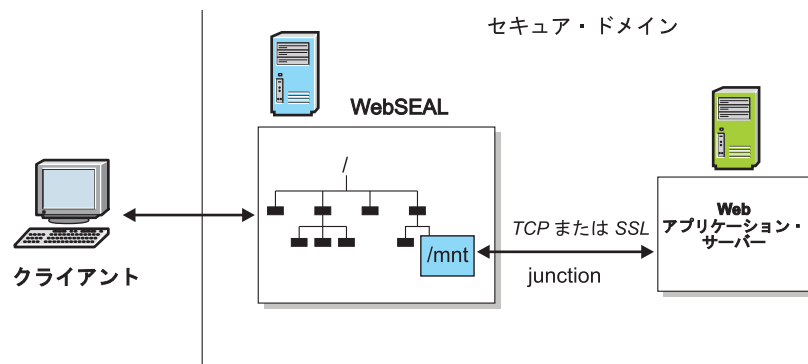


図 6. *junction* による WebSEAL とバックエンド・リソースの接続

Junction により、WebSEAL がバックエンド・サーバーに代わって、保護サービスを提供できます。WebSEAL は、すべての要求について、その要求をバックエンド・サーバーに渡す前に、認証検査と許可検査を実行できます。バックエンド・サーバーで、オブジェクトに対するきめ細かいアクセス・コントロールが必要な場合は、追加の構成ステップを実行して (query_contents CGI プログラムを使用)、Security Access Manager セキュリティー・サービスにサード・パーティー Web スペースを記述する必要があります。

Junction を使用することで、ロード・バランシング、高可用性、および状態管理機能を備えた、拡張が容易でセキュリティー機能のある環境を達成でき、そのための

すべての処理をクライアントには見えない形で実行することができます。アドミニストレータは、この集中管理による Web スペースから多大な利点を得ることができます。

WebSEAL junction によって、バックエンド・サーバーの Web スペースを WebSEAL サーバーの Web スペースに論理的に結合するという付加価値が得られます。連携サーバー間を junction することで、単一の、統一された、シームレスでユーザーに透過的な分散 Web スペースができ up します。

クライアントが Web リソースの物理的な場所を知る必要はまったくありません。WebSEAL は、論理 URL アドレスをバックエンド・サーバーが予期している物理アドレスに変換します。Web オブジェクトをサーバー間で移動でき、しかもそれによってクライアントがオブジェクトにアクセスする方法に影響が生じることはありません。

Web スペースが統一されていることにより、システム・アドミニストレータにとって、すべてのリソースの管理が単純化されます。管理上の利点としては、これに加えて、スケーラビリティ、ロード・バランシング、高可用性があります。

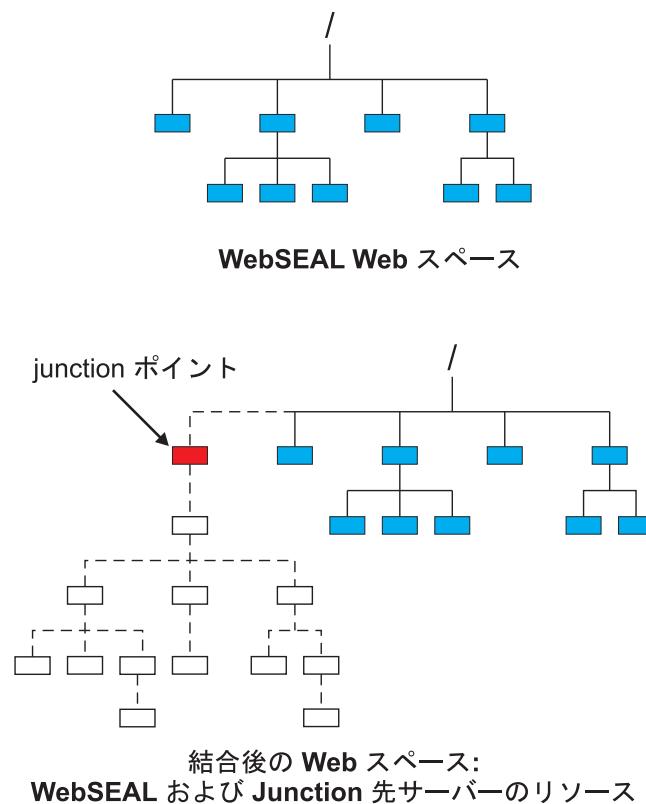


図 7. WebSEAL junction による統一された Web スペースの実現

ほとんどの商用の Web サーバーには、論理 Web オブジェクト・スペースを定義できる機能はありません。それどころか、アクセス・コントロールは、物理ファイルとディレクトリー構造に接続されます。WebSEAL junction では、標準的な Web サーバーによく見られる物理マシンとディレクトリー構造でなく、組織の構造を反映するオブジェクト・スペースを透過的に定義できます。

また、WebSEAL junction では、シングル・サインオン・ソリューションも作成できます。シングル・サインオン構成を使用すると、ユーザーは、1 回の初期ログインを使用するだけで、リソースの場所に関係なく、リソースにアクセスできます。バックエンド・サーバーからの後続のログイン要件は、すべてユーザーには透過的に処理されます。

WebSEAL junction は、Web サイトの拡張を容易にするための重要なツールです。Junction を使用して追加のサーバーを接続することにより、Web サイトで増加の一端をたどる要求に応えることができます。

Web スペース・スケーラビリティ

WebSEAL junction を使用すると、スケーラブルな Web スペースを作成することができます。Web スペースでの要求が増大するにつれて、サーバーを簡単に追加でき、サイトの能力を拡張することができます。

サーバーを追加する目的としては、次のような場合が挙げられます。

- コンテンツを追加して Web スペースを拡張するため
- 既存のコンテンツを重複させることによって、ロード・バランシング、フェイルオーバー機能、高可用性を確保するため

複製フロントエンド WebSEAL サーバー

バックエンド・サーバー用の junction サポートには、手始めとして少なくとも 1 台のフロントエンド WebSEAL サーバーが必要です。複製フロントエンド WebSEAL サーバーは、サイトに対する要求が混み合う期間にロード・バランシングの機能を提供します。ロード・バランシング・プロセスは、Cisco Local Director などのサード・パーティー・デバイスによって処理されます。

また、フロントエンドを複製すると、サイトにフェイルオーバー機能が提供されるため、何らかの理由でサーバーに障害が起ころうとしても、残りのレプリカ・サーバーによって引き続きサイトへのアクセスが得られます。優れたロード・バランシングとフェイルオーバー機能を実現することにより、サイトのユーザーに対して高可用性を提供することができます。

フロントエンド WebSEAL サーバーを複製する場合は、サーバーごとに、それぞれ Web スペースと junction データベースの正確なコピーを持っている必要があります。

認証のためのアカウント情報は、フロントエンド・サーバーとは別個のユーザー・レジストリーに入っています。

Junction されたバックエンド・サーバー

Web サイト・コンテンツに対するサービスは、WebSEAL サーバー自体、バックエンド・サーバー、またはその両方の組み合わせから提供することができます。バックエンド・サーバー用の WebSEAL junction サポートを使用することによって、コンテンツとリソースを追加し、Web サイトを拡張できます。

固有のバックエンド・サーバーはそれぞれ、別々の junction マウント・ポイントに junction されなければなりません。追加コンテンツに対する要求の増大に応じて、junction を使用してサーバーを追加できます。このシナリオは、サード・パーティーの Web サーバーに既に大きな投資をしているネットワークに適したソリューションとなります。

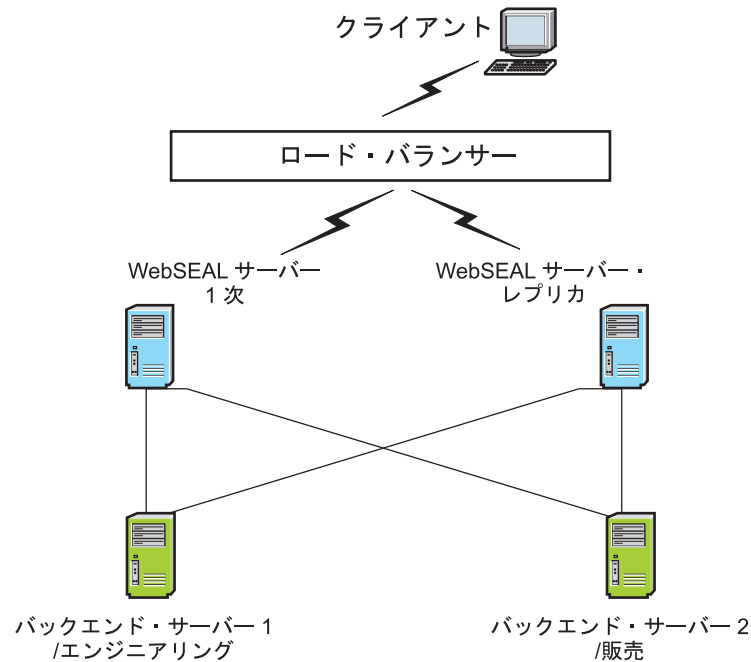


図 8. Junction されたバックエンド・サーバー

2 つのバックエンド・サーバーの結合された Web スペースは、ユーザーに対して透過的で、集中管理を行うことができます。

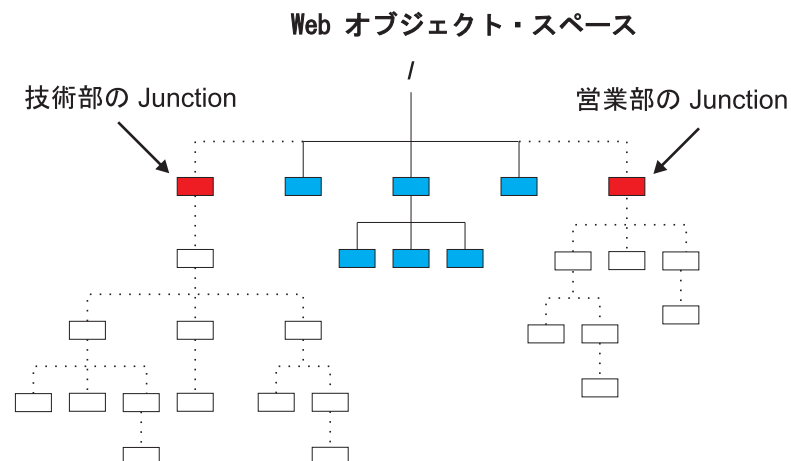


図 9. 統一された Web スペース

複製バックエンド・サーバー

スケーラビリティ機能をバックエンド・サーバー構成に適用して、バックエンド・サーバーを複製できます。複製フロントエンド・サーバーの場合と同じように、複製バックエンド・サーバーには、それぞれが相互にミラー・イメージとなる Web スペースが存在しなければなりません。

WebSEAL では、「一番すいている」スケジューリング・アルゴリズムを使用して、複製サーバー間のロード・バランシングを図ります。このアルゴリズムによって、各新規要求は、既に進行中の接続が最も少ないサーバーに送信されます。

WebSEAL はまた、サーバーがダウンした場合には正しくフェイルオーバーし、そのサーバーが再始動した後で、サーバーの再使用を開始します。

バックエンド・アプリケーションで複数のページに渡って状態を維持することが必要な場合は、ステートフル junction を使用して、各セッションが必ず同じバックエンド・サーバーに戻るようにすることができます。

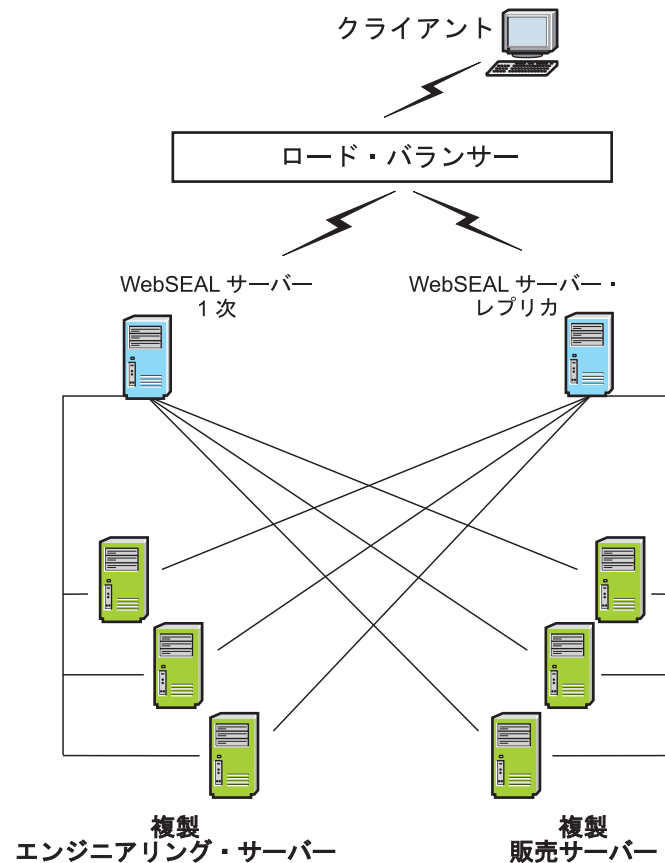


図 10. 複製バックエンド・サーバー

第 2 章 サーバー管理

この章では、WebSEAL サーバーを管理するためにユーザーが実行できる管理タスクについて説明します。

この章で扱うトピックは以下のとおりです。

- 『WebSEAL インスタンス管理』
- 22 ページの『複数のサーバー間での WebSEAL データの同期化』
- 26 ページの『WebSEAL 用リソースの監査およびロギング』
- 28 ページの『WebSEAL の問題判別リソース』

WebSEAL インスタンス管理

LMI の「リバース・プロキシ」管理ページを使用して、アプライアンス上の WebSEAL インスタンスを管理できます。

LMI の上部メニューで、「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「管理」>「リバース・プロキシ」を選択します。「リバース・プロキシ」管理ページが表示されます。

このページを使用して各 WebSEAL インスタンスを管理できます。選択可能なインスタンスのリストから、管理する WebSEAL インスタンスを選択します。WebSEAL インスタンスの管理で使用可能なオプションの詳細を以下の表に示します。

表 2. WebSEAL インスタンス管理

オプション	説明
新規作成	WebSEAL インスタンスを作成します。
編集	基本構成を行います。
削除	WebSEAL インスタンスを削除します。
開始	WebSEAL インスタンスを開始します。
停止	WebSEAL インスタンスを停止します。
再始動	WebSEAL インスタンスを再始動します。
最新表示	WebSEAL インスタンスおよび関連する詳細情報のリストを最新表示します。

表 2. WebSEAL インスタンス管理 (続き)

オプション	説明
管理	次の管理オプションのメニューから選択します。 構成 拡張構成オプションが用意されています。このメニューから直接に WebSEAL 構成ファイルを変更できます。 トラブルシューティング トレースおよび統計などのトラブルシューティング・リソースを含みます。 管理ルート WebSEAL ルート・ファイルにアクセスできます。 Junction の管理 junction の作成と管理が含まれます。 ログイン WebSEAL ログ・ファイルにアクセスできます。

複数のサーバー間での WebSEAL データの同期化

WebSEAL **server sync** コマンドを使用して、WebSEAL サーバーの構成を別の WebSEAL サーバーの構成と同期させることができます。

server sync コマンドは、他のサーバー・タスク・コマンドを実行することによりこのタスクを実行します。つまり、**cfgdb export** および **cfgdb import** コマンドにより WebSEAL 構成データを同期し、**jdb import** および **jdb export** コマンドにより Junction データベースを同期します。

注: 同期できるのは同じタイプのサーバーのみです。WebSEAL サーバーのタイプは次のいずれかです。

- Web Gateway Appliance で稼働する WebSEAL。
- 標準オペレーティング・システムで稼働する WebSEAL。

次のようなさまざまなタスクで以下のサーバー・タスク・コマンドのリストを使用することができます。

- 複製された WebSEAL サーバーを同じタイプの別の WebSEAL サーバーと同期させる。
- WebSEAL 環境を別の環境に (例えば、テスト環境から実稼働環境に) マイグレーションする。

server sync

提供された WebSEAL サーバーの構成を、現在の WebSEAL サーバーと同期するのに使用されます。**server sync** コマンドは、このリスト内の他のコマンドを呼び出して、完全な同期操作を実現します。同期できるデータには、構成エントリ、Junction データベース、および選択データ・ファイルなどがありますが、オブジェクト・スペースおよびポリシーは同期できません。

ん。同期される構成エントリーおよびデータ・ファイルは WebSEAL 構成ファイルでカスタマイズ可能です。詳しくは、743 ページの『server task server sync』を参照してください。

server restart

WebSEAL インスタンスを再始動するのに使用されます。詳しくは、741 ページの『server task server restart』を参照してください。

以下のリストでは、**server sync** コマンドの通信フローについて説明します。

1. **server sync** コマンドは管理コンソールから発行されます。
2. データ要求が WebSEAL サーバーから新規サーバー・タスク・コマンドとして発行されます。
3. ソース WebSEAL サーバーは同期させるデータを収集し、それをターゲット WebSEAL サーバーに送信します。
4. ターゲット WebSEAL サーバーは取得したデータを適用します。

データ要求は、**server sync** タスクを処理している WebSEAL サーバーから発行されます。データが 1 つの WebSEAL サーバーから他のサーバーへプルされ、その際 Security Access Manager **server task** フレームワークによって許可が自動的に適用されます。既存の通信チャネルを使用するので、WebSEAL サーバー用に他のポートを開く必要はありません。

さまざまな情報に対して複数の要求が行われる可能性があります。例えば、Junction データベースを同期させる場合、**jdb export** コマンドと **file cat** コマンドを使用して、Junction データベースを WebSEAL サーバー上の単一のファイルにエクスポートし、次にエクスポート済みのファイルに含まれるデータを取得できます。

サーバー間で複製される構成エントリーおよびファイルは、基礎となるスタンプ内のエントリーで定義されます。詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンプ・リファレンス*」の **[cfg-db-cmd:entries]** スタンプおよび **[cfg-db-cmd:files]** スタンプを参照してください。

jdb import コマンドのマッピング・ルールは **[jdb-cmd:replace]** スタンプに定義されます。これらのマッピング・ルールは、新規 Junction データベースのインポート前に Junction アーカイブ・ファイル内の各属性に適用されます。詳細は、834 ページの『**[jdb-cmd:replace]** スタンプ』を参照してください。

重要: **[cfg-db-cmd:entries]** スタンプおよび **[cfg-db-cmd:files]** スタンプ内のデフォルト・エントリーは、UNIX プラットフォームと UNIX プラットフォームとの間、または Windows と Windows との間での複製用に設計されています。Windows プラットフォームと UNIX プラットフォームとの間で複製を行う場合は、複製する構成エントリーおよびファイルに関して慎重に考慮する必要があります。

同期の自動化

ご使用の環境に複数の WebSEAL サーバーを構成できます。クラスター という用語は、連携するよう構成された WebSEAL サーバーのグループを意味します。WebSEAL クラスターを使用して、異なる WebSEAL サーバー間で構成情報の同期

化を自動化することができます。また、クラスタリングによってシステムの可用性およびパフォーマンスを向上させることもできます。

注: すべてのクラスター・メンバーは、**同じサーバー・タイプ**でなければなりません。クラスター用の WebSEAL サーバー・タイプ は、以下のいずれかです。

- Web ゲートウェイ・アプライアンス上で稼働する WebSEAL。
- 標準オペレーティング・システム上で稼働する WebSEAL。

クラスター環境では、すべてのクラスター構成変更を担当するマスター・サーバーを指定する必要があります。クラスター内のその他のサーバーは、スレーブ として指定されます。

スレーブは、再始動するたびにクラスター・マスターの構成情報をチェックします。スレーブの内部構成情報が最新ではない場合、そのスレーブは自動的にマスターと同期します。クラスター内のすべてのサーバーを同期するには、**cluster restart** サーバー・タスク・コマンドを使用します。

クラスターの再始動

このタスクについて

さまざまな WebSEAL サーバーを同期させるには、まず、マスター・サーバーを選択する必要があります。

例えば、汎用構成を格納する `default-webseald-master.ibm.com` などです。すべてのサーバーで使用可能にする次のような構成変更は、まず、マスター・サーバーに対して手動で行う必要があります。

- 構成ファイル。
- Junction 定義。

構成情報はマスターでのみ変更することが重要です。スレーブで構成を変更すると、サーバーの次の再始動時に、スレーブがその構成情報をマスターと同期させる際に情報を失う恐れがあります。

ご使用の環境に WebSEAL クラスタが構成されている場合は、マスター・サーバーから **cluster restart** サーバー・タスク・コマンドを実行して、すべての構成変更を適用し、更新したサーバーを再始動できます。クラスタ化された WebSEAL サーバーを同時に再始動するか、順番に再始動するかを指定する場合は、**-ripple** オプションを使用できます。クラスタ再始動の進行状況をモニターする場合に使用できる **-status** オプションもあります。716 ページの『server task cluster restart』を参照してください。

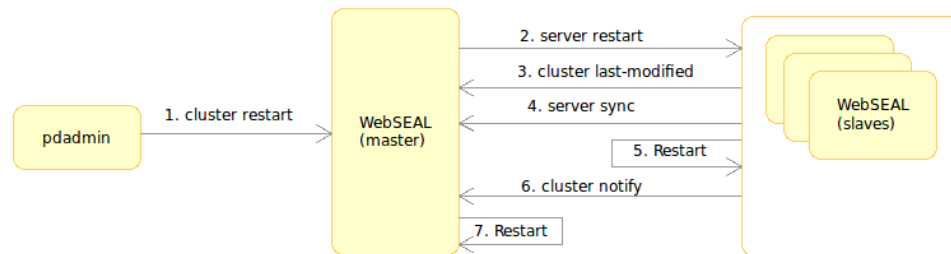


図 11. クラスター・サポート

図 11 は、以下の概略手順を含む、クラスターの再始動プロセスを示したものです。

手順

1. マスター・サーバーで **cluster restart** サーバー・タスク・コマンドを実行して、再始動を開始します。
2. マスターは、クラスター内の各スレーブで **server restart** コマンドを実行します。

注: これは、**-ripple** オプションがクラスター再始動コマンドで使用されたかどうかに応じて、同時または順番に行われます。

3. スレーブは、マスター構成情報が最後に変更された時刻を示すタイム・スタンプを取得します。
4. スレーブでローカルに保管された構成データが最新のものとない場合、スレーブはマスター構成データと同期します。
5. ステップ 4 でスレーブの構成データが更新された場合は、スレーブ・サーバーが再始動します。スレーブのローカル構成データベースで更新がなかった場合、スレーブは再始動しません。
6. スレーブは、その操作が完了したことをマスターに通知します。
7. 必要に応じて、すべてのスレーブが更新され、再始動されたら、マスター・サーバーが再始動します。

クラスター・サポート用の WebSEAL の構成

クラスター環境内の各 WebSEAL サーバーを構成するには、**[cluster]** 構成スタンザ・エントリーを使用します。これらのエントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」内の **[cluster] stanza** を参照してください。

WebSEAL クラスターのマスターとなるサーバーを 1 つ指定する必要があります。マスター・サーバーの **is-master** 構成エントリーは、**yes** に設定します。また、マスターの **max-wait-time** を構成することもできます。これは、マスター・サーバーが各スレーブ・サーバーの再始動を待機する最大秒数を表します。

クラスター内のその他すべてのサーバーでは、**is-master** 構成エントリーを **no** に設定する必要があります。これらのサーバーは、クラスター内のスレーブです。さらに、各スレーブで **master-name** という構成エントリーも構成する必要があります。

す。**master-name** 構成エントリーは、マスター・サーバーの許可サーバー名を指定します。例えば、`default-webseald-master.ibm.com` です。

注: クラスター環境でのフェイルオーバーの考慮事項については、288 ページの『新規マスターへのフェイルオーバー』を参照してください。

WebSEAL 用リソースの監査およびロギング

このセクションでは、WebSEAL の以下の監査およびロギング・リソースについて説明します。

- 『エラー・メッセージ・ロギング』
- 『WebSEAL サーバー・アクティビティの監査』
- 27 ページの『従来の HTTP イベントの監査およびロギング』

エラー・メッセージ・ロギング

WebSEAL サーバーのエラー・メッセージは、標準フォーマットを使用してフォーマットされます。WebSEAL では、ルーティング・ファイルを使用してエラー・メッセージの表示とメッセージ・データのロギングを制御します。デフォルトでは、WebSEAL はサーバー・エラー・メッセージ・データを 1 つのログ・ファイルに記録します。

WebSEAL のルーティング・ファイルによりデフォルト・メッセージ・ルーティングが制御されます。カスタマイズしたメッセージ・ロギングに合わせてルーティング・ファイルを変更できます。

WebSEAL ルーティング・ファイルとエラー・メッセージ・ロギングについて詳しくは、「*IBM Security Access Manager for Web: Troubleshooting Guide*」を参照してください。

WebSEAL サーバー・アクティビティの監査

Security Access Manager イベント・ロギング・メカニズムは、WebSEAL により生成される共通イベントおよびアクティビティを取り込むことができます。イベント・ロギングは、監査目的での情報収集のための構造化階層を提供します。また、イベント・ロギング機能では、ロギング出力用の代替宛先 (ファイル、リモート・サーバーなど) を使用できます。

イベント・ロギングをインプリメントするには、**logcfg** イベント・ロギング・スタンザ・エントリーを使用して、特定のカテゴリの監査情報をイベント・プールから収集し、この情報を宛先に送る 1 つ以上のログ・エージェント (ロガー) を定義します。

logcfg スタンザ・エントリーは、`pdaudit.conf` 構成ファイルの **[pdaudit-filter]** スタンザに入力します。`pdaudit.conf` 構成ファイルは、Common Auditing and Reporting Service (CARS) のコンポーネントです。

WebSEAL では WebSEAL 構成ファイルの **[aznapi-configuration]** スタンザにより、イベント・ロギングの構成がサポートされています。**logcfg** イベント・ロギン

グ・スタンザ・エントリーの形式は、pdaudit.conf 構成ファイルの **[pdaudit-filter]** スタンザに入力されている場合でも、WebSEAL 構成ファイルの **[aznapi-configuration]** スタンザに入力されている場合でも同一です。

イベント・ロギング・メカニズムについて詳しくは、「*IBM Security Access Manager for Web: Auditing Guide*」を参照してください。

『従来の監査メカニズム』も参照してください。

注: WebSEAL では、従来型の HTTP イベントの監査とロギングもサポートしています。この従来型メカニズムは、古い Security Access Manager のインストール済み環境との互換性が必要な状況でのみ使用してください。『従来の HTTP イベントの監査およびロギング』を参照してください。

従来の監査メカニズム

従来の監査は、WebSEAL 構成ファイルの **[aznapi-configuration]** スタンザ内の、以下のスタンザ・エントリーのそれぞれに値を指定することによって構成できます。

```
[aznapi-configuration]
logaudit
auditlog
auditcfg
logsize
logflush
```

このメソッドを使用すると、出力をファイルにダイレクトするときのイベント・ロギング・メソッドに類似した結果が得られます。ただし、イベント・ロギング・メソッドでは、バッファ・サイズとイベント・キューに対して、さらにコントロールができることに注意してください。また、従来の監査では、コンソール、パイプ、またはリモート・サーバーへの出力をサポートしていません。

認証イベント、許可イベント、および HTTP イベントの従来型監査構成の設定の詳細については、「*IBM Security Access Manager for Web: Auditing Guide*」を参照してください。

従来の HTTP イベントの監査およびロギング

WebSEAL では、次に示す 3 つの従来型 HTTP ログ・ファイルを維持します。これらのログ・ファイルには、HTTP アクティビティが記録されます。

- request.log
- agent.log
- referer.log

request-log-format エントリーを **[logging]** スタンザに追加することにより、要求ログをカスタマイズできます。カスタマイズされたロギングや HTTP イベントの従来の監査およびロギングなどの情報について詳しくは、「*IBM Security Access Manager for Web: Auditing Guide*」を参照してください。

WebSEAL 構成ファイルの **[logging]** スタンザにある以下のスタンザ・エントリーを使用して、文書 MIME タイプおよび応答コードで HTTP リソース・アクセス監査イベントをフィルターできます。

[logging]
audit-mime-types
audit-response-codes

WebSEAL の問題判別リソース

このセクションでは、以下の WebSEAL 用問題判別リソースについて説明します。

- 『構成データ・ログ・ファイル』
- 30 ページの『統計』
- 30 ページの『Trace ユーティリティ』

構成データ・ログ・ファイル

各 WebSEAL インスタンスは、インスタンスが開始されるたびに、ログ・ファイルに構成ファイルの全内容を記録します。この構成データを使用して、WebSEAL インストールおよび操作に関する問題をトラブルシューティングできます。WebSEAL の問題は、構成ファイルの設定の誤りや不完全さによるものが大半です。ログに記録された構成データは、指定された WebSEAL インスタンスの現在の構成設定の正確なスナップショットを提供します。

IBM サポートは、WebSEAL 関連の問題のトラブルシューティングを依頼されると、このログに記録された構成データに含まれる情報を利用することができます。この情報の正確性および完全性によって、トラブルシューティング・プロセスの効率が向上します。また、このログに記録された構成データを使用して、WebSEAL の問題を独自に診断することもできます。

構成データ・ログ・ファイルのロケーション

IBM Security Web Gateway Appliance LMI を使用して、構成データ・ログ・ファイルにアクセスすることができます。「リバース・プロキシ」管理ページで、該当する WebSEAL インスタンスを選択してから、「管理」>「ロギング」をクリックします。選択したインスタンスの「リバース・プロキシ・ログ・ファイルの管理」管理ページが表示されます。

構成データ・ログ・ファイルの増加に関する注意点

- WebSEAL インスタンスが初めて開始されると、ログ・ファイルは 1 つだけ作成されます。
- 完全な WebSEAL 構成ファイルのデータは、WebSEAL インスタンスが開始されるたびに、ログ・ファイルに記録されます。
- 後続の開始ごとに作成される構成ファイル・データは、同じログ・ファイルに追加されます。
- ログ・ファイルの各エントリはタイム・スタンプで始まります。このタイム・スタンプ・エントリによって、各データを他のデータから識別できます。
- ログ・ファイルのサイズは、新規エントリのたびに大きくなります。LMI を使用することで、このファイルのサイズを制御できます。

構成データ・ログ・ファイルの形式

WebSEAL 構成ファイルでは、スタンザ・エントリーのデフォルト値は、最初のインストールおよび構成中に WebSEAL によって提供された値です。アドミニストレータがその後構成ファイルの値を変更した場合、そのカスタム値は非デフォルト値になります。

構成データ・ログ・ファイルでは、特殊なマーカー (**[default]**) によって、デフォルトのスタンザ・エントリーの値が指定されます。非デフォルト値には、このマーカーは含まれません。例:

```
...
https = yes
https-port = [default] 443
http = yes
http-port = [default] 80
...
```

構成データ・ログ・ファイルの情報は、実際の WebSEAL 構成ファイルにある同じスタンザに応じてグループ化されます。以下は構成データ・ログ・ファイルを部分的に表示したもので、スタンザごとにグループ化されたスタンザ・エントリーおよび値が示されています。また、この例にはサンプルのタイム・スタンプ行も含まれています。

```
=====
Configuration Data Logged: Fri Jul 23 15:37:02 2004
...
[gso-cache]
gso-cache-enabled = [default] no
gso-cache-size = [default] 1024
gso-cache-entry-lifetime = [default] 900
gso-cache-entry-idle-timeout = [default] 120

[ltpa-cache]
ltpa-cache-enabled = [default] yes
ltpa-cache-size = [default] 4096
ltpa-cache-entry-lifetime = [default] 3600
ltpa-cache-entry-idle-timeout = [default] 600

[ba]
ba-auth = [default] https

[forms]
forms-auth = [default] none

[certificate]
accept-client-certs = [default] never
cert-cache-max-entries = [default] 1024
cert-cache-timeout = [default] 120

...
```

構成データ・ログ・ファイルに関連したメッセージ

構成データ・ログ・ファイルが作成されると、その WebSEAL ログ・ファイルに、ログ・ファイルの場所を指定する以下のような保守容易性メッセージが入力されます。

```
"The configuration data for this WebSEAL instance has been logged in %s"
```

`%s` マクロは、ログ・ファイルへの絶対パス (**config-data-log** スタンザ・エントリーで指定) を含むストリングで置換されます。

何らかの理由で WebSEAL 開始中に構成ファイル・データをログに記録できなくても、WebSEAL の始動は中断されることなく続行され、WebSEAL ログ・ファイルにエラー・メッセージが記録されます。

"An error occurred trying to log the WebSEAL configuration data at startup."

統計

WebSEAL は、特定のサーバー・アクティビティをモニターしたり、これらのアクティビティに関する情報を収集したりするための、一連の組み込みソフトウェア・モジュールを提供します。いつでも、アクティブなすべてのモジュールから統計を表示できます。さらに、統計情報はログ・ファイルに送信できます。

アクティブな各モジュールは、統計情報を収集します。一定間隔で統計を収集し、いつでもこの情報のスナップショットを表示するように、WebSEAL を構成することができます。

IBM Security Web Gateway Appliance は、フロー・データを収集します。フロー・データは、WebSEAL のパフォーマンスおよびスループットの統計を提供します。アプライアンスは、各 WebSEAL インスタンスの個別のデータベース内に統計を記録します。これらの統計を検索するには、ローカル管理インターフェース (LMI) を使用します。

アプライアンスのフロー・データ統計を構成するには、**[flow-data]** および **[user-agent]** スタンザを使用します。詳しくは、「*IBM Security Web Gateway Appliance: 管理ガイド*」を参照してください。

Trace ユーティリティー

trace ユーティリティーは、Security Access Manager、WebSEAL、Plug-in for Web Servers におけるエラー条件およびプログラム制御に関する情報をキャプチャーします。この情報はファイルに格納され、デバッグおよび問題判別に使用されます。

trace ユーティリティーの主な目的は、サポート要員が Security Access Manager ソフトウェア機能で発生する問題を診断するのを支援することです。ユーザーは、Security Access Manager、WebSEAL、および Plug-in for Web Servers のトレース・コンポーネントの一部を利用できることがあります。しかし大半のコンポーネントは、技術サポート要員の支援のもとで複雑な問題を診断しているのでない限り、あまり役に立ちません。

トレース・データは主に、IBM Tivoli Software Support の要員が使用することを意図して作成されており、報告された問題の診断処理の一環として要求される場合もあります。しかし、経験のある製品アドミニストレーターなら、トレース・データを使用して Security Access Manager 環境における問題の診断および修正を行うこともできます。

トレース・ファイルにアクセスするには、IBM Security Web Gateway Appliance LMI を使用します。「リバース・プロキシ」管理ページで、該当の WebSEAL イ

インスタンスを選択してから、「管理」>「トラブルシューティング」>「トレース」をクリックします。選択したインスタンスの「リバース・プロキシ・トレースの管理」管理ページが表示されます。

第 2 部 構成

第 3 章 Web サーバー構成

この章では、WebSEAL の Web サーバー機能の構成について説明します。

この章で扱うトピックは以下のとおりです。

- 『WebSEAL サーバーおよびホスト名指定』
- 37 ページの『WebSEAL 構成ファイル』
- 39 ページの『ディレクトリーの索引付け』
- 40 ページの『コンテンツのキャッシング』
- 46 ページの『通信プロトコル構成』
- 54 ページの『インターネット・プロトコル・バージョン 6 (IPv6) のサポート』
- 57 ページの『LDAP ディレクトリー・サーバー構成』
- 58 ページの『ワーカー・スレッドの割り振り』
- 62 ページの『HTTP データ圧縮』
- 65 ページの『UTF-8 でのマルチロケール・サポート』
- 76 ページの『要求データ内の文字エンコード方式の検証』
- 77 ページの『サポートされているワイルドカード・パターン・マッチング文字』
- 78 ページの『システム環境変数の設定』

WebSEAL サーバーおよびホスト名指定

このセクションは、以下のトピックで構成されています。

- 『構成ファイル内の WebSEAL サーバー名』
- 36 ページの『「pdadmin サーバー・リスト」内の WebSEAL サーバー名』
- 36 ページの『保護オブジェクト・スペース内の WebSEAL サーバー名』
- 37 ページの『WebSEAL ホスト (マシン) 名の指定』

構成ファイル内の WebSEAL サーバー名

WebSEALサーバー名によって、WebSEAL サーバー・プロセスが一意的に識別されます。1 つのコンピューター・システムに複数の WebSEAL サーバーをインストールおよび構成することができます。したがって、WebSEAL サーバー・プロセスごとに固有の名前がなければなりません。各 WebSEAL サーバー・プロセスは、インスタンスと呼ばれます。

各 WebSEAL インスタンスは、固有の構成ファイルを保有しています。WebSEAL インスタンスごとに、構成ファイルの **[server]** スタンザ内の **server-name** スタンザ・エントリーに WebSEAL インスタンスの固有の名前を指定します。

server-name スタンザ・エントリーは、WebSEAL がインストールされている物理的なマシンの**ホスト名**と、WebSEAL **インスタンス名**の組み合わせで構成されています。両方の名前は WebSEAL の構成時に指定します。


```
[server]
server-name = host_name-instance_name
```

マシンの**ホスト名**は常に完全修飾名で (例えば **abc.ibm.com**)、さらに短縮名を追加することもできます (例えば **abc**)。WebSEAL の構成時に**ホスト名**を要求された場合、完全修飾名と短縮名のどちらかを指定することができます。

以下の例では、WebSEAL インスタンス名 **web1** は、**abc.ibm.com** (WebSEAL 構成中に指定した値) という完全修飾ホスト名のマシンにあります。

```
[server]
server-name = abc.ibm.com-web1
```

WebSEAL の構成中にこの名前を変更しない限り、最初の WebSEAL サーバーには自動的に **default** のインスタンス名が割り当てられます。例:

```
[server]
server-name = abc.ibm.com-default
```

「pdadmin サーバー・リスト」内の WebSEAL サーバー名

また、インスタンス名は、**pdadmin server list** コマンドの実行によって WebSEAL サーバーがどのようにリストされるかに影響を与えます。**pdadmin** コマンドは全 Security Access Manager ファミリーで使用できるため、コマンド構文に製品コンポーネント名が必要になります。WebSEAL のコンポーネント名は、**webseald** です。**pdadmin server list** コマンドの場合、WebSEAL サーバー名の形式は次のようになります。

instance_name-webseald-host_name

例えば、**abc.ibm.com** というホストにインストールされているインスタンス **web1** の **pdadmin server list** の出力内容は、次のようになります。

web1-webseald-abc.ibm.com

以下の **pdadmin server list** コマンドの出力では、最初のデフォルト WebSEAL サーバーと 2 番目の WebSEAL インスタンス **web1** が表示されています。

```
pdadmin> server list
web1-webseald-abc.ibm.com
default-webseald-abc.ibm.com
```

保護オブジェクト・スペース内の WebSEAL サーバー名

各 WebSEAL インスタンスは、保護オブジェクト・スペースで **/WebSEAL** コンテナ・オブジェクトのメンバーとして表されます。

ホスト **abc.ibm.com** にある 2 つの WebSEAL インスタンス (**default** および **web1**) は、保護オブジェクト・スペースでは以下の形式で表示されます。

```
/WebSEAL/abc.ibm.com-web1
/WebSEAL/abc.ibm.com-default
```

WebSEAL ホスト (マシン) 名の指定

このタスクについて

通常、WebSEAL ホスト・マシンの名前は、この情報が必要なときに自動的に決定されます。しかし、仮想ホスト Junction のように WebSEAL ホストが複数の名前を使用する状態もあります。多数のホスト名、インターフェース、または WebSEAL インスタンスが存在するシステムでは、自動的に決定された名前が、常に指定された状態に適した値であるとは限らない場合があります。

WebSEAL 構成ファイルの **[server]** スタンザ内の **web-host-name** スタンザ・エントリーによって、WebSEAL サーバーのホスト (マシン) 名を手動で指定することができます。この値は、完全修飾名でなければなりません。この手動設定によって、ホスト名を決定する際の競合を解決することができます。例えば、従来の junction 環境で WebSEAL HTTP/HTTPS 応答および認証メカニズムによって使用されるホスト名などです。

デフォルトでは、**web-host-name** は使用不可に設定されており、値は指定されていません。必要なときに WebSEAL が自動的にホスト名を決定します。

ホスト名の値を指定して自動決定をオーバーライドするには、手順に従ってください。

手順

1. WebSEAL サーバー・プロセスを停止します。
2. 手動で WebSEAL 構成ファイルを編集し、スタンザ・エントリーの値を入力します。
3. 行のコメントを外します。
4. WebSEAL を再始動します。

タスクの結果

以下に例を示します。

```
[server]
web-host-name = abc.ibm.com
```

server-name と **web-host-name** の値の構文上の違いに注意してください。例:

```
[server]
server-name = abc.ibm.com-default
web-host-name = abc.ibm.com
```

WebSEAL 構成ファイル

WebSEAL サーバーの操作は、WebSEAL 構成ファイルおよび機密データに使用される対応する obfuscate ファイルを使用して制御します。

WebSEAL 構成ファイル内で使用可能なスタンザ・エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

構成ファイルの編成

構成ファイルには、WebSEAL 機能の特定の部分をコントロールするいくつかのセクションが入っています。各セクションは、さらにスタンザと呼ばれるセクションに分かれています。

スタンザのラベルは、次のようにブラケットで囲んで表します。

[*stanza_name*]

例えば、[ssl] スタンザは、WebSEAL サーバーで使用する SSL 構成の設定値を定義します。

Security Access Manager 構成ファイル内の各スタンザには、1 つ以上のスタンザ・エントリーが入っています。スタンザ・エントリーは、スタンザ・エントリーのペアのセットとして表される情報が入るキーと値の組から成っています。各スタンザ・エントリーでは、次の形式が使用されます。

key = value

WebSEAL の初期のインストールによって、デフォルト値のほとんどが設定されます。一部の値は静的なもので、変化することはありません。その他の値は、サーバーの機能とパフォーマンスをカスタマイズするために変更できます。

ASCII ベースのテキスト・ファイルは、一般的なテキスト・エディターで編集できます。

構成ファイルの名前とロケーション

それぞれの WebSEAL インスタンスごとに、固有の WebSEAL 構成ファイルが作成されます。構成ファイルの名前にはインスタンス名が組み込まれています。

次の形式が使用されます。

webseald-instance_name.conf

管理者は、ローカル管理インターフェース (LMI) を使用して、追加の WebSEAL インスタンスを構成し、個々の新しい *instance_name* を指定できます。

構成ユーティリティーは、指定された *instance_name* を使用して、新しい WebSEAL 構成ファイルの名前を付けます。例えば、新しい WebSEAL インスタンスに **webseal2** という名前を付けた場合は、次の構成ファイルが作成されます。

webseald-webseal2.conf

WebSEAL インスタンスの構成について詳しくは、663 ページの『第 36 章 WebSEAL インスタンスのデプロイメント』を参照してください。

構成ファイルの設定の変更

このタスクについて

構成設定を変更するには、以下のステップに従います。

手順

1. LMI を使用して構成ファイルを開きます。「リバース・プロキシ」管理ページで、該当する WebSEAL インスタンスを選択して「管理」>「構成」>「構成ファイルの編集」をクリックすると、拡張構成ファイル・エディターが開きます。

注: 基本的な構成ファイルの更新を完了するには、「リバース・プロキシの基本構成」ページを使用する方法もあります。選択した WebSEAL インスタンスの該当するページにアクセスするには、「リバース・プロキシ」管理ページで「編集」をクリックします。

2. この構成に必要な変更を加えます。
3. 変更内容を保管します。デプロイされていない変更があることを示す警告メッセージが LMI に表示されます。
4. 変更をデプロイするには、「変更を確認する場合、または変更をシステムに適用する場合は、ここをクリックしてください」というリンクをクリックします。変更内容を確認して「デプロイ」をクリックします。デプロイが完了し、再始動が必要であることを示すシステム警告が表示されます。
5. これらの変更を有効にするため、WebSEAL インスタンスを「リバース・プロキシ」管理ページから再始動します。

ディレクトリーの索引付け

このセクションは、以下のトピックで構成されています。

- 『ディレクトリーの索引付けの構成』
- 40 ページの『ファイル・タイプごとのグラフィカル・アイコンの構成』

ディレクトリーの索引付けの構成

要求の URL 表現がディレクトリー名で終わっている場合は、WebSEAL によって戻されるデフォルト・ファイルの名前を指定できます。このデフォルト・ファイルが存在していれば、WebSEAL はこのファイルをクライアントに戻します。このファイルが存在しない場合は、WebSEAL はディレクトリー索引を動的に生成し、そのリストをクライアントに戻します。

ディレクトリー索引ファイルの構成に使用する **directory-index** スタンザ・エントリーは、WebSEAL 構成ファイルの **[content]** スタンザ内にあります。索引ファイルのデフォルト値は次のとおりです。

```
[content]
directory-index = index.html
```

ご使用のサイトで異なる規則を使用している場合は、このファイル名を変更できます。以下に例を示します。

```
[content]
directory-index = homepage.html
```

要求の中で指定されているディレクトリーに、**directory-index** スタンザ・エントリーで定義されている索引ファイルが含まれていない場合は、WebSEAL は動的にディレクトリー索引を生成します。生成した索引には、ディレクトリーの内容のリス

トと、そのディレクトリー内の各エントリーへのリンクが含まれています。索引が生成されるのは、ディレクトリーへのアクセスを要求しているクライアントが、そのディレクトリー用の ACL 内に「list」(l) 許可を持っている場合に限られます。

ファイル・タイプごとのグラフィカル・アイコンの構成

生成された索引にリストされている各ファイル・タイプについて、WebSEAL が使用するグラフィック・アイコンを構成することができます。WebSEAL 構成ファイルの **[content-index-icons]** スタンザには、文書 MIME タイプのリストと、表示される関連の .gif ファイルが入っています。

```
[content-index-icons]
image/* = /icons/image2.gif
video/* = /icons/movie.gif
audio/* = /icons/sound2.gif
text/html = /icons/generic.gif
text/* = /icons/text.gif
application/x-tar = /icons/tar.gif
application/* = /icons/binary.gif
```

このリストを構成することにより、各 MIME タイプについて他のアイコンを指定できます。アイコンはリモートで配置することもできます。以下に例を示します。

```
application/* = http://www.example.com/icons/binary.gif
```

また、以下に示す追加のアイコン値も構成できます。

- サブディレクトリーを表すアイコン。

```
[icons]
diricon = /icons/folder2.gif
```

- 親ディレクトリーを表すアイコン。

```
[icons]
backicon = /icons/back.gif
```

- 不明ファイル・タイプを表すアイコン。

```
[icons]
unknownicon = /icons/unknown.gif
```

注: 提供されたアイコンは GIF 形式ですが、この形式は必須ではありません。

コンテンツのキャッシング

このセクションは、以下のトピックで構成されています。

- 41 ページの『コンテンツ・キャッシングの概念』
- 41 ページの『コンテンツ・キャッシングの構成』
- 42 ページの『WebSEAL コンテンツ・キャッシングへの HTTP ヘッダーの与える影響』
- 45 ページの『すべてのキャッシュのフラッシュ』
- 45 ページの『特定文書に関するキャッシュ・コントロール』

コンテンツ・キャッシングの概念

ユーザーは、Web 文書検索のパフォーマンスが低いため、ネットワークのアクセス時間とファイルのダウンロード時間が長引くという状況をしばしば経験します。パフォーマンスの低下は、junction 先バックエンド・サーバーから文書が検索されるのを WebSEAL サーバーが待っている場合に起こります。

Web コンテンツのキャッシングを利用すれば、junction 経由のバックエンド・サーバーからではなく、WebSEAL からローカルで文書にサービスできる柔軟性が得られます。コンテンツのキャッシング機能を使用すると、共通にアクセスされる Web 文書タイプを WebSEAL サーバーのメモリーに保管できます。WebSEAL サーバー内にキャッシュされた文書を、後でクライアントが要求した場合には、応答速度がはるかに高速になります。

キャッシュされるコンテンツには、静的テキスト文書とグラフィック・イメージを入れることができます。データベース照会の結果など、動的に生成された文書はキャッシュできません。

キャッシングは、MIME タイプに基づいて実行されます。コンテンツ・キャッシング用に WebSEAL を構成する場合、以下の 3 つの設定を指定します。

- 文書 MIME タイプ
- ストレージ・メディアのタイプ
- ストレージ・メディアのサイズ
- 元の応答から有効期限切れ情報が欠落している場合の最大存続期間

コンテンツ・キャッシングの構成

コンテンツのキャッシュは、WebSEAL 構成ファイルの **[content-cache]** スタンザで構成します。適用される構文は、次のとおりです。

```
<mime-type> = <cache-type>:<cache-size>
```

変数	説明
mime-type	HTTP 「Content-Type:」 応答ヘッダーに入れて伝達される有効な MIME タイプを表します。この値には、アスタリスク (*) を使うことができます。*/* という値は、明示的に構成されたキャッシュに対応しないすべてのオブジェクトを保持するデフォルト・オブジェクト・キャッシュを表します。ここで使用されているアスタリスクは、MIME タイプのディレクトリーおよびその内容を表すためのみのワイルドカードであることに注意してください。このアスタリスクは、正規表現マッチングのためのワイルドカードではありません。
cache-type	キャッシュに使用するストレージ・メディアのタイプを指定します。Security Access Manager の本リリースがサポートしているのは、「メモリー」キャッシュだけです。
cache-size	「LRU (Least Recently Used: 最低使用頻度)」アルゴリズムに従ってオブジェクトが削除される前に、与えられたキャッシュが自動拡張できる最大サイズを (キロバイトで) 指定します。

変数	説明
def-max-age	元の応答から有効期限切れ情報が欠落している場合の最大存続期間 (秒) を指定します。値が指定されない場合、デフォルトの最大存続期間の 3600 (1 時間) が適用されます。

例:

```
[content-cache]
text/html = memory:2000
image/* = memory:5000
*/* = memory:1000
```

コンテンツのキャッシング構成に影響を与える条件

コンテンツ・キャッシュ・メカニズムは、以下の条件を監視します。

- WebSEAL 構成ファイルでキャッシュが定義されている場合に限り、コンテンツのキャッシングが行われること。
- デフォルトでは、インストール時にコンテンツのキャッシュが定義されていないこと。
- デフォルトのコンテンツのキャッシュを定義していない場合、どの明示的キャッシュにも一致しない文書はキャッシュされないこと。
- 許可は、キャッシュされた情報に対するすべての要求について、引き続き実施されること。
- コンテンツのキャッシュ・メカニズムが、照会ストリングを含む要求に対する応答をキャッシュしないこと。
- コンテンツのキャッシュ・メカニズムが、**-c** および **-C** オプションを指定して構成された junction を介した要求に対する応答をキャッシュしないこと。

WebSEAL コンテンツ・キャッシングへの HTTP ヘッダーの与える影響

このセクションは、以下のトピックで構成されています。

- 『WebSEAL コンテンツ・キャッシングへの応答ヘッダーの影響』
- 43 ページの 『WebSEAL コンテンツ・キャッシングへの要求ヘッダーの影響』
- 44 ページの 『WebSEAL コンテンツ・キャッシングに影響するその他の条件』

WebSEAL コンテンツ・キャッシングへの応答ヘッダーの影響

以下の表では、(junction の) 特定の HTTP 応答ヘッダーが、WebSEAL がリソースのキャッシングを許可するかどうかに与える影響について説明します。

応答ヘッダー	コンテンツ・キャッシングに与える影響
Cache-control	応答にこのヘッダーが存在し、その値が「no-cache」、「no-store」、「private」の場合、リソースはコンテンツ・キャッシュに格納されません。

応答ヘッダー	コンテンツ・キャッシングに与える影響
Cache-control	応答にこのヘッダーが存在し、その値が「public」の場合、ユーザー識別データが junction に渡されても、リソースをコンテンツ・キャッシュに格納できます。
Content-range	応答にこのヘッダーが存在すると、リソースはコンテンツ・キャッシュに格納されません。
Expires	応答にこのヘッダーが存在すると、ユーザー識別データが junction に渡されても、リソースをコンテンツ・キャッシュに格納できます。
Pragma	応答にこのヘッダーが存在し、その値が「non-cache」の場合、リソースはコンテンツ・キャッシュに格納されません。
Transfer-encoding	WebSEAL は、junction に送信された要求から TE: ヘッダーを取り除くため、Transfer-Encoding: ヘッダーが送り戻されることを予想していません。しかし、Transfer-Encoding: ヘッダーが送り戻された場合、WebSEAL は、リソースをコンテンツ・キャッシュに格納しません。
Vary	応答にこのヘッダーが存在すると、リソースはコンテンツ・キャッシュに格納されません。

その他の条件:

- Age、Date、Last-Modified、および Expires ヘッダーを使用して、キャッシュ内のデータが最新で利用できるかどうかを計算します。
- Date: ヘッダーの値が Expires: ヘッダーの値よりも大きい場合、リソースはキャッシュに格納されません。これは、HTTP/1.0 仕様に準拠しています。
- WebSEAL は、**http-equiv** 属性を持つ **<meta>** タグを処理しません。

WebSEAL コンテンツ・キャッシングへの要求ヘッダーの影響

クライアントからの HTTP 要求ヘッダーは、WebSEAL がキャッシュを使用して、要求されたリソースを生成する方法に影響します。

以下の表では、(クライアントの) 特定の HTTP 要求ヘッダーが、WebSEAL がキャッシュを使用して要求されたリソースを作成するかどうか、または要求を宛先サーバーに送信するかどうかに与える影響について説明しています。

要求ヘッダー	コンテンツ・キャッシングに与える影響
Accept-encoding:	要求にこのヘッダーが存在すると、キャッシュされているエンコード・タイプと値が一致した場合、応答をキャッシュから取得します。
Authorization:	POP 値 DocumentCacheControl=public が設定されていない場合、要求にこのヘッダーが存在すると、junction が -b フィルター・セットを保有していない限り、応答をキャッシュから取得しません。 45 ページの『特定文書に関するキャッシュ・コントロール』を参照してください。

要求ヘッダー	コンテンツ・キャッシングに与える影響
Cache-control:	要求にこのヘッダーが存在し、その値が「no-cache」の場合、応答をキャッシュから取得しません。
Cache-control:	要求にこのヘッダーが存在し、その値が「no-store」の場合、応答のキャッシュからの取得や応答のキャッシュへの格納は行いません。
Cache-control:	応答でキャッシュが使用されるかどうかは、値「max-age」、「max-stale」、または「min-fresh」を使用して判断します。
Pragma:	要求にこのヘッダーが存在し、その値が「non-cache」の場合、応答をキャッシュから取得しません。
Range:	要求にこのヘッダーが存在すると、応答のキャッシュからの取得や応答のキャッシュへの格納は行いません。

WebSEAL コンテンツ・キャッシングに影響するその他の条件

その他の条件も WebSEAL コンテンツ・キャッシングに影響します。

WebSEAL コンテンツ・キャッシングに影響するその他の条件には、以下が含まれます。

- junction からの応答の状況が「200 OK」でない場合、WebSEAL はコンテンツをキャッシュしません。
- 要求 URL に照会ストリングが含まれる場合、WebSEAL はコンテンツをキャッシュしません。
- URL に PUT、POST、または DELETE が使用されている場合、WebSEAL はキャッシュ・エントリをフラッシュします。
- HEAD および GET 要求の場合、WebSEAL はキャッシュからの値のみを戻します。
- WebSEAL は、HEAD 要求に対する junction からの応答をキャッシュしません。
- WebSEAL は、junction が **-b gso**、**-b supply**、**-C**、または **-c** セットを保有している場合は応答をキャッシュしません。ただし、以下のいずれかが当てはまる場合を除きます。
 - **DocumentCacheControl=public** の値を持ったオブジェクト上に POP が存在する場合。45 ページの『特定文書に関するキャッシュ・コントロール』を参照してください。
 - 応答が Expires: ヘッダーを保有する場合。
 - 応答が Cache-Control: 共通セットを保有する場合。
- **DocumentCacheControl=no-cache** の値を持ったオブジェクト上に POP が存在する場合、WebSEAL は応答をキャッシュしません。45 ページの『特定文書に関するキャッシュ・コントロール』を参照してください。
- 日付ヘッダー (Date、Age、Last-modified、Expires、および関連した Cache-control ヘッダーの値) に基づいた計算をオーバーライドすることはできません。
- 他のすべてのヘッダーをオーバーライドすることはできません。

すべてのキャッシュのフラッシュ

このタスクについて

pdadmin ユーティリティを使用して、すべての構成済みコンテンツのキャッシュをフラッシュできます。このユーティリティを使用して、キャッシュを個別にフラッシュすることはできません。

注: **pdadmin** を使用するには、Security Access Manager アドミニストレータ **sec_master** としてセキュア・ドメインにログインする必要があります。

すべてのコンテンツ・キャッシュをフラッシュするには、以下の **pdadmin** コマンドを使用します。

```
pdadmin> server task instance_name-webseald-host_name cache flush all
```

特定文書に関するキャッシュ・コントロール

特定文書に関するキャッシュをコントロールするには、それらの文書 (オブジェクト) に特殊な保護オブジェクト・ポリシー (POP) を付加します。この POP には、**document-cache-control** と呼ばれる拡張属性が含まれていなければなりません。

document-cache-control 拡張属性は、次の 2 つの値を取ることができます。

値	説明
no-cache	no-cache 値は、この文書をキャッシュに入れないことを WebSEAL に指示します。この POP が付随しているオブジェクトのすべての子も、POP 条件を継承するという点に注意してください。
public	public 値を使用すると、WebSEAL は、 -c または -C オプションを指定した junction が作成されているという事実を無視して、文書をキャッシュに入れます。さらに、この値を使用することにより、要求が許可ヘッダー (基本認証など) と共に送られた場合に、この文書がキャッシュに入れられるようにすることもできます。この条件には、WebSEAL が、クライアントに代わって、要求に基本認証情報を挿入した場合 (GSO または -b supply junction を使用) も含まれます。通常は、プロキシ・サーバーは、許可ヘッダーを含む要求に対する応答文書はキャッシュに入れません。

pdadmin pop create、**pdadmin pop modify**、および **pdadmin pop attach** コマンドを使用して、保護オブジェクトに POP を設定します。

次の例では、**document-cache-control** 拡張属性を持つ「doc-cache」という名前の POP を作成し、それをオブジェクト (budget.html) に付加する方法を説明しています。

```
pdadmin> pop create doc-cache
pdadmin> pop modify doc-cache set attribute document-cache-control no-cache
pdadmin> pop attach /WebSEAL/hostA/junction/budget.html doc-cache
```

WebSEAL は budget.html 文書をキャッシュしません。したがって、この文書を要求するには、この文書が格納されているバックエンド・サーバーに対して要求を直接出す必要があります。

pdadmin コマンド行ユーティリティーについて詳しくは、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

通信プロトコル構成

WebSEAL の通信プロトコルは、WebSEAL による要求の処理方法と接続の作成方法を制御するように構成できます。以下のセクションで詳しく説明されているとおり、通信プロトコルを構成するための多くのスタンザ・エントリーが用意されています。

- 『HTTP 要求用の WebSEAL の構成』
- 47 ページの 『HTTPS 要求用の WebSEAL の構成』
- 47 ページの 『特定の SSL バージョンからの接続の制限』
- 48 ページの 『HTTP 持続接続』
- 49 ページの 『HTTP および HTTPS 通信用のタイムアウト設定』
- 51 ページの 『追加の WebSEAL サーバー・タイムアウト設定』
- 52 ページの 『WebDAV のサポート』
- 54 ページの 『Microsoft RPC over HTTP のサポート』
- 54 ページの 『チャンク転送コーディングのサポート』

HTTP 要求用の WebSEAL の構成

WebSEAL は、通常、認証されていないユーザーからの多数の HTTP 要求を処理します。例えば、ご使用の公開 Web サイトにある選択された資料に対しては、匿名ユーザーに、読み取り専用アクセスを可能にすることが一般的です。

TCP を介して HTTP 要求を処理するためのスタンザ・エントリーは、WebSEAL 構成ファイルの **[server]** スタンザにあります。

HTTP アクセスの使用可能化または使用不可化

このタスクについて

IBM HTTP Server、WebSphere Application Server (これは IBM HTTP Server をインストールします)、および WebSEAL は、どれもデフォルト・ポートとしてポート 80 を使用します。IBM HTTP Server と同じシステムに WebSEAL をインストールする場合は、必ずこれらのいずれかのサーバーへのデフォルト・ポートを変更してください。そのためには、`httpd.conf` 構成ファイルまたは WebSEAL 構成ファイルを編集します。

WebSEAL 構成時に、HTTP アクセスを使用可能または使用不可にするには、以下のコマンドを実行します。

```
[server]
http = {yes|no}
```

HTTP アクセス・ポート値の設定

このタスクについて

HTTP アクセス用のデフォルト・ポートは 80 です。

```
[server]
http-port = 80
```

例えば、ポート 8080 に変更する場合は、次のように設定します。

```
[server]
http-port = 8080
```

HTTPS 要求用の WebSEAL の構成

SSL (HTTPS) を介して HTTP 要求を処理するためのスタンザ・エントリーは、WebSEAL 構成ファイルの **[server]** スタンザに入っています。

HTTPS アクセスの使用可能化または使用不可化

このタスクについて

WebSEAL 構成時に HTTPS アクセスを使用可能または使用不可にするには、以下のようになります。

```
[server]
https = {yes|no}
```

HTTPS アクセス・ポート値の設定

このタスクについて

HTTPS アクセス用のデフォルト・ポートは 443 です。

```
[server]
https-port = 443
```

例えば、ポート 4343 に変更する場合は、次のように設定します。

```
[server]
https-port = 4343
```

特定の SSL バージョンからの接続の制限

以下の通信プロトコル・バージョンの場合、接続を個別に使用可能または使用不可にすることができます。

- Secure Sockets Layer (SSL) バージョン 2
- SSL バージョン 3
- Transport Layer Security (TLS) バージョン 1
- TLS バージョン 1.1
- TLS バージョン 1.2

特定の SSL および TLS バージョンの接続を制御するスタンザ・エントリーは、WebSEAL 構成ファイルの **[ssl]** スタンザにあります。デフォルトでは、SSL バージョン 2 は使用不可になっています。デフォルトでは、他のすべての SSL および TLS のバージョンが使用可能です。

```
[ssl]
disable-ssl-v2 = {yes|no}
disable-ssl-v3 = {yes|no}
disable-tls-v1 = {yes|no}
disable-tls-v11 = {yes|no}
disable-tls-v12 = {yes|no}
```

HTTP 持続接続

WebSEAL は、HTTP 通信層でクライアント・ブラウザと WebSEAL との間、および Junction Web サーバーと WebSEAL との間の持続接続を維持します。

クライアント接続は、WebSEAL 構成ファイルの **[server]** スタンザにある次のエントリーによってコントロールされます。

max-idle-persistent-connections

このエントリーは、アイドル・クライアントの持続接続の最大数を制御します。

persistent-con-timeout

このエントリーは、HTTP 持続接続をシャットダウンするまで WebSEAL が新規要求に対して接続をオープン状態にしておく最大秒数をコントロールします。49 ページの『HTTP および HTTPS 通信用のタイムアウト設定』を参照してください。

disable-timeout-reduction

このエントリーは、アクティブなワーカー・スレッド数の制御に役立つように、WebSEAL がタイムアウト期間を短縮するかどうかを決定します。デフォルトでは、WebSEAL は、使用中のワーカー・スレッド数が増加するにつれて、タイムアウト期間を自動的に短縮します。

Junction 接続は、WebSEAL 構成ファイルの **[junction]** スタンザにある次のエントリーによってコントロールされます。

max-cached-persistent-connections

このエントリーは、将来使用するためにキャッシュに保存する持続接続の最大数をコントロールします。

persistent-con-timeout

このエントリーは、キャッシュ内の持続接続を WebSEAL がクローズするまでのアイドル状態の最大秒数をコントロールします。

注: WebSEAL は、HTTP/1.1 持続接続をサポートしています。HTTP/1.0 持続接続はサポートされません。

HTTPOnly Cookie を処理するための WebSEAL 構成

クロスサイト・スクリプトは、Web サーバーに関するセキュリティの問題でも最も代表的なもので、Web サイトのユーザーに関する機密情報が漏えいする可能性があります。クロスサイト・スクリプトの危険性を軽減するために、HTTPOnly 属

性が Cookie に追加され、クライアント側のスクリプトを通して Cookie がアクセスされないようになりました。WebSEAL では、セッション、フェイルオーバー、および LTPA Cookie に使用する Set-Cookie ヘッダーに HTTPOnly 属性を追加可能にするオプションが含まれています。また、HTTP-only Set-Cookie ヘッダー属性をバックエンド Junction サーバーから Web ブラウザーに受け渡すように WebSEAL を構成することもできます。

HTTPOnly 属性を Session、Failover、および LTPA Set-Cookie ヘッダーに追加するように WebSEAL を構成するには、WebSEAL 構成ファイルの **[server]** スタンザ内の **use-http-only-cookies** の値を **yes** に変更します。デフォルト値は **no** です。

```
[server]
use-http-only-cookies = yes
```

Junction 先サーバーから送信された Set-Cookie ヘッダーからの HTTPOnly 属性を受け渡すように WebSEAL を構成するには、WebSEAL 構成ファイルの **[junction]** スタンザ内の **pass-http-only-cookie-attr** の値を **yes** に変更します。デフォルト値は **no** です。

```
[junction]
pass-http-only-cookie-attr = yes
```

これらのエントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

HTTP および HTTPS 通信用のタイムアウト設定

WebSEAL は、HTTP 通信と HTTPS 通信のタイムアウトになる設定をサポートします。タイムアウト設定のためのスタンザ・エントリーは、通常 WebSEAL 構成ファイルの **[server]** スタンザにあります。

- **client-connect-timeout**

このスタンザ・エントリーは、初期接続ハンドシェイクが行われた後で、最初の HTTP または HTTPS 要求を待つために、WebSEAL が接続をオープン状態にしておく時間を指定します。デフォルト値は 120 秒です。

```
[server]
client-connect-timeout = 120
```

- **intra-connection-timeout**

このスタンザ・エントリーは、複数のフラグメントとして送信された要求および応答データに影響を与えます。このスタンザ・エントリーは、WebSEAL が最初のデータ・フラグメントを受信した後の、各要求データ・フラグメント間のタイムアウト (秒数) を指定します。また、スタンザ・エントリーは、最初のデータ・フラグメントが WebSEAL によって戻された後の応答データ・フラグメント間のタイムアウトも管理します。デフォルト値は 60 秒です。

```
[server]
intra-connection-timeout = 60
```

このスタンザ・エントリーの値を 0 に設定した場合 (または設定しない場合)、データ・フラグメント間の接続タイムアウトは、**client-connect-timeout** スタンザ・

エントリーによって管理されます。このルールの例外は、HTTP (TCP) 経由で戻された応答の場合に発生します。この場合、応答フラグメント間でのタイムアウトはありません。

intra-connection-timeout の設定が原因で接続タイムアウトが 2 番目以降のデータ・フラグメントで発生する場合、TCP RST (リセット) パケットが送信されます。

- **persistent-con-timeout**

このスタンザ・エントリーは、HTTP 要求およびサーバー応答の交換が完了した後、HTTP 持続接続をシャットダウンするまで WebSEAL が新規クライアント要求に対して接続をオープン状態にしておく最大秒数をコントロールします。デフォルト値は 5 秒です。

```
[server]
persistent-con-timeout = 5
```

このスタンザ・エントリーの値を 0 に設定した場合、後続の要求では接続が開いたままになりません。値をゼロにすると、WebSEAL は「**Connection: close**」ヘッダーを設定し、すべての応答で接続を閉じます。

注: Junction 側のタイムアウト設定は、[junction] スタンザ内の **persistent-con-timeout** エントリーによって設定されます。

以下のフロー・チャートは、タイムアウト設定が、例に挙げた要求および応答の交換に影響を与える箇所を示しています。要求および応答を表すフラグメントの数は、例として使用しているだけです。

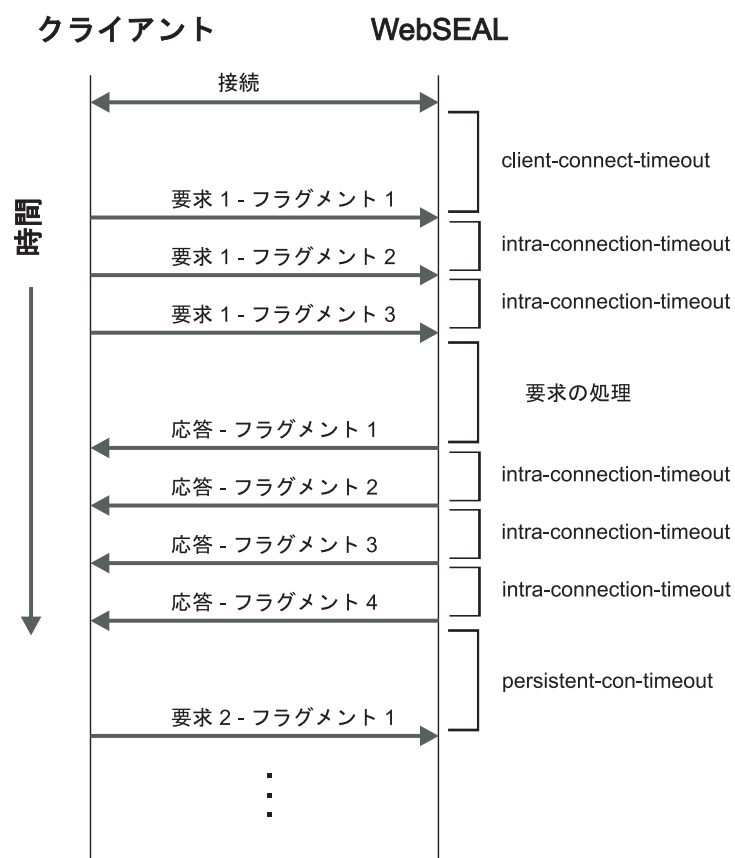


図 12. HTTP および HTTPS 通信のタイムアウト設定

追加の WebSEAL サーバー・タイムアウト設定

以下に示す追加のタイムアウト設定は、WebSEAL 構成ファイルに設定されています。

スタンザ・エントリー	説明	デフォルト値
[junction] http-timeout	TCP junction を通して行うバックエンド・サーバーに対する送信と読み取りのタイムアウト値。	120
[junction] https-timeout	SSL junction を通して行うバックエンド・サーバーに対する送信と読み取りのタイムアウト値。	120
[cgi] cgi-timeout	ローカル CGI プロセスに対する送信と読み取りのタイムアウト値。 UNIX システムでのみサポートされています。	120

スタンザ・エントリー	説明	デフォルト値
[junction] ping-time	WebSEAL では、Junction された個々のサーバーに対して、ping をバックグラウンドで定期的に行うことで稼働しているかどうかを判断します。この値は、これらの ping の間の時間間隔を設定します。 ping をオフにするには、このエントリーをゼロに設定します。このエントリーをゼロに設定する場合、recovery-ping-time を設定する必要があります。	300
[junction] recovery-ping-time	このオプションのエントリーは、サーバーが稼働していないと判断された場合の ping の間隔を秒で設定します。このエントリーが設定されない場合、値のデフォルトは ping-time の設定になります。	300

オプションで、手動で Junction 固有のスタンザ (例えば、[junction:/WebApp]) を WebSEAL 構成ファイル内に作成することにより、特定の WebSEAL Junction の **http-timeout** および **https-timeout** 設定をカスタマイズできます。
[junction:junction_name] スタンザに次の調整済みの構成アイテムを追加することにより、特定の Junction に対してこのスタンザをカスタマイズできます。ここで、junction_name は標準 Junction (先頭の / を含む) の Junction ポイントまたは仮想ホスト Junction の仮想ホスト・ラベルです。次の設定を指定すると、上記のデフォルト設定がこの設定でオーバーライドされます。

スタンザ・エントリー	説明	デフォルト値 (秒)
[junction:junction_name] http-timeout	特定の TCP Junction を通じて行うバックエンド・サーバーに対する送信と読み取りのタイムアウト値。	120
[junction:junction_name] https-timeout	特定の SSL Junction を通じて行うバックエンド・サーバーに対する送信と読み取りのタイムアウト値。	120

WebDAV のサポート

Web-based Distributed Authoring and Versioning (WebDAV) は HTTP プロトコルの拡張機能であり、ユーザーはこれを使用して、リモート Web サーバー上のファイルを編集および管理することができます。

WebSEAL は、Junction アプリケーションのクライアント要求に WebDAV メソッドを使用することをサポートしています。WebSEAL がローカル・コンテンツを提

供する Web サーバーとして動作する場合は、WebDAV メソッドをサポートしません。すべての WebDAV メソッドの許可は、*r* ACL ビットを使用してコントロールされます。

WebSEAL は次の WebDAV メソッドを使用して要求を Junction アプリケーションに転送します。

- ACL
- BASELINE-CONTROL
- BCOPY
- BDELETE
- BIND
- BMOVE
- PROPPATCH
- CHECKIN
- CHECKOUT
- COPY
- LABEL
- LINK
- LOCK
- MERGE
- MKACTIVITY
- MKCOL
- MKREDIRECTREF
- MKWORKSPACE
- MOVE
- NOTIFY
- ORDERPATCH
- PATCH
- POLL
- PROPFIND
- PROPPATCH
- REBIND
- REPORT
- SEARCH
- SUBSCRIBE
- SUBSCRIPTIONS
- UNBIND
- UNCHECKOUT
- UNLINK
- UNLOCK
- UNSUBSCRIBE

- UPDATE
- UPDATEDIRECTREF
- VERSIONCONTROL

Microsoft RPC over HTTP のサポート

Microsoft RPC over HTTP は、HTTP 接続を介した RPC トラフィックの通信に使用されるプロトコルです。Microsoft Outlook クライアントは、このプロトコルを使用して HTTP を介して Microsoft Exchange サーバーにアクセスします。

WebSEAL は、ネイティブでは Microsoft RPC over HTTP 操作をサポートしていません。WebSEAL がサポートするのは、Junction アプリケーションへの RPC over HTTP バージョン 2 メソッドの転送です。すべての RPC over HTTP メソッドの許可は、*r* ACL ビットを使用して制御されます。

WebSEAL は、以下の RPC over HTTP メソッドを使用して、要求を Junction アプリケーションに転送します。

- RPC_IN_DATA
- RPC_OUT_DATA

チャンク転送コーディングのサポート

WebSEAL は、HTTP/1.1 規格で定義されるチャンク転送コーディングをサポートします。クライアントからのチャンクは、Junction Web サーバーに受信されたままの状態で送信されます。

インターネット・プロトコル・バージョン 6 (IPv6) のサポート

このセクションは、以下のトピックで構成されています。

- 『IPv4 および IPv6 の概要』
- 55 ページの 『IPv6 および IPv4 のサポートの構成』
- 56 ページの 『IPv6: 互換性のサポート』
- 56 ページの 『IPv6: アップグレードの注意』
- 57 ページの 『資格情報属性の IP レベル』

IPv4 および IPv6 の概要

Tivoli Access Manager for Web バージョン 6.0 以降、WebSEAL はインターネット・プロトコル・バージョン 6 (IPv6) をサポートしています。

IPv6 は IPv4 に比べて以下のように改善されています。

- IPv6 はアドレス・スペースに 128 ビット割り振ります。IPv4 はアドレス・スペースに 32 ビットしか割り振れません。
- IPv6 は、インターネット・バックボーン経由でパケットをルーティングするために使用する、静的なデフォルト以外のルーティング・テーブルのサイズを縮小できます。

- IPv6 は、IP セキュリティー・プロトコル (IPsec) の準拠を求めることによってエンドツーエンドのセキュリティを提供します。

IPv4 アドレスの基本形式は、4 つの数字がピリオドで区切られた 32 ビットの数値アドレスです。例:

x.x.x.x

各数字の有効範囲は、0 から 255 です。例:

1.160.10.240

IPv6 アドレスの基本形式の 1 つは、8 つの数字がコロンで区切られた 128 ビットの数値アドレスです。例:

x:x:x:x:x:x:x:x

各数字の有効範囲は、0 から ffff です (16 進数)。例:

fec0:fff:0000:0000:0000:0000:0000:1

IPv6 アドレスは、0 のみが含まれる連続したフィールドを縮小することによって、省略形式で表記することができます。例えば、0009:0000:0000:0000:0000:0008:0007:0006 は 9::8:7:6 と表現できます。

IPv6 アドレスの有効な表現の構成要素については、RFC 2373 規格を参照してください。IBM Security Access Manager for Web は、RFC 2373 のセクション 2.2 で説明されている IPv6 アドレスの有効な形式をいずれもサポートしています。ただし、IBM Security Access Manager for Web はネットマスクの接頭表記法はサポートしません。

IPv6 および IPv4 のサポートの構成

このタスクについて

WebSEAL 構成ファイルの [server] スタンザ内の **ipv6-support** スタンザ・エントリによって、WebSEAL で IPv4 または IPv6 をサポートするように構成できます。以下の値が有効です。

- **yes** - IPv6 および IPv4 の両ネットワークをサポートします (デフォルト設定)。

IPv6 ネットワークおよび IPv4 ネットワークの両方がサポートされる場合、WebSEAL は、IPV4 プロトコルまたは IPV6 プロトコルに基づく着信 HTTP 要求および HTTPS 要求を listen します。

junction のホスト名が IPv6 アドレスのみにマップされている場合は、IPv6 が使用されます。ホスト名が IPv4 アドレスにマップされている場合は、IPv4 アドレスが使用されます。DNS 名が IPv6 アドレスと IPv4 アドレスの両方にマップされる場合、選択は任意です。特定のプロトコルを指定するには、junction の作成時に、ホストの IPv6 または IPv4 アドレスを使用します。

マシン上に IPv6 対応のインターフェースがない場合、WebSEAL は IPv4 要求のみを listen します。IPv6 インターフェースおよび IPv4 インターフェースの両方

が使用可能な場合、WebSEAL はデフォルトでは両方のプロトコルの要求を listen します。WebSEAL が IPv6 接続を受け入れないようにしたい場合は、`ipv6-support = no` と設定します。

- **no** - IPv4 ネットワークのみをサポートします。

IPv6 サポートは、Security Access Manager WebSEAL バージョン 6.0 以降のインストールによってデフォルトで提供されます。

```
[server]
ipv6-support = yes
```

IPv6: 互換性のサポート

以下の条件は、前のバージョンの Security Access Manager で、IP バージョンの互換性がどのように維持されているかについて説明しています。

- **ipv6-support** が「no」に設定されている場合、WebSEAL は、前リリースと同様に IPv4 に対するサポートは提供しますが、IPv6 はサポートしません。

例えば、外部認証 C API (「*IBM Security Access Manager for Web: Web Security Developer Reference*」を参照) で使用される **iv-remote-address-ipv6** HTTP ヘッダーおよび XAUTHN_IPADDR_IPV6 ID は使用できません。

- **ipv6-support** が「yes」に設定されている場合、IPv6 はサポートされます。IPv4 アドレスを含む属性は、引き続き IPv4 アドレスを保有します。前のリリース用に作成されたカスタム・モジュールは、引き続き動作します。

ただし、WebSEAL が古いカスタム・モジュール (IPv6 形式をサポートするように作成されていない) に IPv6 限定アドレスを渡した場合、IPv6 アドレス形式を処理するには古いモジュールを更新する必要があります。

IPv6 のアドレス範囲は、IPv4 で使用可能な範囲よりもはるかに広がっています。古いモジュールを使用する際、WebSEAL は可能な場合、IPv6 アドレスを IPv4 形式にマップします。例えば、IPv6 アドレス `::c0a8:1` は IPv4 アドレス `192.168.0.1` にマップされます。

IPv6 アドレスが IPv4 の範囲を超えた場合、WebSEAL はデフォルトでアドレスを IPv4 形式の `0.0.0.0` にマップします。例えば、IPv6 アドレス `fec0::1` は IPv4 に相当するアドレスが存在しないため、IPv4 アドレス `0.0.0.0` にマップされます。

IPv6: アップグレードの注意

前のバージョンの Security Access Manager WebSEAL から WebSEAL バージョン 7.0 にアップグレードすると、IPv6 サポートは自動的に使用不可に設定されます。アップグレード・モジュールがインスタンス固有の構成ファイルに `ipv6-support = no` と設定します。アップグレード後、WebSEAL 構成ファイルを手動で編集して **ipv6-support** の値を「yes」に変更することによって、IPv6 のサポートを使用可能にすることができます。

資格情報属性の IP レベル

ネットワーク (IP) 情報は、ユーザーの資格情報に拡張属性として保管できます。IPv4 と IPv6 の情報を保有するために、異なる形式の構造を使用できます。資格情報に属性を追加すると、WebSEAL のパフォーマンスに影響を与える可能性があります。

WebSEAL 構成ファイルの **[server]** スタンザ内の **ip-support-level** スタンザ・エントリーによって、必要な IP レベルを指定して資格情報に保管するネットワーク情報の量を制御できます。以下の値が使用可能です。

- **displaced-only**

WebSEAL は、ユーザー資格情報の作成時と、外部認証 C API モジュールを使用したユーザーの認証時に、IPv4 属性のみを生成します。

マイグレーションによる WebSEAL インストールの場合 (ipv6-support=no)、この値がデフォルトで設定されます。

```
[server]
ip-support-level = displaced-only
```

この値は ipv6-support=yes の場合は設定できません。

- **generic-only**

WebSEAL は、ユーザー資格情報の作成時と、外部認証 C API モジュールを使用したユーザーの認証時のみに、IPv4 と IPv6 の両方をサポートする新規汎用属性を生成します。

新規 WebSEAL インストールの場合 (ipv6-support=yes)、この値がデフォルトで設定されます。

```
[server]
ip-support-level = generic-only
```

- **displaced-and-generic**

ユーザー資格情報の作成時と、外部認証 C API モジュールを使用したユーザーの認証時に、両方の属性タイプのセット (**displaced-only** および **generic-only** に よって生成) が使用されます。

```
[server]
ip-support-level = displaced-and-generic
```

LDAP ディレクトリー・サーバー構成

Security Access Manager が LDAP ベースのユーザー・レジストリーを使用するように構成されている場合 (IBM Tivoli Directory Server など)、WebSEAL を LDAP クライアントとして構成し、LDAP サーバーと通信できるようにする必要があります。

LDAP サーバー (およびその構成ファイル ldap.conf) の場所は、Security Access Manager のランタイム構成中に提供されます。スタンザ・エントリーと ldap.conf および WebSEAL 構成ファイル (webseald.conf) の値の組み合わせによって、WebSEAL が LDAP クライアントを実行するための適切な情報が提供されます。

- WebSEAL は、構成済みユーザー・レジストリーが LDAP ベースのディレクトリー・サーバーであることを判別します。
- webseald.conf の **[ldap]** スタンザ内の以下のスタンザ・エントリーが有効です。

```
enabled
host
port
ssl-port
max-search-size
replica
ldap-server-config
auth-using-compare
cache-enabled
prefer-readwrite-server
bind-dn
bind-pwd
ssl-enabled
ssl-keyfile
ssl-keyfile-dn
timeout
auth-timeout
search-timeout
default-policy-override-support
user-and-group-in-same-suffix
login-failures-persistent
```

- **enabled** スタンザ・エントリーの値が ldap.conf から読み込まれます。値が「yes」の場合、webseald.conf の **[ldap]** スタンザに、**bind-dn** および **bind-pwd** スタンザ・エントリー（およびその値）を指定する必要があります。
- また、ldap.conf 内の以下のスタンザ・エントリーの値は、webseald.conf 内の既存の値をオーバーライドします。

```
enabled
host
port
ssl-port
max-search-size
replica
```

ここで説明したスタンザ・エントリーについては、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

ワーカー・スレッドの割り振り

このセクションは、以下のトピックで構成されています。

- 『WebSEAL ワーカー・スレッド構成』
- 59 ページの『junction 用のワーカー・スレッドの割り振り (Junction フェアネス)』

WebSEAL ワーカー・スレッド構成

構成されたワーカー・スレッドの数は、サーバーがサービスできる同時着信要求の数を指定します。すべてのワーカー・スレッドが使用中である場合に到着した他の接続は、ワーカー・スレッドが使用可能になるまでバッファーに入れられます。

WebSEAL への着信接続に対してサービスを提供できる使用可能なスレッドの数を指定できます。ワーカー・スレッドの数はパフォーマンスに影響する可能性があるため、その構成は注意深く行う必要があります。

この構成スタンザ・エントリーが同時接続の数に上限を設けることはありません。このスタンザ・エントリーは、潜在的に無限の作業キューにサービスするために使用可能にされるスレッドの数を指定するだけです。

ワーカー・スレッドの最適数の選択は、ネットワーク上のトラフィックの量とタイプについての知識に基づいて行います。どの場合も、必要なのは、オペレーティング・システムが課するワーカー・スレッドの制限数より少ない値のみを入力するということです。

スレッドの数を増やせば、一般的には、要求処理の完了にかかる平均時間が短縮されることになります。ただし、スレッドの数を増やすと、他の要因にも影響が及び、そのためにサーバー・パフォーマンスに悪影響を生じる恐れがあります。

WebSEAL は、TCP または SSL を使用するクライアントからの要求を処理するための、単一の総称ワーカー・リストおよびワーカー・スレッド・プールを保持しています。この拡張メカニズムがあるため、WebSEAL では、取り扱うロードが大幅に増えても、使用するシステム・リソースは少なく済みます。

ワーカー・スレッド・プールのサイズは、WebSEAL 構成ファイルの **[server]** スタンザ内で、**worker-threads** スタンザ・エントリーを設定することによって構成できます。

```
[server]
worker-threads = 50
```

注: このスタンザ・エントリーの値は、オペレーティング・システムが設定するワーカー・スレッド数の制限範囲内の値でなければなりません。

junction 用のワーカー・スレッドの割り振り (Junction フェアネス)

複数の junction にわたる要求を処理するために使用する WebSEAL ワーカー・スレッドの割り振りは、グローバルでも、個々の junction 単位でも構成できます。構成メカニズムは、すべての junction 間でのワーカー・スレッドの「フェア」な配分を維持し、どれか 1 つの junction によってワーカー・スレッド・プールが消費し尽されないようにします。

junction フェアネスの概念

WebSEAL は、ワーカー・スレッドのプールからスレッドを取り出して、複数の要求を処理します。WebSEAL が使用できるワーカー・スレッドの数は、WebSEAL 構成ファイル内の **worker-threads** スタンザ・エントリーで指定します。

worker-threads は、特定の WebSEAL インプリメンテーションに最も適した値に調整できます。着信要求を処理するために使用できるワーカー・スレッドがないと、WebSEAL サーバーがユーザーに応答しなくなることがあります。

ワーカー・スレッドは、junction された複数のバックエンド・サーバー上のアプリケーションに対する着信要求を処理するために使用されます。しかし、大量の要求に回答しそれを処理するために特定のバックエンド・アプリケーションの速度が異常に低下した場合に、ワーカー・スレッド・プールが早く消耗してしまふことがあります。このように 1 つのアプリケーションがワーカー・スレッド・プールを消費し尽くしてしまった場合は、WebSEAL は、junction されたその他のアプリケーション・サーバーのサービスを求める要求に回答できなくなります。

複数の junction 上のアプリケーションにサービスを提供するために使用するワーカー・スレッドの数の制限は、グローバルまたは個々の junction 単位で構成できます。この制限を設定することにより、すべての junction にわたる「フェアネス」を確保し、どれか 1 つのアプリケーションが自身に割り当てられている数を超えるワーカー・スレッドを要求するのを防ぐことができます。

注: ワーカー・スレッドのリソース使用制限およびワーカー・スレッドの不足状態の検出方法については、「*IBM Security Access Manager for Web Performance Tuning Guide*」を参照してください。

junction 用のワーカー・スレッドのグローバル割り振り

特定の WebSEAL サーバー用のすべての junction を対象としたワーカー・スレッドのグローバル割り振りをコントロールするには、WebSEAL 構成ファイルの **[junction]** スタンザ内にある 2 つのスタンザ・エントリーを使用します。これらのスタンザ・エントリーの値は、0 から 100 の範囲内のパーセンテージで表します。

- **worker-thread-soft-limit**

このスタンザ・エントリーは、「ハード」の制限に達する前に警告を送るために設定します。**worker-thread-soft-limit** を超過すると、WebSEAL エラー・ログ・ファイルに警告メッセージが送られます (30 秒ごと)。

例えば、**worker-threads=50** の場合に、このパラメーターを 60 (%) に設定したとすれば、junction が 30 個を超えるワーカー・スレッドを消費すると、警告メッセージが発行されます。しかし、「ハード」制限に達するまでは、30 個のワーカー・スレッドを超える要求もすべて処理されます。

デフォルト値は 90% です。

- **worker-thread-hard-limit**

このスタンザ・エントリーは、junction 経由の要求に対するサービスを提供するときのカットオフ・ポイントを決定します。**worker-thread-hard-limit** を超過すると、WebSEAL エラー・ログ・ファイルにエラー・メッセージが送られます (30 秒ごと)。さらに、ユーザーには、503 「サービスが使用できません」メッセージが送られます。

worker-threads=50 の場合にこのパラメーターを 80 (%) に設定したとすれば、junction が 40 個を超えるワーカー・スレッドを消費しようとする、エラー・メッセージが発行されます。junction 上の 40 個を超えるワーカー・スレッドを必要とするすべての要求に対して、503 「サービスが使用できません」メッセージが戻されます。

100 (%) というデフォルト値は無制限を意味します。

これらのグローバル設定は、構成されているすべての `junction` に等しく適用されます。当然のことながら、この 2 つのスタンザ・エントリーを構成するときは、「ソフト」制限を「ハード」制限より小さい値に設定するようにしてください。

junction 用のワーカー・スレッドの junction 単位での割り振り

ワーカー・スレッドの消費数の制限は、個々の `junction` 単位で設定することもできます。特定の `junction` についてワーカー・スレッド数のハード制限とソフト制限を指定するには、`pdadmin server task create` コマンドで以下に示すオプションを使用します。

- `-I percent_value`

このオプションは、`junction` について、ワーカー・スレッドの消費数に対するソフト制限を定義する値 (パーセント値) を設定します。グローバルのソフト制限の設定の場合と同様に、このオプションの場合も、`junction` が設定値を超えるワーカー・スレッドを消費すると、警告メッセージが発行されます。

- `-L percent_value`

このオプションは、`junction` について、ワーカー・スレッドの消費数に対するハード制限を定義する値 (パーセント値) を設定します。グローバルのハード制限の設定の場合と同様に、このオプションの場合も、`junction` が設定値を超えるワーカー・スレッドを消費しようとする、警告メッセージが発行されます。さらに、ユーザーには、503「サービスが使用できません」メッセージが送られます。

例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task webseald-<server-name> create -t tcp -h <host-name>
-I 60 -L 80 jct-point
```

`junction` 単位の設定は、常に WebSEAL 構成ファイル内のグローバル設定をオーバーライドします。個々の `junction` で不適切な値を設定すると、グローバル設定により確立されているポリシーに悪影響を与えるおそれがあります。

トラブルシューティングに関する注意事項

- `pdadmin server task show` コマンドを使用して、特定 `junction` に関するアクティブなワーカー・スレッドの数を表示することができます。

```
pdadmin> server task webseald-<server-name> show /<jct-point>
```

この情報は、割り当て数を超えるワーカー・スレッド・リソースを消費している `junction` の場所を判別したいときに役立ちます。

- 特定 `junction` について、ハード制限値よりも大きなソフト制限値を指定した場合は、その `junction` は作成されません。
- 1 つの `junction` について、ソフト制限とハード制限の両方の値 (`-I` オプションと `-L` オプションの両方) を指定する必要があります。

HTTP データ圧縮

WebSEAL サーバーは、HTTP を介して WebSEAL サーバーとクライアントとの間で転送されるデータを圧縮するように構成できます。WebSEAL は、RFC 1952 で説明されている **gzip** 圧縮アルゴリズムを使用します。Gzip は、主なブラウザのすべてでサポートされています。

WebSEAL における HTTP 圧縮は、MIME タイプ、ブラウザ・タイプ、および保護オブジェクト・ポリシー (POP) のそれぞれに基づいて構成できます。

このセクションは、以下のトピックで構成されています。

- 『MIME タイプに基づく圧縮』
- 63 ページの『ユーザー・エージェント・タイプに基づく圧縮』
- 64 ページの『POP 内の圧縮ポリシー』
- 64 ページの『データ圧縮の制限』
- 65 ページの『データ圧縮ポリシーの構成』

MIME タイプに基づく圧縮

データ圧縮が必要なそれぞれの MIME タイプごとに、あるいは、MIME タイプのグループ用に、WebSEAL 構成ファイルの中にエントリーを作成できます。構文は次のようになります。

```
[compress-mime-types]
mime_type = minimum_doc_size[:compression_level]
```

mime-type は、1 つの特定の MIME タイプ、または、ワイルドカード文字を使用して MIME タイプのクラスを指定できます。それぞれの MIME タイプ宣言は、**[compress-mime-types]** スタンザの中で別々のエントリーとして表されます。ワイルドカード文字 (*) は、複数の MIME タイプの 1 つの集合を表すエントリーにのみ使用できます。例えば、`text/*` などです。このスタンザにリストされていない MIME タイプは圧縮されません。順序が重要です。戻された文書に一致する最初のエントリーが、その文書に使用されます。

minimum_doc_size 値は、どのサイズの文書を圧縮するかに関するポリシーを指定します。この値は整数です。有効な値を、次のリストに示します。

- -1

最小サイズが -1 であるときは、指定した MIME タイプの文書は圧縮されません。

- 0

最小サイズが 0 であるときは、指定した MIME タイプの文書は必ず圧縮されます。

- ゼロより大きい整数

最小サイズがゼロより大きいときは、WebSEAL に対する応答の中のバイト数がこの整数値を超える場合に、指定した MIME タイプの文書が圧縮されます。

-1 以外の負の数値を使用すると、エラー・メッセージが生成されます。

WebSEAL がブラウザから要求を受け取ると、サーバーは、HTTP ヘッダーの中の「内容の長さ」フィールドを調べて着信データのサイズを判別します。ただし、すべての HTTP 応答に「内容の長さ」フィールドがあるわけではありません。「内容の長さ」フィールドがないときは、該当の MIME タイプが圧縮を行わない (*minimum_doc_size* が -1) ように構成されていない限り、WebSEAL は文書を圧縮します。

compression_level はオプションの設定で、データ圧縮のレベルを指定します。有効な値は 1 から 9 の整数で、整数が大きいほど圧縮の量が多くなります。ただし、圧縮量が大きいほど、CPU の負荷が大きくなることに注意してください。圧縮量を増やすときは、パフォーマンスに与える影響を考慮に入れる必要があります。

compression_level が指定されないときは、1 というデフォルトのレベルが使用されます。

次の例では、1000 バイトよりも大きいサイズの文書がすべて圧縮されます。

```
[compress-mime-type]
*/* = 1000
```

次のセットのエントリを設定すると、すべてのイメージの圧縮を使用不可にし、CSS ファイルの圧縮を使用不可にし、すべての PDF 文書のレベル 5 での圧縮を使用可能にし、2000 バイトより大きいサイズの HTML 文書の圧縮を使用可能にし、さらに、その他のテキスト文書の圧縮をサイズに関係なく使用可能にします。

```
[compress-mime-type]

image/* = -1
text/css = -1
application/pdf = 0:5
text/html = 2000
text/* = 0
```

ユーザー・エージェント・タイプに基づく圧縮

ユーザー・エージェント は、要求を開始するクライアントのことです。ユーザー・エージェントの例には、ブラウザ、エディター、スパイダー (Web のリンクを張るロボット)、またはその他のエンド・ユーザー・ツールなどがあります。

WebSEAL は、圧縮データを要求するユーザー・エージェントに圧縮データを返します。WebSEAL は、圧縮データを要求しないユーザー・エージェントには、圧縮データを返しません。しかし、ユーザー・エージェントの一部には、圧縮データを要求することはできるが、データを正しく処理できないものがあります。WebSEAL アドミニストレータは、WebSEAL 構成ファイルを使用して、さまざまなブラウザの圧縮を明示的に使用可能あるいは使用不可にすることができます。

構成ファイルのエントリの構文は次のようになります。

```
[compress-user-agents]
user_agent_pattern = {yes|no}
```

user_agent_pattern は、WebSEAL に送られたユーザー・エージェント・ヘッダーにある文字に一致するワイルドカード・パターンで構成されます。yes という値は、ブラウザに戻されるデータを圧縮することを意味します。no という値は、データを圧縮しないままです。

ユーザー・エージェント・ヘッダーが、WebSEAL 構成ファイルにあるスタンザ・エントリーのいずれにも一致しない場合は、WebSEAL は、ブラウザから送られた「エンコードを受け入れる」ヘッダーを使用します。

例えば、次のエントリーは、Internet Explorer 6 の圧縮を使用可能にしますが、その他のすべてのブラウザの圧縮を使用不可にします。

```
[compress-user-agents]
*MSIE 6.0" = yes
* = no
```

POP 内の圧縮ポリシー

圧縮しない という圧縮ポリシーを、保護オブジェクト・ポリシー (POP) 内に指定することができます。この POP を 1 つのオブジェクトに付加することができ、そのようにすると、WebSEAL はそのオブジェクトの圧縮を使用不可にします。このポリシーを指定するには、以下の属性を POP に追加します。

```
document-compression = no
```

この属性セットがない POP、あるいは、この属性セットが no 以外の任意の値 に設定されている POP では、文書が圧縮されます。

例えば、junction /appOne の圧縮を使用不可にするには、次のようにします。

```
pdadmin> pop create appOnePop
pdadmin> pop modify appOnePop set attribute document-compression no
pdadmin> pop attach /WebSEAL/host/appOne appOnePop
```

/appOne の下の、オーバーライドする POP を持つサブディレクトリーの圧縮をできるようにするには、document-compression 属性を持たない別の POP を付加します。例:

```
pdadmin> pop create dataPop
pdadmin> pop attach /WebSEAL/host/appOne/data dataPop
```

圧縮ポリシーを適用するこのメソッドは、URL で使用できます。例えば、URL に適用されたワイルドカード・パターンに基づいて圧縮を使用不可にするには、dynurl を使用できます。照会ストリングに特定の引数を持つ junction へのすべての要求の圧縮を使用不可にするには、以下のようなエントリーを持つ dynurl.conf ファイルを作成します。

```
/disableCompression      /appOne/*¥?want-response=text/xml
```

次に、document-compression 属性を no に設定した POP を /WebSEAL/host/disableCompression に付加します。

データ圧縮の制限

WebSEAL とバックエンド・サーバーとの間で圧縮データを転送することはサポートされていません。

WebSEAL は、さまざまな目的のために、URL に対してフィルター操作を実行します。WebSEAL は、圧縮データに対してはフィルター操作を実行しません。

データ圧縮ポリシーの構成

このタスクについて

WebSEAL とクライアント・ブラウザとの間の通信のためのデータ圧縮ポリシーを指定するには、以下のステップを実行します。

手順

1. WebSEAL 構成ファイルで、データ圧縮ポリシーが適用されるそれぞれの MIME タイプを指定します。ポリシーを実行する値を割り当てます。

```
[compress-mime-types] mime_type = minimum_doc_size
```

デフォルト設定は、すべてのデータを圧縮しないままにしておくことに注意してください。

```
[compress-mime-types]  
*/* = -1
```

このスタンザにエントリーを作成することについて詳しくは、62 ページの『MIME タイプに基づく圧縮』を参照してください。

2. WebSEAL 構成ファイルで、データ圧縮ポリシーが適用されるそれぞれのタイプのユーザー・エージェント (ブラウザ) を指定します。値 `yes` を割り当てて、データ圧縮の使用可能にします。値 `no` を割り当てて、データ圧縮を使用不可にします。

```
[compress-user-agents]  
user_agent = {yes|no}
```

デフォルトでは、エントリーは設定されません。ユーザー・エージェントの「エンコードを受け入れる」ヘッダーに一致するエントリーがないときは、「エンコードを受け入れる」ヘッダーの値が使用されます。このスタンザでのエントリーの作成について詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の `[compress-user-agents]` スタンザを参照してください。

3. オプションで、POP に圧縮ポリシーを指定し、その POP を保護オブジェクト・スペースの中の該当オブジェクトに適用します。詳細については、64 ページの『POP 内の圧縮ポリシー』を参照してください。

UTF-8 でのマルチロケール・サポート

このセクションは、以下のトピックで構成されます。

- 『マルチロケール・サポートの概念』
- 70 ページの『マルチロケール・サポートの構成』

マルチロケール・サポートの概念

このセクションは、以下のトピックで構成されています。

- 66 ページの『UTF-8 を使用した WebSEAL データ処理』
- 67 ページの『ユーザー・レジストリー構成への UTF-8 の依存関係』
- 67 ページの『UTF-8 データ変換の問題』

- 68 ページの『認証に与える UTF-8 の影響』
- 69 ページの『認可に与える UTF-8 の影響 (動的 URL)』
- 70 ページの『URL の使用は 1 つのエンコード・タイプのみ』

UTF-8 を使用した WebSEAL データ処理

WebSEAL は、内部ですべてのデータを UCS Transformation Format 8 バイト (UTF-8) エンコード方式を使用して維持し処理することによって、マルチロケール・サポートをインプリメントします。UTF-8 は、可変幅を持つマルチバイト・コード・ページです。

WebSEAL は、すべての内部データ処理用のデフォルト・コード・ページとして UTF-8 を使用しています。このサポートによって、WebSEAL は、同時に複数の言語のデータを処理できるようになります。

WebSEAL アドミニストレーターは、WebSEAL がどのようにデータ入力とデータ出力を処理するかを構成することができます。データ入力には、例えば、ユーザーがログインしてデータを形成するときに、ブラウザによって WebSEAL に送られる文字があります。データ出力の例には、Security Access Manager イベント・ロギング・マネージャーによってファイル・システムに書き込まれるロギング情報があります。

WebSEAL は、WebSEAL プロセスが実行されているロケールに関係なく、内部でデータを UTF-8 を使用して処理します。入力または出力としてロケール固有のデータが必要になると、どのロケールで WebSEAL プロセスが実行されているかが重要になります。ほとんどのオペレーティング・システムでは、デフォルトでは UTF-8 を使用していません。ロケール固有の振る舞いを期待するアドミニストレーターは、どのロケールが使用されているかを知ることが必要であり、さらに、必要な振る舞いに一致するように WebSEAL UTF-8 構成オプションを設定する必要があります。

システム・ロケールは、言語およびローカル・コード・ページという 2 つの部分から成っています。ローカル・コード・ページは UTF-8 でも UTF-8 でなくてもかまいません。従来、ほとんどのオペレーティング・システムでは、UTF-8 ではないローカル・コード・ページが使用されています。例えば、米国英語用の 8 ビット ASCII 文字セットを表すために使用されている共通のローカル・コード・ページは en_US.ISO88591 であり、これは、ISO-8859-1 文字セットを使用しています。

クライアントの要求を処理し、ローカル・コード・ページでデータを形成する必要があるシステムを実行しているアドミニストレーターは、URL サポート (**utf8-url-support-enabled**) およびフォーム・サポート (**utf8-forms-support-enabled**) のデフォルトの設定値を変更できます。デフォルトの WebSEAL 設定では、データを UTF-8 形式でしか処理しません。

例えば、クライアント要求を処理し、UTF-8 ではないローカル・コード・ページを使用してデータを形成するシステムについては、デフォルト設定を変更する必要があります。例:

- スペイン語、フランス語、ドイツ語のような単一バイトのラテン語文字セット。
- 日本語、中国語のようなマルチバイトの文字セット。

複数の言語でユーザーとデータを扱うために、真の意味でマルチロケール・サポートを提供する必要があるシステムを実行している場合は、以下の設定を確認します。

- ご使用のローカル・コード・ページ設定。UTF-8 コード・ページに変換することを確認してください。
- デフォルトの WebSEAL マルチロケール UTF-8 設定。

ご使用のデプロイメントに最適なように、これらの構成設定をカスタマイズできます。

ユーザー・レジストリー構成への UTF-8 の依存関係

最適なマルチロケール・サポートを使用できるようにするには、すべてのユーザーを、ユーザーが使用する言語に関係なく、1 つの共通ユーザー・レジストリーに保管します。

ほとんどのユーザー・レジストリーは、デフォルトで UTF-8 をサポートしています。LDAP ユーザー・レジストリーの一部、およびそれをサポートするデータベースを、オプションで UTF-8 をサポートしないように構成することもできます。Security Access Manager で使用されている LDAP ユーザー・レジストリーおよびデータベースが UTF-8 を使用していることを確認してください。

IBM Tivoli Directory Server は、デフォルトで UTF-8 を使用するよう構成されています。

UTF-8 データ変換の問題

WebSEAL は、ローカル・コード・ページが UTF-8 を使用している環境にデプロイできます。また、同様に、WebSEAL を、ローカル・コード・ページが UTF-8 を使用していない環境にデプロイすることもできます。UTF-8 以外のローカル・コード・ページを使用しているオペレーティング・システム環境で WebSEAL を使用すると、データの入力と出力の際に WebSEAL がデータを変換する必要があります。WebSEAL がデータを読み取るときに、データを非 UTF-8 から UTF-8 に変換しなければなりません。WebSEAL がデータを書き込むときに、データを UTF-8 から非 UTF-8 に変換しなければなりません。

ローカル・コード・ページへの変換が必要な場合、UTF-8 ロケールで実行するときはデータ損失が起こることはありません。

UTF-8 ロケールから非 UTF-8 ロケール (ローカル・コード・ページ) への変換では、場合によっては、データ損失が起こります。

UTF-8 から非 UTF-8 ロケールへのデータの変換では、データ損失が起こります。例えば、WebSEAL が en_US.ISO8859 環境で実行していて、日本語のユーザー名をローカル・コード・ページに変換しなければならない場合、変換の結果は疑問符 (?) のストリングになります。この結果は、ISO-8859-1 では、日本語の文字を表す方法がないためです。このため、WebSEAL は UTF-8 を使用して稼働する必要があります。

管理コマンド (**pdadmin**) を非 UTF-8 環境から実行すると、データ損失が発生するリスクがあります。バージョン 5.1 より前では、WebSEAL は常に **pdadmin** ユー

ティリティーと同じロケールで実行されていました。マルチロケール・サポートができるようになり、WebSEAL は、別のロケールで実行できます。WebSEAL がアドミニストレーターにメッセージを戻すときは、アドミニストレーターが選択した言語で戻す必要があります。これを行うには、WebSEAL は、**pdadmin** によって提示されるロケールが決めた該当の言語バックからメッセージを入手します。すべてのメッセージは UTF-8 で伝送されますが、**pdadmin** は、これらのメッセージをローカル・コード・ページに変換してから表示します。したがって、ローカル・コード・ページが UTF-8 でないときは、データ損失が起こり得ます。**pdadmin** が UTF-8 環境で実行されるときは、データ損失は起こりません。

WebSEAL は、ロギング・データおよび監査データを UTF-8 を使用して生成します。起こりうるデータ損失を回避するには、UTF-8 を使用してデータを該当のロギング・ファイルおよび監査ファイルに書き込みます。ローカル・コード・ページが UTF-8 でないときは、データを書き込む前に 非 UTF-8 に変換しなければなりません。この場合、データ損失が起こる可能性があります。

WebSEAL で生成されるログ監査ファイルはすべて、サーバーが実行されるロケールによって指定された言語で生成されます。メッセージを書き込むために使用されるコード・ページは、WebSEAL ルーティング・ファイルの中に構成することが可能です。例えば、UNIX および Linux システムでは、このファイルは `/opt/pdweb/etc/routing` になります。

認証に与える UTF-8 の影響

内部データの処理に UTF-8 を使用すると、WebSEAL による証明書要求の処理に以下の影響があります。

- 基本認証での UTF-8 ログインはサポートされていません。

基本認証 (BA) ログインで UTF-8 を使用することはサポートされていません。BA で UTF-8 ログインがサポートされていないのは、矛盾する方法でブラウザーがデータを伝送するためです。ブラウザーの矛盾のため、WebSEAL はこれまで常にマルチバイト BA ログインはサポートしていませんでした。

WebSEAL は、BA ログイン・ストリングを、ストリングがローカル・コード・ページにあると想定して読み取ります。WebSEAL は、7 ビット ASCII および単一バイトのラテン語コード・ページをサポートします。例えば、フランス語ユーザーが BA ログインを使用することを許可する必要があるサーバーは、ラテン語のロケールで実行しなければなりません。WebSEAL は BA ログイン・ストリングを読み取ってそれを内部で UTF-8 に変換します。しかし、フランス語のユーザーが UTF-8 コード・ページを使用している場合、ログイン・ストリングがマルチバイトになるため、BA ログインは使用できなくなります。

- フォーム・ログイン

WebSEAL の前のバージョンでは、フォーム・ログインのデータは、WebSEAL によって、必ず `auto` 機能性を使用して読み取られていました。つまり、WebSEAL は、ログイン・データが UTF-8 形式であるかどうかを調べていました。ログイン・データが UTF-8 形式でない場合は、ローカル・コード・ページとして処理されました。

WebSEAL バージョン 5.1 以上では、72 ページの『POST 本体情報内の UTF-8 サポート (フォーム)』で説明するように、この設定は構成可能になっています。

- クロスドメイン・シングル・サインオン、e-community シングル・サインオン、およびフェイルオーバー認証

これらの認証方式は、すべて、エンコードされたトークンを使用しています。これらのトークンのエンコードは、UTF-8 エンコード方式あるいは UTF-8 以外のエンコード方式のいずれかを使用するように構成する必要があります。詳細については、70 ページの『マルチロケール・サポートの構成』を参照してください。

認可に与える UTF-8 の影響 (動的 URL)

WebSEAL は、許可検査を必要とするすべての要求を、UTF-8 または WebSEAL ホストのロケール設定を使用する要求に制限します。すべてのバックエンド・サーバーも、これらの設定に制約されます。WebSEAL は、既知の保護オブジェクトにセキュリティ・ポリシーを適用できるように、この制限を施行する必要があります。

この制限は、WebSEAL 動的 URL 機能をいつ使用可能にするかを検討する際に、特に重要です。WebSEAL 動的 URL は、POST 本体および照会ストリングからのデータを処理します。文字パターンから認可オブジェクトへのマッピングを正常に行うためには、POST 本文および照会ストリングからのデータは、WebSEAL に認識されている文字エンコード方式を使用している必要があります。

設計上、WebSEAL 動的 URL は、Web アプリケーション・インターフェース宛ての動的データが配置された、要求の照会ストリング部分を処理します。GET 要求の標準は、この照会ストリング・フォーマットを使用します。動的 URL の照会ストリング要件をサポートするため、WebSEAL は POST 要求の本体に含まれるデータを、照会ストリング・フォーマットに変換します。

動的 URL が使用可能である場合、WebSEAL は、照会ストリングからのデータを保護 (アクセス・コントロール) が必要なオブジェクトにマップします。安全に照会ストリングをオブジェクトにマップするには、ストリング・データには、WebSEAL およびバックエンド・アプリケーション・サーバーに認識される同一の文字セットを使用する必要があります。同一の文字セットを使用しない場合、バックエンド・アプリケーションに受け入れられるが WebSEAL には受け入れられない文字を使用する要求によって、動的 URL アクセス・コントロールが迂回される可能性があります。UTF-8 ではなく、WebSEAL が実行されている文字セットのものでもない文字を使用した動的データを (POST 本体または照会ストリングで) WebSEAL が受け取った場合、WebSEAL は要求をリジェクトし、エラーを戻します。

WebSEAL (動的 URL が使用可能) が UTF-8 以外の環境で実行されており、要求 POST 本体または照会ストリングに UTF-8 文字が含まれている場合、WebSEAL がこれらの要求の UTF-8 コーディングをデコードできるように、WebSEAL 構成ファイルの [server] スタンザ内の **utf8-form-support-enabled** スタンザ・エントリを構成することができます。

動的 URL について詳しくは、685 ページの『第 38 章 動的 URL』を参照してください。

URL の使用は 1 つのエンコード・タイプのみ

WebSEAL では、処理のために提示される URL では 1 つのタイプの文字エンコード方式 (UTF-8 または ShiftJIS) のみを使用していることを必要とします。URL で複数のタイプの文字エンコード方式を使用していると、WebSEAL は、要求内のデータの正確性を保証できません。これは、UTF-8 文字のデコードされた値が、ローカル・コード・ページ内の同じ文字のデコードされた値に一致しない場合があるからです。このような不正確性がデータにあると、WebSEAL が、許可されていないユーザーに、保護オブジェクトへのアクセスを誤って認可してしまう原因になります。

WebSEAL が、複数の文字エンコード方式タイプを持つ URL を検出すると、URL は無効な要求として戻されます。

マルチロケール・サポートの構成

このセクションは、以下のトピックで構成されます。

- 『Uniform Resource Locator (URL) 用の UTF-8 サポート』
- 72 ページの『POST 本体情報内の UTF-8 サポート (フォーム)』
- 73 ページの『照会ストリングにおける UTF-8 サポート』
- 74 ページの『クロスドメイン・シングル・サインオン用のトークンの UTF-8 エンコード』
- 74 ページの『e-community シングル・サインオン用のトークンの UTF-8 エンコード』
- 75 ページの『フェイルオーバー認証用の Cookie の UTF-8 エンコード』
- 75 ページの『junction 要求における UTF-8 エンコード』

Uniform Resource Locator (URL) 用の UTF-8 サポート

ブラウザーで使用できる文字セットは、URL 内での使用が有効とされているものに限定されています。この有効範囲は、ASCII 文字セット内の印刷可能文字 (16 進コード 0x20 から 0x7e までの間) です。URL では、英語以外の言語用として、またはその他の目的のために、印刷可能な ASCII 文字セットの範囲には含まれていない文字が必要になることがよくあります。このような文字は、伝送および解釈用の印刷可能文字を使用してエンコードすることができます。

印刷可能な ASCII 文字セットの範囲外の文字を伝送するためのエンコード方式はいくつかあります。Web プロキシとしての役割を果たす WebSEAL は、これらすべてのケースに対応できることが必要です。UTF-8 ロケール・サポートは、この必要性に対応できます。

WebSEAL がブラウザーからの URL を処理する方法は、次に示すように、WebSEAL 構成ファイルで指定できます。

```
[server]
utf8-url-support-enabled = {yes|no|auto}
```

可能な 3 つの値は、次のとおりです。

- はい

このモードでは、WebSEAL は、URL ストリングの中の URI エンコード UTF-8 データのみを認識し、変更せずに使用します。そして、UTF-8 文字は、URL へのアクセス権を決定する際に、検証されて考慮されます。WebSEAL は、ロー UTF-8、および URL 内の URI エンコード UTF-8 ストリングの両方をサポートします。このモードでは、その他のエンコード技法は受け入れられません。

これがデフォルト値で、大半の環境で使用できます。

7 ビット ASCII 英語ロケールで実行されるサーバーは、この値を使用する必要があります。

- **no**

このモードでは、WebSEAL は、URL ストリング内の UTF-8 形式のデータを認識しません。ローカル・コード・ページのみで使用されます。ストリングが妥当性検査にパスした場合は、内部使用のために UTF-8 に変換されます。

マルチバイト入力データを処理する必要がなく、単一バイトのラテン語ロケール (フランス語、ドイツ語、スペイン語など) で実行されるサーバーはこの設定を使用します。

アプリケーションをサポートしているとき、および UTF-8 サポートが使用可能になっている場合に Web サーバーが WebSEAL で正しく作動しないときには、この設定を使用します。これらのアプリケーションが、URL で DBCS (Shift-JIS など) またはその他のエンコード・メカニズムを使用している可能性があります。

注: この値を no に設定するときは、junction されたすべてのサーバーが、UTF-8 形式の URL を受け入れないようにします。セキュリティの観点からは、WebSEAL が、junction 先サーバーと同じ方法で URL を解釈することが重要です。

- **auto**

WebSEAL は、UTF-8 と他の形式の言語文字エンコードを区別しようとします。WebSEAL は、正しく構成された UTF-8 エンコードを正しく処理します。エンコード方式が UTF-8 と認められない場合は、コーディングは DBCS または Unicode として処理されます。

URL が「%uHHHH」という形式の Unicode を使用している場合、WebSEAL はこれを UTF-8 に変換します。その他のデコード処理は、構成設定が yes である場合と同様に進みます。[server] スタンザの **double-byte-encoding** オプションが yes に設定されている場合、WebSEAL は %HH%HH を UTF-8 に変換します。

単一バイト・ラテン語ロケールで実行され、マルチバイト・ストリングを処理する必要があるサーバーは、auto 設定を使用する必要があります。

マルチバイト・ロケールで実行されているが、1 つの言語 (例えば、日本語) だけをサポートすれば済むサーバーは、auto 設定を使用できます。

お勧めするデプロイメント方針は、以下のようになります。

1. 既存の実動デプロイメントにおける **default-webseal** ACL をただちにチェックし、コンテンツ上の目的のために必要でない限り、この ACL を、非認証「r」

アクセスを受け入れないように設定します。この設定により、機密漏れを Security Access Manager ドメイン内の有効アカウントを持つユーザーに制限することができます。

2. **utf8-url-support-enabled** スタンザ・エントリーを、必ずデフォルト値の **yes** に設定します。
3. アプリケーションをテストします。そして、アプリケーションが正しく機能している場合は、この設定を使用します。
4. アプリケーションが「無効な要求」エラーで失敗した場合、**utf8-url-support-enabled** スタンザ・エントリーを **no** に設定して、そのアプリケーションを再試行してください。これが成功した場合、この設定を使用してデプロイできます。ただし、junction 先 Web サーバーが、いずれも、UTF-8 でエンコードされた URL を受け入れるように構成されていないことを確認してください。
5. アプリケーションの問題が解決しない場合は、**utf8-url-support-enabled** を **auto** に設定してみてください。

POST 本体情報内の UTF-8 サポート (フォーム)

WebSEAL がフォーム (例えば、ログイン・フォーム) からの情報を含む POST 本体のデータを処理する方法は、次に示すように、WebSEAL 構成ファイルで指定できます。

```
[server]
utf8-form-support-enabled = {yes|no|auto}
```

サーバーにデータを提供するフォームは、例えばログイン・フォームのように、WebSEAL の一部を成すフォームです。これらのフォームはすべて、文字セットが UTF-8 であると宣言しています。したがって、デフォルト値は **yes** です。アドミニストレーターがこれらのフォームを編集して文字セットを非 UTF-8 設定 (例えばローカル・コード・ページ) に変更した場合は、この構成の設定も変更する必要があります。フォームの一部が UTF-8 を使用し、別のフォームがローカル・コード・ページを使用する場合は、値 **auto** を使用します。すべてのフォームが非 UTF-8 設定を使用するように変更された場合は、値 **no** を使用します。

可能な 3 つの値は、次のとおりです。

- **はい**

WebSEAL は、フォームの UTF-8 エンコードのみを認識し、データは変更せずに使用されます。これらの UTF-8 文字は、妥当性検査され、データを処理する際に考慮されます。その他のエンコード技法は受け入れられません。

double-byte-encoding が **yes** に設定されていると、%HH%HH という形式の Unicode がサポートされます。2 バイトの Unicode 文字が検出されるときは、string 全体が 2 バイトでエンコードされていなければなりません。

これがデフォルト値で、大半の環境で使用できます。

- **no**

WebSEAL は、フォーム内の UTF-8 エンコードを認識しません。ローカル・コード・ページのみに使用されます。フォームのデータが妥当性検査にパスした場合は、内部使用のために UTF-8 に変換されます。

- **auto**

WebSEAL は、UTF-8 と他の形式の言語文字エンコードを区別しようとします。WebSEAL は、正しく構成された UTF-8 入力データを正しく処理します。エンコード方式が UTF-8 と認められない場合は、コーディングは非 UTF-8 として処理されます。

照会ストリングにおける UTF-8 サポート

WebSEAL が照会ストリングからのデータを処理する方法は、次に示すように、WebSEAL 構成ファイルで指定できます。

```
[server]
utf8-qstring-support-enabled = {yes|no|enabled}
```

デフォルトの設定は **no** です。したがって、WebSEAL のデフォルトの振る舞いでは、すべての照会ストリングはローカル・コード・ページであると想定します。

可能な 3 つの値は、次のとおりです。

- **はい**

WebSEAL は、照会ストリングの UTF-8 エンコードのみを認識し、データは変更せずに使用されます。これらの UTF-8 文字は、妥当性検査され、データを処理する際に考慮されます。その他のエンコード技法は受け入れられません。

ご使用の WebSEAL サーバーが UTF-8 を使用する照会ストリングを処理する必要があるときは、この設定を使用します。

単一バイト・ラテン語ロケール (フランス語、ドイツ語、スペイン語など) で操作され、UTF-8 を使用するアプリケーションからの照会を処理するサーバーは、この設定を使用する必要があります。マルチバイト・ロケールで実行され、UTF-8 照会ストリングのみを処理するサーバーは、この設定を使用できます。

- **no**

WebSEAL は、照会ストリング内の UTF-8 エンコードを認識しません。ローカル・コード・ページのみに使用されます。フォームのデータが妥当性検査にパスした場合は、内部使用のために UTF-8 に変換されます。

これがデフォルト値で、大半の環境で使用できます。

7 ビット ASCII 英語ロケールで操作されるサーバーは、この設定を使用できません。

- **auto**

WebSEAL は、UTF-8 と他の形式の言語文字エンコード (DBCS および Unicode) を区別しようとします。WebSEAL は、正しく構成された UTF-8 エンコードを正しく処理します。エンコード方式が UTF-8 と認められない場合は、コーディングは DBCS または Unicode として処理されます。

マルチバイト・ロケールで実行され、UTF-8 と非 UTF-8 が混用されている照会ストリングを処理するサーバーは、この設定を使用できます。

単一バイト・ラテン語ロケール (フランス語、ドイツ語、スペイン語など) で操作され、UTF-8 と非 UTF-8 が混用されている照会ストリングを処理するサーバーは、この設定を使用できます。

クロスドメイン・シングル・サインオン用のトークンの UTF-8 エンコード

クロスドメイン・シングル・サインオンに使用するトークン内のストリングに UTF-8 エンコードを使用するか否かは、WebSEAL 構成ファイルで指定します。

```
[cdsso]
use-utf8 = {true|false}
```

デフォルト値は true です。

use-utf8 が false に設定されていると、ストリングはローカル・コード・ページを使用してエンコードされます。バージョン 5.1 より前のバージョンの WebSEAL を使用してクロスドメイン・シングル・サインオンをインプリメントするときはこの値を使用します。5.1 より前のバージョンの WebSEAL では、トークンに UTF-8 エンコードを使用していません。これらの古いサーバーが含まれる環境をデプロイするときは、WebSEAL で UTF-8 エンコードを使用しないように構成してください。この設定は、後方互換性を保つために必要です。

注: この値が false に設定されているとき、UTF-8 から非 UTF-8 ローカル・コード・ページに変換する際にデータ損失が発生することに注意してください。

e-community シングル・サインオン用のトークンの UTF-8 エンコード

e-community シングル・サインオンに使用するトークン内のストリングに UTF-8 エンコードを使用するか否かは、WebSEAL 構成ファイルで指定します。

```
[e-community-sso]
use-utf8 = {yes|no}
```

デフォルト値は yes です。

use-utf8 が no に設定されていると、ストリングはローカル・コード・ページを使用してエンコードされます。バージョン 5.1 より前のバージョンの WebSEAL を使用して e-community シングル・サインオンをインプリメントするときはこの値を使用します。5.1 より前のバージョンの WebSEAL では、トークンに UTF-8 エンコードを使用していません。これらの古いサーバーが組み込まれている環境をデプロイするときは、UTF-8 エンコードを使用しないように WebSEAL サーバーを構成してください。この設定は、後方互換性を保つために必要です。

注: この値が no に設定されているとき、UTF-8 から非 UTF-8 ローカル・コード・ページに変換する際にデータ損失が発生します。

フェイルオーバー認証用の Cookie の UTF-8 エンコード

フェイルオーバー認証 Cookie 内のストリングに UTF-8 エンコードを使用するか否かは、WebSEAL 構成ファイルで指定します。

```
[failover]
use-utf8 = {yes|no}
```

デフォルト値は yes です。

use-utf8 が no に設定されていると、フェイルオーバー認証 Cookie はローカル・コード・ページを使用してエンコードされます。バージョン 5.1 より前のバージョンの WebSEAL を使用してフェイルオーバー認証をインプリメントするときはこの値を使用します。5.1 より前のバージョンの WebSEAL では、フェイルオーバー認証 Cookie に UTF-8 エンコードを使用していません。これらの古いサーバーが組み込まれている環境をデプロイするときは、UTF-8 エンコードを使用しないように WebSEAL サーバーを構成してください。この設定は、後方互換性を保つために必要です。

注: この値が no に設定されているとき、UTF-8 から非 UTF-8 ローカル・コード・ページに変換する際にデータ損失が発生することに注意してください。

LTPA 認証用の Cookie の UTF-8 エンコード

LTPA 認証に関して WebSEAL がサポートするのは LTPA バージョン 2 Cookie のみです。このバージョンの LTPA Cookie の仕様では、UTF-8 エンコード方式を使用することが求められます。

この要件のため、LTPA Cookie について UTF-8 エンコード方式を使用可能または使用不可にするオプションはありません。LTPA Cookie は常に UTF-8 でエンコードされます。

junction 要求における UTF-8 エンコード

WebSEAL は、バックエンド・サーバーへの要求の HTTP ヘッダーに情報を挿入します。この情報には、拡張属性またはユーザー・データを組み込むことができます。バージョン 5.1 より前の WebSEAL バージョンでは、ローカル・コード・ページを使用して、ヘッダーが要求に追加されていました。WebSEAL バージョン 5.1 以上では、ヘッダー・データは、構成可能な形式で伝送されます。

デフォルトでは、WebSEAL は UTF-8 コード・ページを使用して情報を HTTP ヘッダーに追加します。このアクションにより、非 UTF-8 コード・ページに変換する際に発生する可能性のあるデータ損失を防げるようになりました。また、このデータは、デフォルトで、URI でエンコードされて送られます。後方互換性を保つために、ヘッダー・データの形式は、ローカル・コード・ページに合わせて構成できます。さらに、ロー UTF-8 および URI エンコード・ローカル・コード・ページという 2 つの形式がサポートされます。

junction を作成するための **-e** オプションは、HTTP ヘッダーに入れられてバックエンド・サーバーに送られるユーザー名、グループ、およびその他の拡張属性のエンコード方式を指定します。エンコード・オプションは、以下の引数のいずれかを使用します。

引数	説明
utf8_uri	URI でエンコードされた UTF-8 データ。 すべての空白文字および非 ASCII バイトは %XY とエンコードされます。ここで、X および Y は 16 進値 (0 から F) です。 エンコード方式は、以下の文字に対しても適用されます。 • 0x1F より小さく 0x7F より大きいコードの ASCII 文字 • タブ、復帰、および改行のエスケープ文字 • パーセント記号
utf8_bin	エンコードされていない UTF-8 データ。 この設定により、データは、データ損失なしに伝送され、ユーザーはデータを URI でデコードする必要がありません。 この設定はよく注意して使用する必要があります。それは、この設定が HTTP 仕様の一部ではないからです。
lcp_uri	URI でエンコードされたローカル・コード・ページ・データ。 ローカル・コード・ページに変換できない UTF-8 文字はすべて疑問符 (?) に変換されます。このオプションはよく注意して、ローカル・コード・ページが望ましいストリングを生成する環境でのみ使用してください。
lcp_bin	エンコードされていないローカル・コード・ページ・データ。 このモードは、5.1 より前のバージョンの WebSEAL で使用されていたものです。このモードを使用すると前のバージョンからのマイグレーションを可能にするので、アップグレード環境で使用されます。 このモードを使用すると、データ損失が起こる可能性があることに注意してください。十分注意して使用してください。

要求データ内の文字エンコード方式の検証

WebSEAL は、要求を構文解析して、文字エンコード方式がバックエンド・サーバーの要件と互換性があることを確認します。例えば、要求の照会ストリングに WebSEAL では受け入れられない文字エンコード (ロー・バイナリー・データ) が入っていて、そのために WebSEAL がリジェクトすることが起こります。

無効な文字エンコードという問題は、バックエンド・サーバー・アプリケーションの特定の要件が原因で起こる場合もあります。典型的なシナリオでは、クライアントがこのバックエンド・アプリケーションに要求を出します。要求には、バックエンド・アプリケーションが必要とする照会ストリングが組み込まれますが、その照会ストリングには WebSEAL にとっては不明な文字エンコードが含まれています。WebSEAL は要求をリジェクトし、「無効な要求」(400) エラーを戻します。エラー・ログには、例えば、「URL に正しくない文字があります (Illegal character in URL)」というメッセージが入ります。

文字エンコード方式の誤った検証という問題に対するソリューションの 1 つには、要求の照会ストリングおよび POST 本体のデータを検証しないように WebSEAL を構成する方法があります。これによって、要求データを未変更のままバックエンド・アプリケーションに受け渡すことができます。

WebSEAL に照会ストリングおよび POST 本体データを検証しないように指示するには、以下のように、WebSEAL 構成ファイルの **[server]** スタンザ内の **decode-query** スタンザ・エントリーを「no」に設定します。

```
[server]
decode-query = no
```

デフォルト設定は、以下のとおりです。

```
decode-query = yes
```

decode-query が「yes」に設定されていると、WebSEAL は、**utf8-qstring-support-enabled** スタンザ・エントリーにしたがって、要求内の照会ストリングを検証します。73 ページの『照会ストリングにおける UTF-8 サポート』を参照してください。動的 URL が使用可能である場合に、この設定が要求内の POST 本体データに適用されます。動的 URL は、要求内の POST 本体データを照会ストリング・フォーマットに変換します。686 ページの『照会ストリング・フォーマットへの POST 本体動的データの変換』を参照してください。

decode-query が「yes」に設定されていると、WebSEAL は、**utf8-form-support-enabled** スタンザ・エントリーにしたがって、要求内の POST 本体を検証します。72 ページの『POST 本体情報内の UTF-8 サポート (フォーム)』を参照してください。

decode-query=no と設定する場合は、保護オブジェクトの保護により起こりうる結果を理解している必要があります。特に、WebSEAL が、要求内の照会ストリングを検証しないように構成されている場合 (**decode-query=no**) は、許可検査用の動的 URL マッピングが使用可能であれば、使用不可にする必要があります。

動的 URL 機能を使用不可にするには、以下のようにして、WebSEAL 構成ファイルの **[server]** スタンザ内の **dynurl-map** スタンザ・エントリーをコメント化します。

```
[server]
#dynurl-map = bin/dynurl.conf
```

サポートされているワイルドカード・パターン・マッチング文字

以下の表に、WebSEAL によってサポートされているワイルドカード・パターン・マッチング文字をリストします。

表 3. サポートされているワイルドカード・マッチング文字

文字	説明
¥	円記号の後に続く文字は、特殊シーケンスの一部です。例えば、¥t は TAB 文字です。その他のパターン・マッチング文字 (? * [] ^) をエスケープするために使用されます。円記号 (¥) 文字に一致させるには「¥¥」を使用します。

表 3. サポートされているワイルドカード・マッチング文字 (続き)

文字	説明
?	単一の文字に対応するワイルドカード。例えば、ストリング「abcde」と表現「ab?de」は一致します。
*	ゼロ個以上の文字に対応するワイルドカード。
[]	どれでも対応できる一組の文字を定義します。例えば、ストリング「abcde」と正規表現「ab[cty]de」は一致します。
^	否定を示します。例えば、表現「^ab」は、「a」または「b」以外のすべての文字に一致します。

この文書内のワイルドカードを使用したパターン・マッチング例には以下のものがあります。

- CDSO トークンに追加する拡張属性の指定 (627 ページの『トークンに追加する拡張属性』)。
- e-community トークンに追加する拡張属性の指定 (655 ページの『トークンに追加する拡張属性』)。
- 動的 URL への ACL および POP オブジェクトのマッピング (687 ページの『動的 URL への ACL および POP オブジェクトのマッピング』)。

システム環境変数の設定

system-environment-variables スタンザを使用して、初期化時に WebSEAL デーモンがエクスポートするシステム環境変数をリストします。エクスポートするシステム環境変数ごとに個別のエントリーを含めます。

注:

- この機能は、Windows プラットフォームではサポートされていません。
- 環境変数名は大文字小文字の区別があります。

各構成エントリーの形式は、以下のとおりです。

```
<env-name> = <env-value>
```

説明:

```
<env-name>
```

システム環境変数の名前。

```
<env-value>
```

システム環境変数の値。

例:

```
[system-environment-variables]
LANG = de
```

第 4 章 Web サーバー応答の構成

この章では、WebSEAL サーバーがクライアント要求への応答で使用可能なリソースについて説明します。

この章で扱うトピックは以下のとおりです。

- 『静的 HTML サーバー応答ページ』
- 84 ページの『HTML サーバー応答ページのロケーション』
- 86 ページの『HTML サーバー応答ページの変更』
- 94 ページの『アカウント管理ページの構成』
- 96 ページの『エラー・メッセージ・ページの構成』
- 98 ページの『サーバー応答でのマルチロケール・サポート』
- 100 ページの『Mozilla Firefox での favicon.ico ファイルの処理』
- 102 ページの『リダイレクト応答内のロケーション URL フォーマットの構成』
- 103 ページの『ローカル応答リダイレクト』

静的 HTML サーバー応答ページ

WebSEAL は、クライアント要求に対する応答の表示に使用できる静的 HTML サーバー応答ページを多数提供します。

これらのページには、以下のものがあります。

- エラー・メッセージ
- 情報メッセージ
- ログイン・フォーム
- パスワード管理フォーム

ユーザーは、これらのページのコンテンツを変更してサイト固有のメッセージ表示したり、サイト固有のアクションを実行したりできます。ほとんどのページは、HTTP または HTTPS を介したフォームおよび基本認証に適しています。

以下の表に HTML メッセージ・ページの名前、コンテンツ、説明をリストします。以下のコードを使用して、メッセージ・タイプを指定します。

- **ER** - エラー・メッセージ
- **IN** - 情報メッセージ
- **LG** - ログイン・フォーム
- **PW** - パスワード管理フォーム
- **NA** - 適用外

ファイル名	状況および HTTP コード	説明	タイプ
132120c8.html	認証が失敗しました (HTTP 403)	使用されるクライアント証明書で資格情報を検索できません。考えられる理由は、以下のとおりです。 • ユーザーが誤った証明書を提供した。 • ユーザーの資格情報が認証データベースに含まれていない。	ER
38ad52fa.html	ディレクトリーが空ではありません (HTTP 500)	要求した操作を実行するためには、空でないディレクトリーを除去する必要があります。要求された操作は、正しくない操作です。	ER
38b9a4b0.html	アプリケーション・サーバーがオフラインです (HTTP 503)	アクセス中のアプリケーション・サーバーは、システム・アドミニストレーターによってオフラインにされました。スロットルされた、またはオフラインの操作状態にある junction があるために要求がブロックされた場合に戻されます。	ER
38b9a4b1.html	Service Unavailable (HTTP 503)	必要なリソースを使用できないため、WebSEAL サーバーは要求を処理できません。	ER
38b9a41f.html	追加ログインが拒否されました (HTTP 200)	この Web サーバーには、別のクライアントから既にログインしています。初期セッションが終了するまで、新規ログインは許可されません。	ER
38cf013d.html	キャッシングの要求が失敗しました (HTTP 500)	request-max-cache または request-body-max-read の値を超過しました。	ER
38cf0259.html	ユーザーをサインオンできません (HTTP 500)	要求されたリソースでは、WebSEAL サーバーがユーザーを別の Web サーバーにサインオンさせる必要があります。しかし、WebSEAL が情報を検索しようとしていたときに、問題が発生しました。	ER
38cf025a.html	ユーザーにはシングル・サインオン情報がありません (HTTP 500)	WebSEAL では、要求されたリソースの GSO ユーザーを見つけられませんでした。	ER
38cf025b.html	ユーザー用のシングル・サインオン・ターゲットがありません (HTTP 500)	WebSEAL では、要求されたリソースの GSO ターゲットを見つけられませんでした。	ER
38cf025c.html	ユーザーに複数のサインオン・ターゲットがあります (HTTP 500)	要求されたリソースに関して、複数の GSO ターゲットが定義されています。これは誤った構成です。	ER

ファイル名	状況および HTTP コード	説明	タイプ
38cf025d.html	ログインが必要です (HTTP 500)	要求されたリソースが junction 先バックエンド Web サーバーによって保護されているので、WebSEAL がユーザーをその Web サーバーにサインオンさせる必要があります。そのためには、ユーザーがまず WebSEAL にログインする必要があります。	ER
38cf025e.html	ユーザーをサインオンできません (HTTP 500)	要求されたリソースでは、WebSEAL がユーザーを別の Web サーバーにサインオンさせることを必要としています。しかし、そのユーザー・アカウントに関するサインオン情報に誤りがあります。	ER
38cf025f.html	予期しない認証によるユーザー確認 (HTTP 500)	WebSEAL が、予期しない認証によるユーザー確認を junction 先バックエンド Web サーバーから受信しました。	ER
38cf0421.html	Moved Temporarily (HTTP 302)	要求したリソースが一時的に移動されました。通常、リダイレクトの取り扱いを誤った場合にこのイベントが発生します。	ER
38cf0424.html	Bad Request (HTTP 400)	WebSEAL が無効な HTTP 要求を受信しました。	ER
38cf0425.html	ログインが必要です (HTTP 401)	要求されたリソースは WebSEAL によって保護されているので、アクセスするためには、まずログインする必要があります。	ER
38cf0427.html	禁止 (HTTP 403)	要求されたリソースにアクセスする許可がユーザーに与えられていません。	ER
38cf0428.html	Not Found (HTTP 404)	要求されたリソースが見つかりません。	ER
38cf0432.html	Service Unavailable (HTTP 503)	WebSEAL が要求の処理を完了するために必要とするサービスが、現在は使用不能です。	ER
38cf0434.html	プライバシーが必要 (HTTP 403)	プライバシー・レベルでの保護品質が必要です。	ER
38cf0437.html	サーバーが中断状態です (HTTP 500)	WebSEAL サーバーが、システム・アドミニストレーターによって一時的に中断状態にされています。サーバーがアドミニストレーターによってサービスに戻されるまで、要求は処理されません。	ER

ファイル名	状況および HTTP コード	説明	タイプ
38cf0439.html	セッション情報が失われています (HTTP 500)	ブラウザおよびサーバー対話が、応答しなくなっている junction 先バックエンド・サーバーとのステートフル・セッションでした。WebSEAL では、このサーバー上にあるサービスが要求の処理を完了することが必要です。	ER
38cf0442.html	Service Unavailable (HTTP 503)	WebSEAL が必要とするサービスは junction 先バックエンド・サーバー上にありますが、ここでの SSL 相互認証に障害が起きています。	ER
38cf04c6.html	サード・パーティー・サーバーが応答していません (HTTP 500)	要求したリソースはサード・パーティー・サーバーにあります。WebSEAL がそのサーバーに接触しようとしたますが、応答がありません。	ER
38cf04d7.html	サード・パーティー・サーバーが応答していません (HTTP 500)	要求したリソースはサード・パーティー・サーバーにあります。WebSEAL がそのサーバーに接触しようとしたますが、応答がありません。	ER
38cf07aa.html	CGI プログラムが失敗しました (HTTP 500)	CGI プログラムが正しく実行されませんでした。	ER
38cf08cc.html	アクセスが拒否されました (HTTP 403)	要求したリソースは、アクセスを特定の期間に制限するポリシーによって保護されています。現在時刻は、許可時間枠から外れています。	ER
acct_locked.html	アカウントがロックされました (HTTP 200)	以下の場合に表示されるページ。 - ログイン失敗数が多すぎてユーザーのアカウントが一時的にロックされたために認証に失敗した。 - ユーザーがログインしようとした時点で、そのユーザーの nsAccountLock が true である (Sun Directory Server 内)。このページは、ログイン中にユーザーが正しいパスワードを入力した場合のみ表示されます。 注: Security Access Manager で基本認証 (BA-auth) を使用している場合、acct_locked.html ファイルをカスタマイズして追加イメージを含めることはできません。このファイルにイメージを組み込むことはできますが、今後この組み込みイメージにアクセスするように要求しても、失敗します。	ER
certfailure.html	証明書認証が失敗しました (HTTP 200)	クライアント証明書による認証が失敗しました。 accept-client-certs = required の場合に、クライアントが証明書の認証に失敗した場合に表示されるページ。この接続を行うには、有効なクライアント証明書が必要です。	ER

ファイル名	状況および HTTP コード	説明	タイプ
certlogin.html	Access Manager ログイン (HTTP 200)	accept-client-certs = prompt_as_needed のときに使用される証明書ログイン・フォーム。	LG
certstepuphttp.html	証明書認証へのステップアップが失敗しました (HTTP 200)	HTTP を使用した証明書へのステップアップが失敗しました。HTTPS を使用する必要があります。HTTPS を使用して再度ページにアクセスしてください。	ER
default.html	Server Error (HTTP 500)	予期しないエラーによって、WebSEAL が要求を完了できませんでした。	ER
deletesuccess.html	成功 (HTTP 200)	クライアントが開始した DELETE 要求が正常に完了しました。	IN
help.html	PKMS 管理 (HTTP 200)	pkmslogout および pkmspasswd に関するヘルプ情報。	IN
login.html	Access Manager ログイン (HTTP 200)	ユーザー名およびパスワードを求める標準要求フォーム。	LG
login_success.html	成功 (HTTP 200)	通常 WebSEAL は、要求されたリソースの URL をキャッシュに入れ、ログインが成功するとユーザーにそのリソースを戻します。何らかの理由で WebSEAL が最初に要求されたリソースの URL を判別できない場合、ログインの成功後、login_success.html ページが表示されます。	IN
logout.html	PKMS 管理: ユーザー・ログアウト (HTTP 200)	ログアウトの成功後に表示されるページ。ユーザー USERNAME はログアウトしました。	IN
passwd.html	PKMS 管理: パスワード変更 (HTTP 200)	パスワード変更フォーム。パスワード変更要求が失敗した場合も表示されます。	PW
passwd_exp.html	PKMS 管理: 期限切れパスワード (HTTP 200)	パスワードが期限切れのためユーザー認証が失敗した場合に表示されるページ。期限切れパスワードを変更します。	PW
passwd_rep.html	PKMS 管理: パスワード変更 (HTTP 200)	パスワード変更要求が成功した場合に表示されるページ。	PW
passwd_warn.html	パスワードの管理 パスワードの有効期限切れが近づいています (HTTP 200)	ユーザー LDAP パスワードの有効期限切れが近づいている場合に表示されるページ。	PW

ファイル名	状況および HTTP コード	説明	タイプ
putsuccess.html	成功 (HTTP 200)	クライアントが開始した PUT 操作が正常に完了しました。	IN
query_contents.html	Win32 Web サーバーの junction (HTTP 200)	Access Manager WebSEAL サーバーから Win32 プラットフォームで稼働するサード・パーティー Web サーバーへの junction の作成に関するインストールおよび構成情報。	IN
relocated.html	一時的に移動されました (HTTP 302)	要求したリソースが一時的に移動されました。	IN
stepuplogin.html	Access Manager ステップアップ・ログイン (HTTP 200)	ステップアップ認証用ログイン・フォーム。	LG
switchuser.html	Access Manager ユーザー切り替え (HTTP 200)	ユーザー切り替え用ログイン・フォーム。	LG
template.html	テンプレート	カスタム・エラー・メッセージ用テンプレート・フォーム。	NA
too_many_sessions.html	PKMS 管理: セッション強制終了 (HTTP xxx)	単一ユーザーによる並行ログインの制限を超えた場合のエラー・メッセージ。	ER
websealerror.html	WebSEAL サーバー・エラー (HTTP 400)	WebSEAL サーバーの内部エラーです。	ER

HTML サーバー応答ページのロケーション

WebSEAL サーバー上の保管場所として、静的 HTML サーバー応答ページは以下の 3 つの一般的なカテゴリーにグループ化されます。

- アカウント管理ページ
- エラー・メッセージ・ページ
- junction 固有の静的サーバー応答ページ

以下のセクションで説明するとおり、これらの 3 つの保管カテゴリーのロケーションは構成することができます。

- 85 ページの『管理ルート』
- 85 ページの『アカウント管理ページのロケーション』
- 85 ページの『エラー・メッセージ・ページのロケーション』
- 86 ページの『junction 固有の静的サーバー応答ページ』

管理ルート

IBM Security Web Gateway Appliance では、LMI を使用して管理ルート・ファイル構造にアクセスすることができます。

ファイル構造は、各インスタンスの「リバース・プロキシ管理ルートの管理」ページで使用できます。この管理ルート・ページを開いて、LMI Web インターフェースからファイル構造にアクセスするには、以下のようにする必要があります。

1. 上部のメニューから、「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「管理」>「リバース・プロキシ」を選択します。「リバース・プロキシ」管理ページが表示されます。
2. 選択可能なインスタンス名のリストから、管理対象のインスタンスをクリックします。
3. 「管理」>「管理ルート」を選択します。選択したインスタンスの「リバース・プロキシ管理ルートの管理」ページが表示され、使用可能なファイル構造が示されます。

使用可能なディレクトリーは以下のとおりです。

- 管理
- errors
- oauth
- junction-root

アカウント管理ページのロケーション

アカウント管理ページには、ログイン・フォーム、パスワード管理フォーム、およびいくつかの情報メッセージが含まれています。

LMI を使用して、アカウント管理ページにアクセスできます。「リバース・プロキシ管理ルートの管理」ページの `management` ディレクトリーを開きます。ページは、このロケーションの言語固有のサブディレクトリーにあります。ユーザーのロケールに固有のサブディレクトリーは、WebSEAL のインストールおよび構成中に、ディレクトリー階層の最下層に自動的に付加されます。

デフォルトの米国英語ディレクトリーは、次のとおりです。

`management/C`

日本語ロケール・ディレクトリーは次のようになります。

`management/JP`

マルチロケール・サポートについての詳細は、98 ページの『サーバー応答でのマルチロケール・サポート』を参照してください。

エラー・メッセージ・ページのロケーション

ページは、`errors` ディレクトリーの言語固有サブディレクトリーにあります。これらのディレクトリーには、「リバース・プロキシ管理ルートの管理」ページから

アクセスできます。ユーザーのロケールに固有のサブディレクトリーは、WebSEAL のインストールおよび構成中に、ディレクトリー階層の最下層に自動的に付加されます。

デフォルトの米国英語ディレクトリーは、次のとおりです。

errors/C

日本語ロケール・ディレクトリーは次のようになります。

errors/JP

マルチロケール・サポートについての詳細は、98 ページの『サーバー応答でのマルチロケール・サポート』を参照してください。

junction 固有の静的サーバー応答ページの作成について詳しくは、『junction 固有の静的サーバー応答ページ』を参照してください。

junction 固有の静的サーバー応答ページ

静的サーバー応答ページは、junction 単位でカスタマイズできます。

カスタマイズした静的サーバー応答ページのファイルを junction 固有のディレクトリーに追加します。

errors/language/junction_id

ここで、*junction_id* は標準 junction の junction ポイント (先行する / 文字は除く)、または仮想ホスト junction の仮想ホスト・ラベルを表します。例えば、次のとおりです。

errors/C/test_junction

WebSEAL は以下の順序で静的サーバー応答ページ・ファイルを検索し、最初に見つかったファイルをクライアントに戻します。

1. errors/language/junction_id/page.html
2. errors/language/junction_id/default.html
3. errors/language/page.html
4. errors/language/default.html

junction 名では / 文字を使用できます。例えば、jct ディレクトリー内に junction ディレクトリー test を作成した場合、作成された junction は /jct/test として指定されます。この場合、WebSEAL はファイルを errors/language/jct/test ディレクトリー内で検索します。

HTML サーバー応答ページの変更

このセクションは、以下のトピックで構成されています。

- 87 ページの『HTML 応答ページのカスタマイズのためのガイドライン』
- 87 ページの『HTML 応答ページのカスタマイズのためのマクロ・リソース』
- 93 ページの『カスタム・ログイン・フォームへのイメージの追加』

HTML 応答ページのカスタマイズのためのガイドライン

静的 HTML サーバー応答ページをカスタマイズして、現在の WebSEAL インプリメンテーションをより適切に反映させることができます。以下の注意事項を守ってください。

- ファイルの名前は変更してはなりません。16 進数は、WebSEAL が正しいエラー・ファイルを表示するために使用しています。
- ページの内容を変更するときは、HTML エディターまたはテキスト・エディターを使用してください。有効な HTML タグ付けを使用してください。
- イメージや CSS などのリソースの URI には、(相対 URI でなく) サーバー相対 URI を指定します。WebSEAL に仮想ホスト junction が使用されている場合は、これらのリソースには絶対 URI を使用する必要があります。
- WebSEAL には、動的情報を取り込むために使用できる一連のマクロが用意されています。『HTML 応答ページのカスタマイズのためのマクロ・リソース』を参照してください。

HTML 応答ページのカスタマイズのためのマクロ・リソース

マクロは事前定義された特殊な形式のストリングで、静的 HTML サーバー応答ページに情報を直接追加するために使用されます。

WebSEAL は、静的 HTML ページで応答する場合、ページを解析してマクロがあるかどうか検索します。マクロが検出された場合、適切な内容に直接置換されます。値がそのページに関連がある場合のみ、マクロが取り込まれます。

例えば、WebSEAL は、結果がエラーになった要求の応答として静的サーバー応答ページを戻します。WebSEAL は、静的サーバー応答ページで **ERROR** マクロを検出した場合、要求の処理時に生成されたエラー・コードのストリング表記に置換します。

以下のマクロは、WebSEAL が提供する静的 HTML サーバー応答ページの一部で指定されるもので、ページのカスタマイズに使用できます。

マクロ	説明
AUTHNLEVEL	認証強度ポリシー (ステップアップ) で使用する認証レベルに置換します。
BACK_NAME	要求内にリファラー・ヘッダーがある場合は値「BACK」、リファラー・ヘッダーがない場合は「NONE」に置換します。
BACK_URL	要求のリファラー・ヘッダーの値、またはリファラー・ヘッダーがない場合「/」に置換します。
BASICAUTHN	certlogin.htmlおよび stepuplogin.html ログイン・フォームにおける情報表示のコントロールに使用します。認証方式 (マクロ名で示される) が有効な場合、マクロが管理するフォーム内のセクションが表示されます。認証方式が有効でない場合、マクロはコメント開始区切り文字 (<!--) に置き換えられます。フォーム内のすべての後続の情報は、コメント終了区切り文字 (-->) に達するまで、コメントとして扱われます。

マクロ	説明
CERTAUTHN	certlogin.htmlおよび stepuplogin.html ログイン・フォームにおける情報表示のコントロールに使用します。認証方式 (マクロ名で示される) が有効な場合、マクロが管理するフォーム内のセクションが表示されます。認証方式が有効でない場合、マクロはコメント開始区切り文字 (<!--) に置き換えられます。フォーム内のすべての後続の情報は、コメント終了区切り文字 (-->) に達するまで、コメントとして扱われます。
CREDATTR{name}	指定された <i>name</i> という名前のユーザー資格情報属性の値。例えば、 CREDATTR{tagvalue_session_index} はセッション・トークンを返します。
EAIAUTHN	certlogin.htmlおよび stepuplogin.html ログイン・フォームにおける情報表示のコントロールに使用します。認証方式 (マクロ名で示される) が有効な場合、マクロが管理するフォーム内のセクションが表示されます。認証方式が有効でない場合、マクロはコメント開始区切り文字 (<!--) に置き換えられます。フォーム内のすべての後続の情報は、コメント終了区切り文字 (-->) に達するまで、コメントとして扱われます。
ERROR	Security Access Manager から戻されたハードコーディング・エラー・メッセージ ERROR_TEXT と同じ。これらの両マクロは、WebSEAL の前のバージョンとの互換性のために存在します。
ERROR_CODE	エラー・コードの数値。
ERROR_TEXT	メッセージ・カタログ内のエラー・コードに対応するテキスト。 ERROR と同じ。これらの両マクロは、WebSEAL の前のバージョンとの互換性のために存在します。
EXPIRE_SECS	パスワード期限切れになるまでの秒数が格納されます。 この値をパスワード警告フォーム (passwd_warn.html) に送信すると、ユーザーがパスワードを変更しなければならない残り時間を表示できます。
FAILREASON	エラー・メッセージ。
HOSTNAME	完全修飾ホスト名。
HTTP_BASE	サーバー「http://host:tcpport/」のベース HTTP URL。
HTTPS_BASE	サーバー「https://host:sslport/」のベース HTTPS URL。
HTTPHDR{name}	特定の HTTP ヘッダーの内容を含めるために使用されます。指定された HTTP ヘッダーが要求内にない場合は、マクロに「Unknown」というテキストが含まれます。 例えば、'Host' HTTP ヘッダーを含めるためのマクロ名は HTTPHDR{Host} になります。

マクロ	説明
LOCATION	クライアントがリダイレクトされる先の URL が入ります。リダイレクトでのみ送信されます。
METHOD	クライアントによって要求される HTTP 方式。
OLDSESSION	WebSEAL が、WebSEAL セッション・キャッシュ内のいずれの既存エントリーにも一致しない、古い (「失効」) セッション Cookie を含むユーザー要求を受信した場合、マクロの値は「1」に設定されます (通常は「0」に設定)。マクロは、WebSEAL が認識されていないセッション Cookie を検出すると必ず設定されます。例えば、セッション・タイムアウト、セッション強制終了、およびユーザーが WebSEAL サーバーを切り替えたときなどに、認識されていないセッション Cookie が検出されます。 ユーザーへのカスタマイズされた応答に対するトリガー・メカニズムを提供するために、標準 WebSEAL ログイン・フォームで使用されます。このカスタム応答では、セッションが有効でなくなった理由をより正確にユーザーに説明することができます。 333 ページの『古いセッション Cookie に対するカスタマイズ応答』を参照してください。
PROTOCOL	使用されるクライアント接続プロトコル。HTTP または HTTPS のどちらでもかまいません。
REFERER	要求からの HTTP リファラー・ヘッダーの値、または「Unknown」 (値がない場合)。
REFERER_ENCODED	HTTP リファラー・ヘッダーおよびマクロの URI エンコード・バージョン。
STEPUP	必要なステップアップ・レベルを指定するメッセージ。ステップアップ・ログイン・フォームを戻すときにのみ送信されます。
URL	クライアントによって要求される URL。
URL_ENCODED	URI およびマクロの URI エンコード・バージョン。
USERNAME	要求を行うユーザーの名前。 (200 ページの『再認証用のログイン・フォームのカスタマイズ』も参照してください。)

マクロ・データ・ストリング形式

WebSEAL は、HTML サーバー応答ページに挿入するマクロ・データ・ストリングの形式を指定する構成スタンザ・エントリーを提供します。デフォルト設定では UTF-8 形式が指定されています。

```
[content]
utf8-template-macros-enabled = yes
```

静的 HTML サーバー応答ページは、デフォルトで UTF-8 文字セットを使用します。ローカル・コード・ページを指定するように文字セットを変更する場合は、このエントリーを「no」に設定します。

この設定は、WebSEAL 構成ファイルの **[acct-mgt]** スタンザにある **error-dir** および **mgt-pages-root** スタンザ・エントリで指定されたディレクトリーにあるファイルに影響を与えます。84 ページの『HTML サーバー応答ページのロケーション』を参照してください。

テンプレートの組み込みマクロ

Security Access Manager はテンプレートにマクロを組み込む前にマクロの値を検証しますが、カスタマー環境によっては、テンプレートのカスタマイズ方法に応じて追加の変更が必要になる場合があります。Security Access Manager テンプレートで使われるマクロは、URL マクロまたは非 URL マクロです。URL マクロは、HTTP 要求 URL やリファラー URL などの URL を表す場合に使用されます。非 URL マクロは、ユーザー名やエラー・メッセージなど、その他の値を表す場合に使用されます。

以下のマクロが URL マクロです。

- %LOCATION%
- %URL%
- %REFERER%
- %BACK_URL%
- %HOSTNAME%
- %HTTP_BASE%
- %HTTPS_BASE%
- %REFERER_ENCODED%
- %URL_ENCODED%

Security Access Manager で使用されるその他のすべてのマクロが、非 URL マクロです。

Security Access Manager によるマクロのエンコード方法

Security Access Manager は、マクロをエンコードしてから、そのマクロを HTML テンプレートに組み込みます。

エンコードすると、マクロの値が JavaScript または HTML メタ文字でなく、テキストとして解釈されるようになります。URL マクロは Uniform Resource Identifier (URI) 文字としてエンコードされます。例えば、URL 内に左ブラケット (<) 文字がある場合、この文字はエンコード中に **%3c** に変換されます。非 URL マクロは、HTML エンティティーを使用してエンコードされます。例えば、左ブラケット (<) 文字および右ブラケット (>) 文字は **<** および **>** としてそれぞれエンコードされます。非 URL マクロ内にその他の HTML メタ文字がある場合、これらの文字は数字参照を使用してエンコードされます。例えば、二重引用符 (") は **"** として表示されます。

Security Access Manager は URL マクロと 非 URL マクロの両方で以下の文字をエンコードします。

表 4. URL マクロおよび 非 URL マクロでエンコードされる文字

名	文字または説明
より小記号	<
より大記号	>
コロン	:
アポストロフィ	'
引用符	"
円記号 (¥)	¥
ASCII 0x20 より小さいすべての値	例には、エスケープ、タブ、復帰、改行、改ページ、バックスペース、ヌル・バイトが含まれます。

さらに、Security Access Manager は 非 URL マクロ内のアンパーサンド (&) 文字をエンコードします。

Access Manger は、HOSTNAME を除くすべての URL マクロが絶対 URL であるか、またはサーバー相対 URL であるかを検証し、HTTP プロトコルまたは HTTPS プロトコルを使用します。 HOSTNAME マクロには英数字、ASCII 文字、ドット、およびハイフンのみを含める必要があります。

テンプレートでのマクロの使用

Security Access Manager 環境におけるクロスサイト・スクリプトの脆弱性による問題を回避するために、HTML テンプレートにマクロを組み込む場合は注意が必要です。マクロを組み込む場合は、以下のガイドラインを使用してください。

- URL マクロは HTML テキストとして安全に使用することができます。マクロを HTML テキストとして使用するには、マクロを HTML タグ内に組み込みます。以下に例を示します。

```
<b>%URL%</b>
```

- URL マクロは HTML 属性の HTML 属性値として安全に使用することができますが、該当するのは URL で使用することが想定されている属性値のみです。マクロを HTML 属性値として使用する場合は、マクロを二重引用符または単一引用符で囲む必要があります。以下に例を示します。

```
<a href="%URL%">clickable link</a>
```

- URL マクロは JavaScript ストリング値として安全に使用することができますが、二重引用符または単一引用符で囲む必要があります。以下に例を示します。

```
var url = '%URL%';
```

- 非 URL マクロは、HTML テキストとして安全に使用することができます。マクロを HTML テキストとして使用するには、マクロを HTML タグ内に組み込みます。以下に例を示します。

```
<b>%USERNAME%</b>
```

- 非 URL マクロは HTML 属性値として安全に使用することができますが、該当するのは URL で使用することが想定されていない属性値のみです。マクロを HTML 属性値として使用する場合は、マクロを二重引用符または単一引用符で囲む必要があります。以下に例を示します。

```
<input type="text" name="user" value="%USERNAME%">
```


- 非 URL マクロは JavaScript ストリング値として安全に使用することができますが、二重引用符または単一引用符で囲む必要があります。以下に例を示します。

```
var user = '%USERNAME%';
```

HTML タグおよび属性

URL マクロおよび 非 URL マクロは、ほとんどの基本 HTML タグで安全に使用することができます。ただし、通常の HTML タグと異なる特殊な構文を使用する HTML タグおよび属性があります。例えば、<style> タグの内容は、通常の HTML でなく、カスケード・スタイル・シートとして解釈されます。Security Access Manager マクロがエンコードされるのは、通常の HTML タグおよび属性と共に使用する場合です。内容が HTML と解釈されないタグの中では、使用しないでください。

一般に、内容が標準 HTML 構文に従わない HTML タグに Security Access Manager マクロを含めないでください。特定の HTML タグで使われる構文が不明な場合は、W3C HTML 仕様書およびご使用の Web ブラウザーの資料を参照してください。

また、%USERNAME% などの 非 URL マクロは、URL に使用される属性の HTML 属性値として使用しないでください。非 URL マクロを URL として使用すると、デプロイされた Security Access Manager 環境内でクロスサイト・スクリプトの脆弱性による問題が引き起こされることがあります。

マクロを操作するための JavaScript の使用

JavaScript を使用して Security Access Manager マクロを操作する方法は、2 つあります。1 つは、マクロを JavaScript ストリングとして使用する方法です。もう 1 つは、JavaScript を使用して HTML Document Object Model (DOM) を操作する方法です。マクロを JavaScript ストリングとして使用するには、二重引用符または単一引用符の間にマクロ名を挿入するだけです。例:

```
var username = "%USERNAME%";
```

マクロを JavaScript ストリングとして使用する場合は、マクロ値に URI エンコードまたは HTML エンティティー・エンコードが含まれている可能性があることに注意してください。JavaScript unescape() 関数を使用すると、マクロ値から URI エンコードを削除できます。

HTML エンティティー・エンコードを削除するためのお勧めの方法は、JavaScript を使用して HTML DOM を操作することです。例えば、以下の HTML コードを使用すると、%USERNAME% マクロからエンティティー・エンコードを削除できます。

```
<span id='user' style='visibility: hidden'%>USERNAME%</span>
<script>
  var user = document.getElementById('user');
  if (user && user.firstChild)
  {
    var name = user.firstChild.nodeValue;
  }
</script>
```

name 変数には %USERNAME% マクロの内容が含まれています。必要に応じて、この変数を使用できます。ただし、マクロ値を使用する場合は、HTML テンプレート・ページで DOM ベース・クロスサイト・スクリプトの脆弱性による問題が引き起こされないよう注意してください。

カスタム・ログイン・フォームへのイメージの追加

このタスクについて

ログイン・フォームなどのサーバー応答ページをカスタマイズする場合は、ページまたはフォームにイメージ (グラフィックス) を追加できます。以下のステップを実行してください。

手順

1. イメージ・ファイルを格納します。格納先は、junction-root の下の該当するサブディレクトリーです。LMI を使用してこのディレクトリーを管理できます。「リバース・プロキシー管理ルートの管理」ページにアクセスしてください。イメージ・ファイルの推奨ロケーションは、以下のとおりです。

```
junction-root/icons
```

以下の例に示すような HTML コードを使用してカスタム・ログイン・フォームにイメージを記述できます。

```
<image src="/icons/logo.jpg" alt="Company Logo">
```

2. イメージ・ファイル・フォーマットの定義が、WebSEAL 構成ファイルの **[content-mime-types]** に記述されていることを確認します。以下に例を示します。

```
[content-mime-types]
jpg = image/jpeg
```

3. logo.jpg に対する非認証アクセスを許可する ACL を作成します。これがログイン・ページであるため、アクセス時にユーザー ID は確立されません。したがって、イメージを含むイメージ・ファイル・オブジェクトまたはディレクトリー・オブジェクト (icons など) への非認証アクセスを許可する必要があります。最低限必要な許可は、Unauthenticated および Any-other に対する「Tr」です。以下に例を示します。

```
pdadmin> acl show icons-acl
ACL name: icons-acl
Description:
Entries:
    Any-other Tr
    Unauthenticated Tr
    User sec_master TcmdbsvaBRr1
```

注: Unauthenticated ACL エントリーに許可を設定する場合は、Any-other ACL エントリーと同じ許可が最低限必要です。

4. この例では、この ACL を icons ディレクトリーに明示的に付加します (または非認証許可がこのポイントに確実に継承されるようにします)。以下に例を示します。

```
pdadmin> acl attach /WebSEAL/abc.ibm.com-default/icons icons-acl
```

アカウント管理ページの構成

このセクションは、以下のトピックで構成されています。

- 『構成ファイル・スタンザ・エントリーおよび値』
- 『アカウント期限切れエラー・メッセージの構成』
- 95 ページの『パスワード・ポリシー・オプションの構成』

構成ファイル・スタンザ・エントリーおよび値

以下に示す HTML 応答 ページ・スタンザ・エントリーと値は、WebSEAL 構成ファイルの **[acnt-mgt]** スタンザに指定されています。一部のページは、識別情報を提供するフォーム・ログイン方式のみで使用されます。

スタンザ・エントリー	HTML 応答ページ	使用法
login =	login.html	フォーム・ログイン
login-success =	login_success.html	フォーム・ログイン
logout =	logout.html	フォーム・ログイン
account-inactivated =	acct_locked.html	任意のメソッド
account-locked =	acct_locked.html	任意のメソッド
passwd-expired =	passwd_exp.html	任意のメソッド
passwd-change =	passwd.html	任意のメソッド
passwd-change-success =	passwd_rep.html	任意のメソッド
passwd-change-failure =	passwd.html	任意のメソッド
passwd-warn =	passwd_warn.html	任意のメソッド
passwd-warn-failure =	passwd_warn.html	任意のメソッド
help =	help.html	任意のメソッド
certificate-login =	certlogin.html	証明書ログイン
cert-stepup-http =	certstepuphttp.html	証明書ログイン
stepup-login =	stepuplogin.html	ステップアップ認証
switch-user =	switchuser.html	任意のメソッド
cert-failure =	certfailure.html	証明書ログイン
eai-auth-error =	eaiautherror.html	外部認証インターフェース・ログイン・エラー・ページ
too-many-sessions =	too_many_sessions.html	並行セッションのエラー・ページが多すぎる場合に使用
html-redirect =	redirect.html	HTML リダイレクト

アカウント期限切れエラー・メッセージの構成

ログイン試行が失敗すると、WebSEAL はユーザーにエラー・メッセージを返します。メッセージは、ユーザーに戻される適切なアカウント管理ページに含まれている **ERROR** マクロによって伝達されます。汎用エラー・メッセージ（「ログインに失敗しました」）は、ユーザーが無効な認証情報（無効なユーザー名またはパスワードなど）を指定した場合のさまざまな状態に適用されます。

WebSEAL 構成ファイルの **[acct-mgt]** スタンザにある **account-expiry-notification** スタンザ・エントリーを使用して、アカウントの期限切れでログインに失敗したときのエラー・メッセージで追加情報を表示するかどうかを制御できます。

デフォルトの「no」という設定では、ユーザー・アカウントの期限切れでログインに失敗した場合に汎用エラー・メッセージ（「ログインに失敗しました」）しか戻されません。

```
[acct-mgt]
account-expiry-notification = no
```

account-expiry-notification スタンザ・エントリーを「yes」に設定すると、ユーザー・アカウントの期限切れでログインに失敗した場合により詳細なエラー・メッセージを戻すことができます。このより詳細なエラー・メッセージ（「アカウントの有効期限が切れています (Account expired)」）によって、ログイン失敗の正確な理由（アカウントの期限切れ）が示されます。

```
[acct-mgt]
account-expiry-notification = yes
```

「アカウントの有効期限が切れています (Account expired)」というメッセージは、使用されているユーザー名が正しいことを暗黙で示しています。このレベルの情報は、環境によっては機密漏れと見なされる場合もあります。

パスワード・ポリシー・オプションの構成

以下の WebSEAL オプションを **[acct-mgt]** スタンザ内で使用すると、LDAP ユーザーのパスワード・ポリシーおよびアカウント状態を使用することができます。

```
[acct-mgt]
enable-passwd-warn = yes
passwd-warn = passwd_warn.html
passwd-warn-failure = passwd_warn.html
account-inactivated = acct_locked.html
```

これらのオプションは、対応する Security Access Manager LDAP オプションも使用可能になっていて ([ldap] enhanced-pwd-policy=yes)、特定の LDAP レジストリー・タイプでサポートされていない限り、無効です。

enable-passwd-warn スタンザ・エントリーを使用すると、LDAP パスワード・ポリシーがユーザーのパスワードが有効期限切れに近づいていることを示している場合、WebSEAL はユーザーの資格情報に追加された **REGISTRY_PASSWORD_EXPIRE_TIME** 属性を検出できます。この新しい属性値は、ユーザーのパスワードの有効期限が切れるまでの秒数です。この属性が検出された場合は、ユーザーが WebSEAL にログインすると、パスワード警告フォームが表示されます。

ページ・マクロ **EXPIRE_SECS** を使用して、パスワードの有効期限が切れるまでの秒数を格納することができます。パスワード警告フォーム内でこのマクロを使用すると、ユーザーが自身のパスワードを変更するまでの残り時間を表示できます。

account-inactivated スタンザ・エントリーは、Sun Directory のユーザーがログインしようとしたときに、このユーザーの **nsAccountLock** 値が **true** の場合に表示され

るページを指定します。このページが表示されるのは、ログイン中にユーザーが正しいパスワードを入力した場合のみです。

passwd-warn スタンザ・エントリーは、LDAP パスワードが有効期限切れに近づいていることが WebSEAL によって検出された場合に、ログイン後に表示されるページを指定します。

passwd-warn-failure スタンザ・エントリーは、ユーザーが有効期限切れの近いパスワードを変更できなかった場合に表示されるページを指定します。このページは通常、**passwd-warn** スタンザ・エントリーで指定されるページと同じであり、ユーザーにもう一度パスワードを変更する機会を提供するものです。

passwd-warn エントリーおよび **passwd-warn-failure** エントリーで指定されるページを /pkmspasswd.form に送信する場合は、このページ上に warn という名前の (隠し) フィールドが必要です。warn フィールドの値は無視されるため、フィールド値は短くしてください。/pkmspasswd.form 管理 URL はこの隠しフィールドを検出し、引き続いてパスワード変更ページの警告バージョンを使用します。warn フィールドが検出されなかった場合は、警告以外のフォームが代わりに使用されます。

```
<input type="HIDDEN" name="warn" value="*">
```

/pkmskip WebSEAL 管理 URL を使用すると、**passwd-warn** ページでパスワードの変更をスキップし、ログインを継続することができます。この URL は、ログイン・プロセスによって中断される前に、ユーザーが本来アクセスしようとしていたページにユーザーを効率的にリダイレクトします。

passwd_warn、**passwd_warn_failure**、および **acct_inactivated** のローカル応答リダイレクト・オプションを使用できます。詳しくは、106 ページの『ローカル応答リダイレクトの操作』を参照してください。

エラー・メッセージ・ページの構成

WebSEAL がクライアントからの要求を処理できないとき、WebSEAL はそのクライアントに HTML エラー・メッセージ・ページを戻します。エラー・メッセージ・ページには、要求が失敗した理由が説明されています。

エラー・メッセージ・ページは、WebSEAL インスタンスが構成されるときにインストールされます。それぞれのエラー・メッセージ・ページは、別個の静的 HTML ファイルになっています。これらのファイルの名前は、戻されるエラー・コードの 16 進値です。これらのファイル名は変更してはなりません。

注: イメージや CSS などのリソースの URI には、(相対 URI でなく) サーバー相対 URI を指定する必要があります。WebSEAL に仮想ホスト junction が使用されている場合は、これらのリソースには絶対 URI を使用する必要があります。

このセクションは、以下のトピックで構成されています。

- 97 ページの『時刻エラー・ページの使用可能化』
- 97 ページの『新規 HTML エラー・メッセージ・ページの作成』
- 98 ページの『WebSEAL の前のバージョンとの互換性』

時刻エラー・ページの使用可能化

このタスクについて

38cf08cc.html エラー・メッセージ・ページは、保護オブジェクト・ポリシー (POP) の時刻ポリシーの条件が満たされなかったためにアクセスが拒否されたときに使用されます。WebSEAL は、このエラー・メッセージ・ページの使用を構成ファイルの設定によって制御します。

WebSEAL が 38cf08cc.html を表示できるようにするには、WebSEAL 構成ファイルに次のエントリーを設定する必要があります。

```
[acct-mgt]
client-notify-tod = yes
```

client-notify-tod = yes である場合、WebSEAL は、時刻 POP アクセス・チェックが失敗したために許可が失敗したことを通知するエラー・メッセージをクライアントに送ります。

デフォルトでは、このエントリーは「no」に設定されています。

注: **client-notify-tod** に割り当てた値に関係なく、403 エラーは必ずログに記録されます。

新規 HTML エラー・メッセージ・ページの作成

このタスクについて

WebSEAL から戻される 16 進数エラー用の新規エラー・メッセージ・ページを作成できます。WebSEAL から戻される 16 進数エラーについては、「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。

例えば、WebSEAL が無効な HTTP ヘッダーを検出すると、次のエラーが戻されます。

```
wand_s_jct_invalid_http_header          0x38cf04d5
```

このエラー用の新規エラー・メッセージ・ページを作成するには、以下のステップを実行します。

手順

1. 新規 HTML ファイルを作成します。このファイルに名前を付けるには、エラー番号から 0x (16 進) 接頭部文字を削除し、.html サフィックスを付加します。例えば、0x38cf04d5 は次のようになります。

```
38cf04d5.html
```

オプションで、既存の HTTP エラー・ファイルのいずれか 1 つをテンプレートとして使用できます。それをコピーして、名前変更します。

2. 発生するエラーについて詳しくは、「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。その説明を使用して、HTML ページの本体を編集してください。

3. オプションで、87 ページの『HTML 応答ページのカスタマイズのためのマクロ・リソース』に説明があるマクロを使用できます。
4. 新規ファイルを、その他の HTTP エラー・メッセージと同じディレクトリーに保管します。

タスクの結果

特定の junction に対応する、カスタマイズされたエラー・メッセージ・ページを作成する場合は、カスタマイズされた HTML ファイルを junction 固有の場所に配置する必要があります。詳細については、86 ページの『junction 固有の静的サーバー応答ページ』を参照してください。

WebSEAL の前のバージョンとの互換性

WebSEAL バージョン 5.1 では、次のエラー・ページが新たに追加されました。

- 38cf04d7.html
- 38cf04c6.html

これらのメッセージにより、発生した失敗は WebSEAL ではなく、バックエンド・サーバーに起因していることを示す情報が提供されます。

これより前のリリースでは、WebSEAL は、デフォルトのエラー・ページだけを戻しました。前の振る舞いを保存する必要がある場合は、新規のエラー・メッセージ・ページを、エラー・メッセージ・ページ・ディレクトリーから除去してください。

サーバー応答でのマルチロケール・サポート

WebSEAL サーバーからクライアント・ブラウザへの標準応答 (エラー・メッセージ、カスタム HTML ログインおよびログアウト・ページ、保守容易性メッセージなど) は、クライアントが希望する言語で送信できます。

このセクションは、以下のトピックで構成されています。

- 『Accept-Language HTTP ヘッダー』
- 99 ページの『マルチロケール・サポート用プロセス・フロー』
- 100 ページの『WebSEAL におけるマルチロケール・サポートに影響する条件』

Accept-Language HTTP ヘッダー

WebSEAL はマルチロケール機能をサポートしており、**Accept-Language** HTTP ヘッダーに含まれている値を使用して、サーバー生成のメッセージおよび HTML ページに使用する正しい言語を判別します。翻訳されたメッセージ情報は、サーバー・メッセージ用オプション WebSEAL 言語パックをインストールすることによって提供されます。

ブラウザは、標準セットの言語値を使用します。基本言語値は、言語を示す 2 文字で表されます。地域固有の値は、言語とその言語の特定バージョンが使用されている国を示す 2 つの部分から成る形式で表されます。以下に例をいくつか挙げます。

- es (スペイン語)
- de (ドイツ語)
- en (英語)
- it (イタリア語)
- en-US (英語/米国)
- en-GR (英語/英国)
- es-ES (スペイン語/スペイン)
- es-MX (スペイン語/メキシコ)
- pt-BR (ポルトガル語/ブラジル)

Accept-Language ヘッダーには、複数の言語を含めることができます。2 番目以降の言語はそれぞれコンマで区切って追加します。例:

```
accept-language: es-mx,es,en
```

ヘッダー内での値の順序により、重要度の優先順位が決まります。WebSEAL はリスト内の最初の値を見て、それに該当する言語パックがインストールされているかどうかを確認します。その言語用の言語パックがインストールされていない場合は、WebSEAL は、リスト内の次の言語に該当する言語パックがあるかどうかをチェックします。

注: **Accept-Language** ヘッダーでは、「q=x.x」属性を使用して言語の優先レベルを表すことができます。ただし、WebSEAL はこの属性を認識しません。WebSEAL にとっては、優先順位を決定する要因はヘッダー内での言語のリスト順序です。

マルチロケール・サポート用プロセス・フロー

次のプロセス・フローの例は、WebSEAL が **Accept-Language** ヘッダーをどのように評価するかを示しています。

1. **Accept-Language** ヘッダー内のリストの最初の値が **pt-br** であるとします。
2. **pt-br** 言語は、この言語用の WebSEAL 言語サブディレクトリーを表す **pt_BR** に変換されます。
3. 必要とするメッセージについてこのサブディレクトリーが存在しない (例えば、この言語用の言語パックがインストールされていない) 場合は、WebSEAL は **pt** ディレクトリーがあるかどうか調べます。
4. **pt** ディレクトリーが存在しない場合は、WebSEAL はヘッダー内にリストされている次の言語用のメッセージ・サブディレクトリーを見つけようとします。
5. ヘッダー内にリストされたすべての言語に対して言語パックがインストールされていない場合、WebSEAL は現在 WebSEAL が稼働している言語環境をデフォルトで使用します。これは、WebSEAL が開始されたときのオペレーティング・システムの環境に設定されている **LC_ALL** または **LANG** 環境変数によって決まります。

WebSEAL におけるマルチロケール・サポートに影響する条件

- WebSEAL サーバーでは、マルチロケール・サポートは常に使用可能にされています。
- どの言語がサポートされるかは、インストールする言語パックによって決まります。
- WebSEAL は、**Accept-Language** HTTP ヘッダーのないメッセージを受信した場合、デフォルトで **C** を使用します。
- WebSEAL は、**Accept-Charset** HTTP ヘッダーの値が何を要求しているかに関係なく、常に UTF-8 文字セットをユーザーに戻します。
- WebSEAL が、翻訳されたメッセージ用のロケール・ディレクトリーにアクセスしたときに、そのディレクトリーが空であった場合 (例えば、アドミニストレーターがディレクトリーの内容を除去した場合) は、サーバー・エラー・ページが戻されます。

Mozilla Firefox での favicon.ico ファイルの処理

このタスクについて

問題の背景:

favicon.ico ファイルは、一部のブラウザ (Microsoft Internet Explorer および Mozilla Firefox など) で使用される小さなグラフィック・アイコンで、アドレス・バー情報および「お気に入り」ブックマーク・リストの表示を向上させます。リソースの要求時、これらのブラウザはサイトのカスタム favicon.ico ファイルも検索します。

ただし、Internet Explorer と Mozilla Firefox では、favicon.ico ファイルを要求するタイミングを決定する方法に違いがあります。

- Internet Explorer は、戻されたページがブックマークされている場合のみ favicon.ico を要求します。
- Mozilla Firefox は、ページの要求と同時に favicon.ico を要求します。

Mozilla Firefox ブラウザーと WebSEAL サーバー間の要求および応答の交換で favicon.ico が存在しない場合、HTTP 404 「見つかりません」というメッセージが表示される可能性があります。

保護された WebSEAL 環境では、Mozilla Firefox が favicon.ico ファイルにアクセスすると、ログイン・プロンプトが起動します。WebSEAL は、/favicon.ico を「最後に要求された URL」としてキャッシュに入れます。ユーザーがログインに成功すると、WebSEAL は要求をこの「最後に要求された URL」にリダイレクトします。そしてファイル (この例では存在しません) が見つからないため、404 「見つかりません」のエラーがユーザーに戻されます。リダイレクト処理が行われたため、最初に要求されたページにはアクセスできません。

ソリューション:

以下のステップによって、この問題を解決します。

手順

1. favicon.ico ファイルを格納します。格納先は **junction-root** ディレクトリーです。このディレクトリーには、LMI の「リバース・プロキシ管理ルートの管理」ページからアクセスできます。 とします。

2. WebSEAL 構成ファイルの **[content-mime-types]** に ico ファイル形式に関する定義を追加します。

```
[content-mime-types]
ico = image/x-icon
```

3. /favicon.ico に対する非認証アクセスを許可する ACL を作成します。 以下に例を示します。

```
pdadmin> acl show favicon
ACL name: favicon
Description:
Entries:
    Any-other Tr
    Unauthenticated Tr
    User sec_master TcmdbsvaBRr1
```

4. この ACL を /favicon.ico に明示的に付加するか、または非認証許可がこのポイントに確実に継承されるようにします。 以下に例を示します。

```
pdadmin> acl attach /WebSEAL/abc.ibm.com-default/favicon.ico favicon
```

タスクの結果

favicon.ico ファイルを作成およびインストールしない場合、以下のステップ 3 および 4 のみで問題を解決することができます。リソースが物理的に存在しなくても、リソースのオブジェクト・スペース表記に ACL を付加することができます。ブラウザではまだファイルは検出されませんが、非認証 ACL によってログイン・プロンプトは抑制されます。ブラウザは、内部で 404 エラーを処理し、要求されたページへのアクセスを続行します。

サーバー応答ページへのカスタム・ヘッダーの追加

カスタム応答に関する情報を含んだヘッダーを、生成されたサーバー応答に追加できます。

カスタム・ヘッダーの情報を定義するには、以下の表に示すマクロを使用します。

表 5. カスタム・ヘッダーの定義用マクロ

マクロ	説明
TAM_OP	応答の命令コード。このマクロの値は、ローカル応答リダイレクトの値と同一です。 106 ページの『ローカル応答リダイレクトの操作』を参照してください。
AUTHNLEVEL	認証強度ポリシー (ステップアップ) で必要な認証レベル。
ERROR_CODE	16 進のエラー・コードの値。
ERROR_TEXT	メッセージ・カタログ内のエラー・コードに対応するエラー・メッセージ・テキスト。このテキストは、WebSEAL によって入力されます。

表 5. カスタム・ヘッダーの定義用マクロ (続き)

マクロ	説明
USERNAME	ログインしたユーザーの名前。WebSEAL では、ログインしていないユーザーの場合には「unauthenticated」という値を使用します。
CREDATTR{name}	指定された <i>name</i> という名前のユーザー資格情報属性の値。例えば、 CREDATTR{tagvalue_session_index} はセッション・トークンを返します。

http-rsp-header 構成エントリーは、サーバー応答ページに組み込まれるヘッダーを定義します。この構成エントリーは、WebSEAL 構成ファイルの **[acnt-mgt]** スタンザ内にあります。

構成エントリーのフォーマットは次のとおりです。

```
http-rsp-header = <header-name>:<macro>
```

ここで、

<header-name>

値を保持するヘッダーの名前。

<macro>

101 ページの表 5 での説明に従って挿入される値のタイプ。

例えば、次の構成エントリーの **error_msg** という名前のヘッダーには、WebSEAL からのエラー・メッセージ・テキストが組み込まれます。

```
[acnt-mgt]
http-rsp-header = error_msg:ERROR_TEXT
```

注: この構成エントリーを複数回指定することにより、複数のカスタム・ヘッダーを応答に含めることができます。

例:

```
[acnt-mgt]
http-rsp-header = error_msg:ERROR_TEXT
http-rsp-header = tam-error-code:ERROR_CODE
```

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の **[acnt-mgt]** スタンザにある **http-rsp-header** 構成エントリーを参照してください。

リダイレクト応答内のロケーション URL フォーマットの構成

このタスクについて

WebSEAL が HTTP 302 リダイレクト応答を使用して要求に応答する場合、**Location** ヘッダー内の URL のフォーマットは、デフォルトで絶対パスで表されます。例:

```
Location: http://www.example.com/images/logo.jpg
```

ご使用の環境で、**Location** ヘッダー内の URL にサーバー相対フォーマットを使用する必要がある場合は、WebSEAL 構成ファイルでこのように構成することができます。

[server] スタンザに **redirect-using-relative** スタンザ・エントリーを手動で追加して、値を「true」に設定します。

```
[server]
redirect-using-relative = true
```

この構成の場合、上記の **Location** ヘッダー例は以下のように表示されます。

Location: /images/logo.jpg

この非表示構成オプションのデフォルトは、通常、「false」です。

この構成変更は、WebSEAL によって生成されたすべてのリダイレクト応答に影響します。これらのリダイレクト状態には、次のようなものがあります。

- 認証後のリダイレクト
- ログアウト後のリダイレクト
- パスワードの変更後のリダイレクト
- e-community シングル・サインオン認証プロセス中のリダイレクト
- クロスドメイン・シングル・サインオン認証プロセス中のリダイレクト
- ユーザー切り替え処理
- 証明書認証 (**prompt-as-needed** のみ)
- セッション強制終了

この構成変更は、バックエンド・アプリケーション・サーバーから戻されたりダイレクト応答には影響しません。

ローカル応答リダイレクト

このセクションは、以下のトピックで構成されています。

- 104 ページの『ローカル応答リダイレクトの概要』
- 104 ページの『ローカル応答リダイレクトのプロセス・フロー』
- 105 ページの『ローカル応答リダイレクトの使用可能化と使用不可化』
- 105 ページの『リダイレクトされた応答の内容』
- 105 ページの『ローカル応答リダイレクト用 URI』
- 106 ページの『ローカル応答リダイレクトの操作』
- 108 ページの『ローカル応答リダイレクト用マクロ・サポート』
- 112 ページの『ローカル応答リダイレクト構成の例』
- 113 ページの『ローカル応答リダイレクトの技術上の注意点』
- 113 ページの『ローカル認証によるリモート応答処理』

ローカル応答リダイレクトの概要

WebSEAL は、クライアント要求に対するサーバー応答に使用できる静的 HTML サーバー応答ページを多数提供しています。これらのページには、以下のものがあります。

- エラー・メッセージ
- 情報メッセージ
- ログイン・フォーム
- パスワード管理フォーム

環境によっては、WebSEAL がデフォルトの静的 HTML ページのセットで提供しているものよりも、サーバー応答を拡張または変更したほうがよい場合もあります。ローカル応答リダイレクトは、このような応答の処理を外部サーバーに任せる方法を提供します。

ローカル応答リダイレクトを使用する場合、WebSEAL はクライアント要求への応答を生成しません。この場合、WebSEAL のデフォルトのローカル応答は、適切な応答を生成するように設計されたカスタム・アプリケーションを実行する別のサーバーにリダイレクトされます。

ローカル応答リダイレクトが使用可能な場合、ログイン、エラー、情報、パスワード管理というすべてのローカル WebSEAL 応答タイプでリダイレクトが使用されます。

外部認証インターフェース・アプリケーションとローカル応答リダイレクト・アプリケーションを組み合わせ、認証およびサーバー応答を処理する完全なリモート・ソリューションを作成できます。また、認証処理で、外部認証インターフェースを使用せずに、ローカル応答リダイレクトをインプリメントすることもできます。この場合は、サーバー応答処理はリモートで実行され、認証は WebSEAL によってローカルで処理されます。113 ページの『ローカル認証によるリモート応答処理』を参照してください。

ローカル応答リダイレクトのプロセス・フロー

ローカル応答リダイレクト・ソリューションは、大半のデバイスが HTTP 302 リダイレクトを処理するという事実を利用します。以下のプロセス・フローでは、ローカル応答リダイレクトがハイレベルでどのような動作をするかを説明しています。

- WebSEAL がクライアント要求を受信します。
- WebSEAL が、ログイン・ページやエラー・ページなどの静的 HTML ページを戻すことによって処理される応答を戻す必要があるかどうか判別します。
- WebSEAL が、個別のサーバーに配置されたアプリケーションを処理するカスタム応答の URI を含む応答内に、ロケーション・ヘッダーを作成します。
- WebSEAL には、ロケーション・ヘッダーの照会ストリング内の属性として、必要な操作のタイプ（「ログインが必要です」、「期限切れパスワード」、または「エラー」など）と構成可能な WebSEAL マクロおよびその値のオプション・セットが含まれます。

- WebSEAL がクライアントをこのカスタム応答処理アプリケーションにリダイレクトします。
- カスタム応答処理アプリケーションがクライアントに対して適切な応答または応答セットを提示します。

ローカル応答リダイレクトの使用可能化と使用不可化

このタスクについて

WebSEAL 構成ファイルの **[acct-mgt]** スタンザ内の **enable-local-response-redirect** スタンザ・エントリーによって、ローカル応答リダイレクトを明示的に使用可能化または使用不可化することができます。有効な値は、「yes」（使用可能）および「no」（使用不可）です。

デフォルトでは、ローカル応答リダイレクトは使用不可になっています。例:

```
[acct-mgt]
enable-local-response-redirect = no
```

調整した構成項目を **[acct-mgt:{function_name}]** スタンザに追加することによって、特定の Junction 用にこの構成項目をカスタマイズできます。

ここで、**{function_name}** は標準 junction の junction ポイント (先頭の / 文字を含む)、または仮想ホスト junction の仮想ホスト・ラベルを表します。

リダイレクトされた応答の内容

ローカル応答リダイレクトの結果リダイレクトされた応答は、特別に構成されたロケーション・ヘッダーを含む標準の HTTP 302 リダイレクト応答です。

ロケーション・ヘッダーには、以下のコンポーネントが含まれています。

- 構成可能な宛先 URI。
- 必要なサーバー応答操作および構成可能な WebSEAL マクロとその値のオプション・セットを示す照会ストリング。

ロケーション・ヘッダーでは、以下の形式が使用されます。

```
Location header = location-URI?TAM_OP=operation-value[&optional-macros]
```

ロケーション・ヘッダーの各コンポーネントの詳細については、以下のセクションを参照してください。

- 『ローカル応答リダイレクト用 URI』
- 106 ページの『ローカル応答リダイレクトの操作』
- 108 ページの『ローカル応答リダイレクト用マクロ・サポート』

ローカル応答リダイレクト用 URI

WebSEAL 構成ファイルの **[local-response-redirect]** スタンザ内の **local-response-redirect-uri** スタンザ・エントリーで、応答処理サービスを提供する

カスタム・アプリケーションのロケーション (URI) を指定します。WebSEAL はこの URI を使用して、HTTP 302 応答に必要なロケーション・ヘッダーの値を作成します。

ローカル応答リダイレクトで使用するサーバーは、junction サーバーでも、完全に独立したサーバーでも構いません。

local-response-redirect-uri スタンザ・エントリーの URI 値の形式は、絶対 URI でもサーバー相対 URI でも構いません。サーバー相対 URI には、適切な junction 名が含まれている必要があります。

以下の例では、このようになります。

- **jct** は、WebSEAL junction の名前
- **redirect-app** は、カスタム・リダイレクト・アプリケーションの名前
- **response-handler** は、カスタム応答処理サービス (サーブレット、JSP、CGI など) の名前

```
[local-response-redirect]
local-response-redirect-uri = /jct/redirect-app/response-handler
```

enable-local-response-redirect= yes の場合、**local-response-redirect-uri** スタンザ・エントリーを指定する必要があります。

調整した構成項目を **[local-response-redirect:{function_name}]** スタンザに追加することによって、特定の Junction 用にこの構成項目をカスタマイズできます。

ここで、**{function_name}** は、標準 junction の junction ポイント (先頭の / 文字を含む)、または仮想ホスト junction の仮想ホスト・ラベルを表します。

ローカル応答リダイレクトの操作

WebSEAL は、クライアント要求を受信すると、要求に対する応答に必要な適切な操作を決定します。

適切に応答するには、応答処理アプリケーションが WebSEAL が決定した必要な応答操作を認識している必要があります。操作の例として、標準ログイン・フォーム、パスワード変更フォーム、またはアクセス否認エラー・メッセージの提供などがあります。

必要な操作は、HTTP 302 ロケーション URI ヘッダーの照会ストリング内で引数として提供されます。操作の引数のラベルは、**TAM_OP** です。

以下の表に、**TAM_OP** 照会ストリング引数として有効な値をリストします。

TAM_OP 操作引数の値	説明
acct_inactivated	ユーザーは認証に関する詳細を正しく入力しましたが、Sun Java System Directory Server 内のユーザーに対して nsAccountLock は true に設定されています。
acct_locked	アカウントがロックされている (無効である) ためにユーザー認証が失敗しました。

TAM_OP 操作引数の値	説明
cert_login	accept-client-certs = prompt_as_needed の場合、ユーザーは証明書を使用してログインする必要があります。
cert_stepup_http	ユーザーが HTTP を使用して証明書認証へのステップアップを試行しましたが、許可されません (HTTPS が必要)。
eai_auth_error	WebSEAL に戻された外部認証インターフェース情報が無効です。
error	エラーが発生しました。ERROR_CODE マクロ内で 16 進エラー・コードを確認してください。エラーの詳細については、「 <i>IBM Security Access Manager for Web: Error Message Reference</i> 」を参照してください。
failed_cert	クライアント証明書による認証が失敗しました。 accept-client-certs = required でのクライアントの証明書による認証が失敗しました。この接続を行うには、有効なクライアント証明書が必要です。ユーザーの証明書が無効です。
help	ユーザーが無意味なアクションを実行しました。例えば、基本認証を使用したログイン中に /pkmslogout を要求するなどのアクションです。
login	ユーザーを認証する必要があります。
login_success	ユーザーの認証に成功しましたが、リダイレクト先の最後にキャッシュされた URL が存在しません。
logout	ユーザーはログアウトしました。
passwd	ユーザーがパスワード変更を要求しています。
passwd_exp	ユーザーのパスワードの有効期限が切れています。
passwd_rep_failure	パスワード変更要求が失敗しました。
passwd_rep_success	パスワード変更要求が成功しました。
passwd_warn	パスワードの有効期限切れが近づいています。
passwd_warn_failure	パスワードの有効期限切れが近づいていることを示す通知の後に、パスワードが変更されていません。
stepup	ユーザーは別の認証レベルにステップアップする必要があります。AUTHNLEVEL マクロに必要な認証レベルがあるかを確認してください。
switch_user	ユーザーがユーザー切り替えログイン・ページを要求しました。

TAM_OP 操作引数の値	説明
too_many_sessions	許可されたセッションの最大数に達したか、または超過しました。

以下の例のヘッダーは、照会ストリングに示されたパスワード変更操作におけるロケーション URI を表しています。

Location: https://webseal/jct/handler-svr/handler?TAM_OP=passwd

ローカル応答リダイレクト用マクロ・サポート

ローカル応答リダイレクトは、静的サーバー応答ページのカスタマイズに使用する WebSEAL から提供されたマクロのサブセットをサポートします。マクロは、WebSEAL からの情報の動的置換を許可します。

操作情報 (TAM_OP 引数が提供) と同様に、マクロはロケーション・ヘッダーの照会ストリング内に引数として指定されます。マクロの値の特定文字は、URI エンコードです (111 ページの『マクロの内容のエンコード』を参照)。

ローカル応答リダイレクトで利用できる有効な WebSEAL マクロには以下のものがあります。

マクロ	説明
AUTHNLEVEL	認証強度ポリシー (ステップアップ) で必要な認証レベル。
CREDATTR{name}	指定した属性の内容をユーザー資格情報に含めるために使用されます。指定された資格情報属性が要求内にはない場合は、マクロに「Unknown」というテキストが含まれます。 例えば、セッションの秘密トークンが含まれる tagvalue_session_index 属性を含めるには、マクロ名 CREDATTR{tagvalue_session_index} を使用します。
ERROR_CODE	16 進のエラー・コードの値。
ERROR_TEXT	メッセージ・カタログ内のエラー・コードに対応する、WebSEAL 指定のエラー・メッセージ・テキスト。
FAILREASON	Boolean ルール操作に関連したエラー・メッセージ・テキスト。
HOSTNAME	完全修飾ホスト名。
HTTPHDR{name}	特定の HTTP ヘッダーの内容を含めるために使用されます。指定された HTTP ヘッダーが要求内にはない場合は、マクロに「Unknown」というテキストが含まれます。 例えば、「Host」 HTTP ヘッダーを含めるためのマクロ名は HTTPHDR{Host} になります。
METHOD	クライアントによって要求される HTTP 方式。

マクロ	説明
PROTOCOL	使用されるクライアント接続プロトコル。HTTP または HTTPS のどちらでもかまいません。
REFERER	要求からの HTTP リファラー・ヘッダーの値、または「Unknown」(値がない場合)。
URL	クライアントによって要求される URL。
USERNAME	ログインしたユーザーの名前。ログインしていないユーザーには「unauthenticated」という値が使用されます。(200 ページの『再認証用のログイン・フォームのカスタマイズ』も参照してください。) ローカル応答リダイレクトを使用する場合、WebSEAL は非アクティブ・タイムアウト後に値「unauthenticated」も使用します。この動作は、WebSEAL が静的ページを提供する際に発生する処理とは異なります。WebSEAL を構成して、静的ページを提供する際と同様に、認証ユーザー名に USERNAME マクロ値を設定することができます。この動作を可能にするには、[server] スタンザ内の use-existing-username-macro-in-custom-redirects 構成エントリを yes に設定します。この変更を有効にするには、WebSEAL を再始動する必要があります。

例えば、最初に要求された URL は、認証サービスを提供する外部認証インターフェース・サーバーにとって必要です。標準の WebSEAL URL マクロは、HTTP 302 応答のロケーション・ヘッダーの照会ストリングに指定することができます。マクロの値は、WebSEAL が動的に提供します。

マクロが提供する情報 (ロケーション・ヘッダー照会ストリングで戻される情報) を指定するには、WebSEAL 構成ファイルの **[local-response-macros]** スタンザ内の適切な **macro** スタンザ・エントリーのコメントを外します。例:

```
[local-response-macros]
macro = TAM_OP
#macro = USERNAME
#macro = METHOD
#macro = REFERER
#macro = HOSTNAME
#macro = AUTHNLEVEL
#macro = FAILREASON
#macro = PROTOCOL
#macro = ERROR_CODE
#macro = ERROR_TEXT
#macro = HTTPHDR{header_name}
macro = URL
```

注: WebSEAL は、**TAM_OP** マクロが構成ファイルに含まれているかどうかに関係なく、すべての ローカル・リダイレクト応答にこのマクロを挿入します。

以下の例のヘッダーでは、照会ストリング (1 行で入力します) で示された TAM_OP および URL マクロ (および値) のロケーション URI を表しています。

```
Location: https://webseal/jct/handler-svr/handler?TAM_OP=login&
URL=%2FjctB%2Fresource.html
```

この例では、URL マクロの値のスラッシュ文字 (/) は、%2F にエンコードされます。111 ページの『マクロの内容のエンコード』を参照してください。

構成されたマクロに情報が含まれていない場合、マクロ名および「=」区切り文字が照会ストリングに引き続き表示されます。例えば、以下のようにします (1 行で入力します)。

```
Location: https://webseal/jct/handler-svr/handler?TAM_OP=stepup&USERNAME=eric&
        FAILREASON=&AUTHNLEVEL=2
```

HTTPHDR マクロは、照会ストリング内に指定された HTTP ヘッダーを含める場合に使用します。必要なヘッダー名は、[local-response-macros] スタンザ内で構成する必要があります。例えば、照会ストリングに Host ヘッダーを挿入するには、[local-response-macros] スタンザに以下の構成エントリーを追加する必要があります。

```
macro = HTTPHDR{Host}
```

この場合、照会ストリングは以下のようになります。

```
Location: https://webseal/jct/handler-svr/handler?TAM_OP=login&HTTPHDR_Host=webseal
```

マクロ・フィールド名のカスタマイズ

生成された照会ストリングのロケーション URL マクロで WebSEAL が使用する名前をカスタマイズすることができます。これらの値を構成するには、マクロ名の後にコロンを入力し、続いてカスタマイズした名前を配置します。

例:

```
[local-response-macros]
macro = TAM_OP:myOperation
#macro = USERNAME
#macro = METHOD
#macro = REFERER
#macro = HOSTNAME
#macro = AUTHNLEVEL
#macro = FAILREASON
#macro = PROTOCOL
#macro = ERROR_CODE
#macro = ERROR_TEXT
#macro = HTTPHDR{header_name}
#macro = CREDATTR{name}
macro = URL:destination
```

この構成例では、WebSEAL により、その URL と TAM_OP マクロを含んだロケーション URI が生成されます。WebSEAL は、WebSEAL マクロ名ではなく、指定されたカスタム値を照会ストリングに挿入します。この例の場合、生成されたロケーション URI は次の照会ストリングのようになります。

```
Location: https://webseal/jct/handler-svr/handler?myOperation=login&destination=
        %2FjctB%2Fresource.html
```

HTTPHDR マクロのカスタム名を構成すると、ヘッダーの名前 (**header_name**) は照会ストリングに組み込まれません。例えば、[local-response-macros] スタンザに次のようなエントリー例があるとします。

```
macro = HTTPHDR{Host}:myHost
```

結果の照会ストリングは次の例のようになります。

Location: <https://webseal/jct/handler-svr/handler?myOperation=login&destination=%2FjctB%2Fresource.html&myHost=webseal>

複数の HTTPHDR エントリーが照会ストリングに組み込まれるように構成することができます。これらのエントリーごとに、マクロ・フィールド名をカスタマイズできます。

注: WebSEAL は、**TAM_OP** マクロが構成ファイルに含まれているかどうかに関係なく、すべての ローカル・リダイレクト応答にこのマクロを挿入します。このマクロで WebSEAL が使用する名前は、カスタマイズされた名前を構成ファイルに含めることによって構成できます。

マクロの内容のエンコード

マクロの内容には、要求された URI や要求のリファラー・ヘッダーなどのユーザー提供データが含まれている場合があります。セキュリティ上の理由から、クライアント提供データ内の予約文字または特殊文字は必ずエンコードされていることが重要です。

WebSEAL URI は、クライアントに戻されるマクロの内容に予約文字または特殊文字が含まれないようにマクロの内容をエンコードします。URI エンコードは国際標準で、世界中で広く使用されている広範な文字を URI が使用する限られた文字セットにマップできます。

マクロの内容のエンコードに関する注意:

- WebSEAL は常に、マクロの内容に URI エンコードを適用します。元データが既にエンコード済みでも関係ありません。
- エンコードされたマクロの内容は、標準の URI デコード・ルールを使用してデコードする必要があります。
- URI エンコードによってマクロの内容のストリング長が長くなるため、(マクロの内容が照会ストリングとして組み込まれている) ロケーション・ヘッダーも長くなります。ロケーション・ヘッダーの長さの問題については、『マクロの内容の長さについての考慮事項』を参照してください。

マクロの内容の長さについての考慮事項

マクロが提供する情報によって、ロケーション URI ヘッダーのストリングの長さが長くなります。また、マクロの内容の URI エンコードによって、このストリングの長さはさらに長くなります。

クライアント・アプリケーションによっては (携帯電話の WAP ブラウザーなど) デバイスのメモリー容量が小さいため、URI の長さに制限がある場合があります。このようなクライアント・デバイスで URI が長さの制限を超えた場合、エラーが発生し、リンクが欠落する可能性が高くなります。

WebSEAL は、ロケーション URI ヘッダーの長さを制限しません。そこで、ローカル応答リダイレクト用のマクロを構成する場合、サイトにアクセスするクライアント・デバイスの考えられる制限事項を慎重に考慮する必要があります。URI の固定長を決定し、照会ストリング内で使用するマクロの予想サイズを考慮することによって、ロケーション・ヘッダーの長さを見積もることができます。

以下の表で、ローカル応答リダイレクトで使用するマクロが提供する内容の可能な長さについての情報を提供します。

マクロ	内容のサイズ
AUTHNLEVEL	10 文字以内。
ERROR_CODE	20 文字以内。
ERROR_TEXT	エラー・メッセージの長さ。
FAILREASON	エラー・メッセージの長さ。
HOSTNAME	対応する要求の HOST ヘッダーの長さ、または HOST ヘッダーが存在しない場合の WebSEAL システムの完全修飾ホスト名。
METHOD	要求メソッド (GET または POST など) の長さ。20 文字以内。
PROTOCOL	10 文字以内。
REFERER	対応する要求の REFERER ヘッダーの長さ。
URL	要求 URI の長さ。
USERNAME	この WebSEAL のユーザー名の長さを定めるポリシーで定義された最大長。
HTTPHDR{name}	指定された HTTP ヘッダーの長さ。
CREDATTR{name}	ユーザー資格情報に指定された属性のコンテンツの長さ。

ローカル応答リダイレクト構成の例

以下の例で示すステップは、ローカル応答リダイレクトをインプリメントするために必要な構成を要約しています。この例では、ローカル応答リダイレクトと外部認証インターフェース・サービスを組み合わせたインプリメンテーションを説明しています。

この例では、以下の変数を使用しています。

- **jct** は、WebSEAL junction の名前
- **eai-redirect-app** は、外部認証インターフェースおよびローカル応答リダイレクト・サービスを組み合わせて作成するカスタム・アプリケーションの名前
- **authn-handler** は、カスタム認証サービス (サーブレット、JSP、CGI など) の名前
- **response-handler** は、カスタム応答処理サービス (サーブレット、JSP、CGI など) の名前

例:

- カスタム外部認証インターフェース・サービスは **junction** サーバーにインプリメントされ、WebSEAL 認証プロセスを処理します。

```
webseal/jct/eai-redirect-app/authn-handler
```

- 要求への応答を処理するには、ローカル応答リダイレクトを使用可能にします。

```
[acct-mgt]
enable-local-response-redirect = yes
```

- カスタム応答処理アプリケーションのロケーション (ロケーション URI) を指定します。


```
[local-response-redirect]
local-response-redirect-uri = /jct/eai-redirect-app/response-handler
```

- クライアントから要求された URI (URL マクロで提供) が、ローカル応答リダイレクトのロケーション URI 照会ストリングで戻されるように指定します。

```
[local-response-macros]
macro = URL
```

- クライアントは、認証を必要とするリソースを要求します。

```
https://webseal/jctB/resource.html
```

- WebSEAL は、以下のロケーション URI ヘッダー (1 行で入力します) を含む HTTP 302 応答を戻します。必要なログイン操作は、照会ストリング内に TAM_OP=login のように指定します。

```
Location: https://webseal/jct/eai-redirect-app/response-handler?TAM_OP=login&
URL=http%3A//webseal/jctB/resource.html
```

- カスタム応答処理アプリケーションは、クライアントへの応答 (ログイン操作と整合性がとれている応答) を提供し、提供されたリソース URL 情報を活用します。
- クライアントは、カスタム応答ハンドラーとの 1 つ以上の対話を完了し、最終的には、実際に認証が実行される外部認証インターフェースにルーティングされます。

ローカル応答リダイレクトの技術上の注意点

- リダイレクト・エラー・ページの結果として、WebSEAL に対してクライアントが誤ったまたはエラーの要求を行った場合、WebSEAL は再度リダイレクト操作を開始するのではなく、静的エラー・メッセージ・ページを戻します。
- 考えられるリダイレクトのループ状態を回避するには、カスタム・メッセージ処理アプリケーションを非認証ユーザーが使用できるリソースにする必要があります。このように非認証ユーザーが使用できるようにするには、Security Access Manager ACL を構成する必要があります。
- 外部認証インターフェースとローカル応答リダイレクトを一緒にインプリメントする場合、構成済みロケーション URI が外部認証インターフェース・トリガー URL と異なるようにします。
- ローカル応答リダイレクトによってリダイレクトされたすべての要求は、デフォルトの応答処理と同じ方法でキャッシュされます。

ローカル認証によるリモート応答処理

外部認証インターフェースを使用せずに、ローカル応答リダイレクトをインプリメントすることができます。この場合は、サーバー応答処理はリモートで実行され、認証は WebSEAL によってローカルで処理されます。例えば、リモート応答ハンドラーは、pkmslogin.form などのローカル WebSEAL 認証ハンドラーを必要とするログイン・ページを提供して、認証プロセスをインプリメントできます。

この例では、リモート応答ハンドラーが提供するログイン・ページに、ACTION 属性が付いた FORM タグが含まれています。ACTION 属性の値は、ローカル WebSEAL 認証ハンドラー (pkmslogin.form) のロケーションを指しています。クライアントが入力されたログイン・フォームをサブミットすると、このハンドラーにデータが送信されます。

WebSEAL は、pkmslogin.form の要求を受信すると、適切な認証メカニズムを呼び出し、このメカニズムに適切な認証データを渡すことによって応答します。

注: pkmslogin.form 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

WebSEAL によって提供された適切な静的 HTML 応答ページをカスタム・ページのテンプレートとして使用できます。必要に応じてページを編集し、ご使用の環境に合わせてコンテンツをカスタマイズします。すべての URL の表記が、WebSEAL のフィルタリング・ルールを正しく満たすようにしてください。

『ACTION URL の Junction フィルタリングの問題』を参照してください。

ACTION URL の Junction フィルタリングの問題

junction サーバーにある応答ハンドラーが提供するログイン・ページは、junction を使用してクライアントに戻されるときに WebSEAL によってフィルタリングされます。ページには、WebSEAL 標準の静的 HTML ログイン応答ページ (login.html) に似た FORM タグが含まれている場合があります。

```
<FORM METHOD=POST ACTION="/pkmslogin.form">
```

この FORM タグが、junction サーバーにあるリモート応答ハンドラーが提供するページに表示される場合、WebSEAL はパスの先頭に junction 名を付加することによってサーバー相対 ACTION URL をフィルタリングします。

```
/jct/pkmslogin.form
```

pkmslogin.form 認証ハンドラーは WebSEAL に対してローカルであるため、WebSEAL はフィルターされたバージョンの URL を検出できず、WebSEAL からの認証操作要求は失敗します。

応答ハンドラーが、pkmslogin.form を正しく指す ACTION 属性を備えたログイン・ページを提供するようにしてください。カスタム・ログイン・ページが提供されているロケーションから上位にナビゲートするには、相対パス表記を使用します。例えば、ログイン・ページが以下の JSP (JavaServer Pages) アプリケーションから提供されている場合、以下ようになります。

```
/jct/redirect-app/custom-login.jsp
```

このページの ACTION URL は、以下のように表記する必要があります。

```
<FORM METHOD=POST ACTION="../../pkmslogin.form">
```

WebSEAL は相対パス名をフィルタリングしません。したがってこのパスは、WebSEAL サーバーの文書ルート・スペースにある場合と同様に、クライアント・ブラウザによって正しく解決されます。

WebSEAL フィルタリング・ルールについての詳細は、479 ページの『タグ・ベースの静的 URL のフィルター操作ルール』を参照してください。

HTML リダイレクト

ユーザーが正常に認証されると、WebSEAL は通常、HTTP 302 応答を使用してユーザーをリダイレクトして、最初に要求されたリソースに戻します。このプロセスにより、HTML フラグメントに依存するアプリケーションで問題が発生します。これは、ブラウザではフラグメント情報がローカルに保管されており、リダイレクト中に消失するからです。

代わりに、HTML リダイレクトを使用可能にするように WebSEAL を構成できます。HTML リダイレクトにより WebSEAL は、302 リダイレクトの代わりに、静的ページをブラウザに送信します。ブラウザでは、その静的ページに組み込まれている JavaScript または他の任意のコードを使用して、リダイレクトを実行することができます。WebSEAL には、リダイレクト用の URL を含んだ LOCATION というマクロが用意されています。

HTML リダイレクトの使用可能化

このタスクについて

HTML リダイレクトを使用可能または使用不可にするには、WebSEAL 構成ファイルの `[acct-mgt]` スタンザの `enable-html-redirect` スタンザ・エントリーを使用します。HTML リダイレクトを JavaScript コードと併用して、応答に HTML フラグメントを保持することができます。

有効な値は、「yes」(使用可能) および「no」(使用不可) です。デフォルトでは HTML リダイレクトが使用不可になっています。例:

```
[acct-mgt]
enable-html-redirect = no
```

この構成エントリーを使用可能にすると、WebSEAL は、HTTP 302 応答ではなく、`[acct-mgt]` スタンザの `html-redirect` で指定されたリダイレクト・ページを返します。以下に例を示します。

```
[acct-mgt]
html-redirect = redirect.html
```

注: この方法で HTML リダイレクトを構成しても、junction Web サーバーによって返されるリダイレクト応答には影響しません。

リダイレクト時の HTML フラグメントの保持

このタスクについて

WebSEAL には、HTML リダイレクトの構成例が用意されています。この例では、Cookie を使用して元の URL をブラウザに格納し、JavaScript を使用して後からその URL を読み取り、リダイレクトを実行します。例では、認証操作中に HTML フラグメントを保存する方法を示しています。

HTML リダイレクトを使用可能にするほか、以降のリダイレクトのために HTML フラグメントを保管するようにユーザー・インターフェース・フォーム (login.html など) の JavaScript を変更する必要があります。リダイレクト前のペ

ージは、当初要求されたリソース (HTML フラグメントを備えたもの) を、クライアント Web ブラウザーの Cookie jar の Cookie に格納する必要があります。リダイレクト・ページが戻るとき、サンプルの JavaScript は Cookie の値を読み取り、HTML フラグメントを保持した上で、当初要求されたリソースにクライアントをリダイレクトします。

この構成は、WebSEAL がログイン・ページを提供する任意のログイン・メカニズムとともに使用できます。対応する WebSEAL テンプレートで **TAMOriginalURL** という Cookie を設定する JavaScript の行のコメントを外す必要があります。これらの WebSEAL テンプレートには login.html、stepuplogin.html および certlogin.html が該当します。

例えば、フォーム・ベース認証で HTML フラグメントを保持するには、以下のステップを実行する必要があります。

手順

1. **login.html** ページを更新します。JavaScript の太字になっている行のコメントを外し、クライアント・ブラウザの **TAMOriginalURL** という Cookie に、当初要求された URL を URI エンコードしたコピーを設定します。この JavaScript は、WebSEAL に付属しているデフォルトの login.html ファイルに組み込まれています。

```
<SCRIPT LANGUAGE=JavaScript>
var warningString = "<B>WARNING:</B> To maintain your login session,
    make sure that your browser is configured to accept Cookies.";
document.cookie = 'acceptsCookies=yes';
if(document.cookie == ''){
    document.write(warningString);
}
else{
document.cookie = 'acceptsCookies=yes; expires=Fri, 13-Apr-1970 00:00:00 GMT';
document.cookie = 'TAMOriginalURL=' + encodeURIComponent(window.location) +
    "; Path=/;";
}
</SCRIPT>
```

2. **[acct-mgt]** スタンザの **html-redirect** 構成エントリで、WebSEAL に付属している **redirect.html** ファイルが指定されていることを確認します。このファイルには以下のスクリプトが含まれています。このスクリプトは、ブラウザの Cookie を解析し、以前のページによって設定された **TAMOriginalURL** Cookie を検索します。この Cookie は、見つかると URI デコードされ、ただちに有効期限切れに設定されます。その後、window.location.href を含む行がリダイレクトを実行します。

```
<SCRIPT LANGUAGE=JavaScript>
var redirect = "TAMOriginalURL=";
var cookies = document.cookie.split(';');
var redirectURL = "%LOCATION%";
for(var i=0; i<cookies.length; i++) {
    var cookie = cookies[i];
    while (cookie.charAt(0)==' ') {
        cookie = cookie.substring(1, cookie.length);
        if (cookie.indexOf(redirect) == 0) {
            redirectURL = cookie.substring(redirect.length, cookie.length);
            document.cookie = 'TAMOriginalURL='; expires=Thu, 01-Jan-70 00:00:01 GMT;';
            i = cookies.length;
            break;
        }
    }
}
```

```
}  
}  
window.location.href = decodeURIComponent(redirectURL);  
</SCRIPT>
```

注: 状況によっては、リダイレクトの前に Cookie を設定できない場合もあります (ローカル応答リダイレクトが実行される場合など)。この場合のために、WebSEAL にはマクロ %LOCATION% が用意されています。このマクロは静的リダイレクト・ページに挿入されます。このマクロはリダイレクトの完全な URL を含んでおり、Cookie を設定できない場合に使用できます。ただし、この場合は HTTP フラグメントの情報がすべて失われてしまいます。

第 5 章 Web サーバー・セキュリティ構成

この章では、WebSEAL サーバーの追加セキュリティの構成について説明します。

この章で扱うトピックは以下のとおりです。

- 『Suite B 暗号のみをサポートするための WebSEAL の構成』
- 120 ページの『クロスサイト・スクリプトがもたらすぜい弱性の回避』
- 124 ページの『WebSEAL およびバックエンド・サーバー識別の抑止』
- 125 ページの『HTTP メソッドの使用不可化』
- 126 ページの『Platform for Privacy Preferences (P3P)』

Suite B 暗号のみをサポートするための WebSEAL の構成

このタスクについて

Suite B とは、米国国家安全保障局 (NSA) が 2005 年に策定した一連の暗号規格、プロトコル、およびアルゴリズムのことです。このスイートは、機密情報を保護するためのセキュリティ規格を定義します。NSA Suite B には、Advanced Encryption Standard (AES) と、鍵の交換、デジタル署名、およびハッシュ処理のための一連の暗号アルゴリズムが含まれています。

Suite B は、政府の機密通信のための NSA セキュリティ規格を、SECRET レベルまで満たしています。詳しくは、NSA の Web サイト <http://www.nsa.gov> にアクセスし、「*Suite B Cryptography*」を検索してください。

SSL 接続のネゴシエーション時にスイート B 暗号のみを使用するように WebSEAL を構成することができます。Suite B 暗号用の WebSEAL サポートを構成するには、**gsk-attr-name** エントリーおよび **jct-gsk-attr-name** エントリーを使用します。GSKit 属性 454 を値 1 に設定します。

gsk-attr-name 構成エントリーは **[ssl]** スタンザ、**[dsess-cluster]** スタンザ、および **[tfim-cluster:<cluster>]** スタンザで使用できます。**[ssl]** スタンザには **jct-gsk-attr-name** 構成エントリーも含まれています。これらのスタンザ・エントリーは、以下のように SSL 接続の初期化時に使用する追加の GSKit 属性を指定します。

[ssl] スタンザ

gsk-attr-name は、クライアントとの SSL 接続に適用されます。

jct-gsk-attr-name は、junction サーバーとの SSL 接続に適用されます。

[dsess-cluster] スタンザ

gsk-attr-name は、Session Management Server (SMS) との SSL 接続に適用されます。

[tfim-cluster:<cluster>]

gsk-attr-name は、Tivoli Federated Identity Manager との SSL 接続に適用されます。

例

以下のエントリーは、クライアント接続に Suite B 暗号のみを使用するように WebSEAL を構成します。

```
[ssl]
gsk-attr-name = enum:454:1
```

クロスサイト・スクリプトがもたらすぜい弱性の回避

クロスサイト・スクリプトは、ブラウザーに悪意あるスクリプトをデプロイするためのよく知られた技法です。データを適切にエスケープしない誤った方法で、ユーザー提供データをブラウザーに反映する Web サーバーは、このタイプの攻撃に対してぜい弱です。

クロスサイト・スクリプトの仕組みと一般的な予防手段に関する詳細情報については、<http://www.cert.org/advisories/CA-2000-02.html> で CERT の勧告を参照してください。

WebSEAL が提供する、クロスサイト・スクリプトに対する保護は、URL ストリング・フィルターを使用する Junction アプリケーションには限定的です。このタイプの攻撃に対する保護には、他のソリューション (IBM Security Web Gateway Appliance の Web コンテンツ保護機能など) も役立ちます。

URL ストリング・フィルターの構成

要求 URL に定義済みストリング・パターンが含まれる場合に着信要求を拒否するよう、WebSEAL を構成することができます。WebSEAL は、**[illegal-url-substrings]** スタンザ内に定義済みのストリング・パターンを含む着信 URL 要求を拒否します。

注: **[illegal-url-substrings]** 機能は推奨されません。IBM は、製品の以降のリリースでこの機能を削除する可能性があります。

WebSEAL 構成ファイルで、**[illegal-url-substrings]** スタンザに、WebSEAL が拒否するよう設定する各ストリング・パターンを表す個々のエントリーを追加します。例:

```
[illegal-url-substrings]
substring = <script
substring = <applet
substring = <embed
```

WebSEAL は、要求された URL 内に何らかの構成済みストリング・フラグメントを検出した場合、その要求を拒否し、400 「Bad Request」 エラー・ページを返します。

デフォルトでは、WebSEAL は <script を含むストリングをフィルターに掛けます。これ以外のフィルター操作が必要な場合は、**[illegal-url-substrings]** スタンザを作成し、すべてのサブストリングを個別にリストする必要があります。

デフォルトの動作も含む URL ストリング・フィルター機能を、完全に使用不可にすることもできます。そのためには、WebSEAL 構成ファイルに空の **[illegal-url-substrings]** スタンザを入れます。

機能に関する注:

- 構成ファイル内のサブストリング・エントリーは、ASCII でなければなりません。WebSEAL は、これらのストリングの存在を検査する前に URL をデコードします。したがって、これらのストリングが別のエンコードで URL に存在する場合、WebSEAL は引き続きフィルタリングを行います。
- WebSEAL は、大文字小文字を区別しない検索を使用して、これらのサブストリングを検出します。
- サブストリング・フィルターにはマルチバイト文字も含めることができます。

クロスサイト・リクエスト・フォージェリー (CSRF) アタックの防止

クロスサイト・リクエスト・フォージェリー (CSRF) は、悪意ある Web サイト攻撃の一種です。CSRF アタックは、ワンクリック攻撃 やセッション・ライディング とも呼ばれます。このタイプの攻撃は、Web サイトが信頼するユーザーからの無許可の要求を送信します。

CSRF は、サイトが認証済みユーザーのブラウザー内に持つ信頼を悪意ある攻撃に使用します。CSRF はリンクやスクリプトを使用して、不随意の HTTP 要求を、ユーザーが認証済みのターゲット・サイトに送信します。予防措置を取らない限り、**/pkmslogout** などの WebSEAL 管理ページは CSRF アタックからの影響を受けます。例えば、アタッカーは、認証済み WebSEAL ユーザーのブラウザーが **/pkmslogout** へのリンクをたどるよう仕向けて、ユーザーを不随意にログアウトさせる可能性があります。

このタイプのぜい弱性を軽減するよう、WebSEAL を構成することができます。

秘密トークンの検証

WebSEAL は、特定の管理操作要求には秘密トークンが含まれなければならないように構成できます。WebSEAL は、受信した要求に含まれる秘密トークンを使用して、その要求の認証性を検証します。

秘密トークンの検証は、次の WebSEAL 管理ページに影響します。

- **/pkmslogin.form**
- **/pkmslogout**
- **/pkmslogout-nomas**
- **/pkmssu.form**
- **/pkmsskip**
- **/pkmsdisplace**
- **/pkmspaswd.form**

[acnt-mgt] スタンザの **enable-secret-token-validation** 構成エントリーを使用して、秘密トークンの検証を使用可能にします。デフォルトでは、**enable-secret-token-validation** が **false** に設定されており、秘密トークンの検証は使用不可になっています。

WebSEAL で秘密トークンの検証を使用する場合は、このエントリーを **true** に設定します。

```
[acnt-mgt]
enable-secret-token-validation = true
```

秘密トークンの検証を使用可能にすると、WebSEAL は、トークンを各セッションに追加し、これらのアカウント管理要求の「token」照会引数を検証します。例えば、/pkmslogout に対する要求は pkmslogout?token=<value> に変更されます。ここで、<value> は固有のセッション・トークンです。

注: この設定では、これらの WebSEAL 管理ページの URL が変更されます。関係する各管理要求には、現在のセッション・トークンの「token」引数が含まれている必要があります。例えば、/pkmslogout?token=a861582a-c445-4462-94c9-b1074e135b9f のようになります。

秘密トークンの検証が使用可能で、token 引数が要求内に存在しないか、または実際のセッション・トークンと一致しない場合、WebSEAL は「400 Bad Request」エラー・ページを返します。

秘密トークンの検証を使用している場合、WebSEAL は、ユーザー資格情報の **tagvalue_session_index** 属性としてセッション・トークンを含めます。WebSEAL には **CREDATTR** マクロが用意されており、このマクロを使用して、資格情報属性にアクセスし、その属性を次のロケーションに挿入することができます。

- 生成された HTML ページ (例えば、/pkmshelp)。
- ローカル応答リダイレクト URL。108 ページの『ローカル応答リダイレクト用マクロ・サポート』を参照してください。
- HTTP 応答ヘッダー (**http-rsp-header** 構成項目)。101 ページの『サーバー応答ページへのカスタム・ヘッダーの追加』を参照してください。

秘密トークンを参照するには、**CREDATTR{tagvalue_session_index}** マクロを使用します。

注: 秘密トークンの検証が WebSEAL の CDSSO または eCSSO 機能に影響することはありません。

リファラー検証

CSRF 攻撃を緩和するため、着信 HTTP 要求内の **referer** ヘッダーを検証するように WebSEAL を構成することができます。WebSEAL は、構成済みの **allowed-referers** のリストとこの **referer** ヘッダーを比較して、その要求が有効かどうか判別します。

リファラー検証は、次の WebSEAL 管理ページに影響します。

- /pkmslogout
- /pkmslogout-nomas

- /pkmpasswd.form
- /pkmscdsso
- /pkmsvouchfor
- /pkmskip
- /pkmsdisplace

[acct-mgt] スタンザの **allowed-referers** 構成エントリーを使用して、有効な **referer** ヘッダーを定義します。このエントリーの値には、英数字、スペース、ピリオド、およびワイルドカード文字を含めることができます。

注: このエントリーを複数回指定して、複数の有効な **referer** ヘッダーを定義できます。WebSEAL は、これらのエントリーすべてを使用してそのリファラーを検証します。

allowed-referers は、特殊フィルターである `%HOST%` に設定できます。このフィルターは、**referer** HTTP 要求ヘッダーのホスト名の部分が **host** HTTP 要求ヘッダーと一致する場合に、リファラーが有効であることを WebSEAL に対して示します。

WebSEAL でリファラー検証を使用する場合は、少なくとも 1 つの **allowed-referers** エントリーを含める必要があります。例:

```
[acct-mgt]
allowed-referers = %HOST%
```

着信要求の検証時に、WebSEAL が要求の **referer** ヘッダーと一致する **allowed-referers** エントリーを見つけないことができない場合、その要求は失敗します。WebSEAL はエラー・ページを返します。

注: **allowed-referers** エントリーがない場合、リファラー検証は使用不可になり、WebSEAL は着信要求の **referer** ヘッダーを検証しません。

非送信請求認証要求の拒否

クロスサイト・リクエスト・フォージェリー (CSRF) に対する追加対策として、非送信請求ログイン要求をすべて拒否するように WebSEAL を構成することができます。この構成により、WebSEAL が、最初にログイン・フォームを発行することなくログイン要求を処理することのないようにします。

以下に、クライアントが WebSEAL で認証されて保護リソースにアクセスする場合の一般的なプロセスの概要を示します。

1. クライアントが保護リソースを要求します。
2. WebSEAL によりクライアントが認証されていないことが検出され、WebSEAL がログイン・フォームをクライアントに返します。
3. クライアントがログイン情報を入力し、そのフォームを WebSEAL に送信します。
4. WebSEAL は次の手順でログイン情報を処理します。
 - a. ユーザーを認証します。
 - b. セッションを作成します。

- c. リダイレクトを、要求されたリソースに送信します。
5. クライアントが保護リソースを要求します。
6. WebSEAL は、ユーザーが認証されていることを検出し、リソースをクライアントに返します。

デフォルトでは、クライアントがステップ 3 に直接進み、非送信請求ログイン要求で送信することによって WebSEAL での認証を開始することが可能です。ただし WebSEAL は、これらの非送信請求要求を拒否するように構成できます。**[server]** スタンザの **allow-unsolicited-logins** を **no** に設定すれば、クライアントがリソースへのアクセス権を取得する場合に最初の 2 つのステップが必ず行われるようにすることができます。このオプションを **no** に設定すると、WebSEAL では、非認証クライアントに対して常にログイン・フォームを送信することが必要になります。

デフォルトでは **allow-unsolicited-logins** が **yes** に設定されているため、WebSEAL は非送信請求認証要求を受け入れません。

CSRF により、アタッカーによって提供された認証データを使用してユーザーが誤って認証されることが懸念される場合は、このエントリーを **no** に設定してください。

```
[server]
allow-unsolicited-logins = no
```

WebSEAL およびバックエンド・サーバー識別の抑止

このセクションは、以下のトピックで構成されています。

- 『WebSEAL サーバー識別の抑止』
- 125 ページの『バックエンド・アプリケーション・サーバー識別の抑止』

WebSEAL サーバー識別の抑止

HTTP 応答には通常、応答を送信したサーバーの ID およびバージョンを含む **Server** ヘッダーが含まれています。

このタスクについて

以下の例では、WebSEAL から送信された応答に関するヘッダー出力を説明しています。

```
Content-Type: text/html
Date: Tue, 09 Nov 2004 02:34:18 GMT
Content-length: 515
Server: WebSEAL/6.0.0
Last-Modified: Thu, 04 Nov 2004 08:03:46 GMT
Connection: close
```

セキュリティ上の理由により、WebSEAL がクライアントの応答に **サーバー・ヘッダー** を含めないようにすることができます。

HTTP サーバー応答内の WebSEAL サーバー識別を抑止するには、以下のように、WebSEAL 構成ファイルの **[server]** スタンザ内の **suppress-server-identity** スタンザ・エントリーを「**yes**」に設定します。

```
[server]
suppress-server-identity = yes
```

デフォルトの設定は「no」です。

バックエンド・アプリケーション・サーバー識別の抑止

このタスクについて

HTTP 応答には通常、応答を送信したサーバーの ID およびバージョンを含む **Server** ヘッダーが含まれています。以下の例では、バックエンド junction アプリケーション・サーバーから送信された応答のヘッダー出力について説明しています。

```
Content-Type: text/html
Date: Tue, 09 Nov 2004 03:34:18 GMT
Content-Length: 515
Server: IBM_HTTP_SERVER/1.3.19Apache/1.3.20 (Win32)
Last-Modified: Thu, 04 Nov 2004 09:03:46 GMT
Connection: close
```

HTTP サーバー応答内のバックエンド・アプリケーション・サーバー識別を抑止するには、以下のように、WebSEAL 構成ファイルの **[server]** スタンザ内の **suppress-backend-server-identity** スタンザ・エントリを「yes」に設定します。

```
[server]
suppress-backend-server-identity = yes
```

デフォルトの設定は「no」です。

HTTP メソッドの使用不可化

このタスクについて

TRACE メソッドなどの特定の HTTP Web メソッドを使用不可にするように WebSEAL を構成することができます。ローカル・リソースまたはリモート・リソースを要求する HTTP メソッドの使用をブロックすることができます。

ローカル junction を介したリソース要求のための特定のメソッドの使用をブロックするには、**[server]** スタンザの **http-method-disabled-local** メソッドを使用します。

[server] スタンザの **http-method-disabled-remote** メソッドは、リモート・リソースを要求する HTTP メソッドを使用不可にします。

複数のメソッドを区切るにはコンマ（「,」）を使用します。例えば以下の構成エントリは、ローカル junction を介した TRACE メソッドおよび PUT メソッドへのアクセスをブロックします。

```
[server]
http-method-disabled-local = TRACE,PUT
```

デフォルトでは、WebSEAL はすべての TRACE メソッドをブロックします。WebSEAL は、ローカル・リソースおよびリモート・リソースに対する要求のための TRACE メソッドを使用不可にします。これらの構成エントリのデフォルト値は以下のとおりです。


```
[server]
http-method-disabled-local = TRACE
http-method-disabled-remote = TRACE
```

注:

HTTP の RFC 2616 では、TRACE メソッドが次のように定義されています。すなわち、「このメソッドは、要求されたメッセージの、リモートの、アプリケーション層ループバックを起動するために使用されます。要求の受信者は、起点サーバー、または、要求の中にあるゼロ (0) という Max-Forwards 値を受け取る最初のプロキシまたはゲートウェイのどちらかです。」

悪意のあるユーザーが、TRACE メソッドを使用して Web サーバーへのセキュリティ・アタックを実行することがあります。このぜい弱性を緩和するために、WebSEAL は、WebSEAL サーバーへのすべての要求の TRACE メソッドをデフォルトでブロックします。

WebSEAL 構成ファイルの中でこれら 2 つのエントリーから値 TRACE を削除することによって、TRACE メソッドを使用可能にする (ブロックを使用不可にする) ことができます。

ローカル応答に対して TRACE を含むすべての HTTP メソッドを使用可能にするには、以下のエントリーを設定します。

```
[server]
http-method-disabled-local =
```

junction された応答に対して TRACE を含むすべての HTTP メソッドを使用可能にするには、以下のエントリーを設定します。

```
[server]
http-method-disabled-remote =
```

Platform for Privacy Preferences (P3P)

このセクションは、以下のトピックで構成されています。

- 『コンパクト・ポリシーの概要』
- 128 ページの『コンパクト・ポリシーの宣言』
- 129 ページの『Junction ヘッダーの保存』
- 129 ページの『P3P ヘッダー内のデフォルト・コンパクト・ポリシー』
- 130 ページの『P3P ヘッダーの構成』
- 137 ページの『カスタム P3P コンパクト・ポリシーの指定』
- 137 ページの『P3P 構成のトラブルシューティング』

コンパクト・ポリシーの概要

WebSEAL は、Platform for Privacy Preferences (P3P) 1.0 仕様をサポートします。P3P は、プライバシー・ポリシーを機械可読形式で宣言するために使用する 1 つの規格です。この規格を使用することにより、ユーザー・エージェントは、Web サイトによって提示されるポリシーに基づいて、ある URI にアクセスすべきか、あるいは

はある Cookie を受け入れるべきかをユーザー側に立って決定できるようになります。ポリシーがない場合は、サイトのポリシーに関する前提事項のセットを基にして、決定を行います。

市販のブラウザーは、特に、Cookie を受け入れるかリジェクトするかの決定プロセスの一環として、P3P をサポートしています。Microsoft Internet Explorer 6 では、P3P ベースの Cookie フィルターがデフォルトで使用可能になっています。Mozilla を基にしたブラウザーでは、オプションの P3P Cookie フィルターを提供しています。WebSEAL は、これらのブラウザーが WebSEAL のセッション Cookie を受け入れることを確認するために P3P サポートを提供しています。

P3P 仕様書には、コンパクト・ポリシー および完全ポリシー の説明があります。コンパクト・ポリシーは完全ポリシーのサブセットです。WebSEAL には、デフォルトのコンパクト・ポリシーと、コンパクト・ポリシーのカスタマイズを使用可能にする構成設定値が用意されています。WebSEAL では、完全ポリシーを提供していません。完全ポリシーは、WebSEAL がデプロイされるベンダー、アプリケーション、または安全保護環境に特定のポリシーです。完全ポリシーのインプリメンテーションの責任はベンダー (サービス・プロバイダー) にあります。WebSEAL には、クライアントに、完全ポリシーのロケーションを指し示すために使用できる構成の設定が組み込まれています。

P3P 仕様書には、1 つの HTTP ヘッダーが持てるのは、1 つの P3P ヘッダーだけである (その他の P3P ヘッダーは無視される) と記されています。ただし、1 つの HTTP 応答には、複数の Cookie を入れることができます。したがって、HTTP ヘッダーに指定されるコンパクト・ポリシーは、応答の中のすべての Cookie に適用されます。1 つのポリシーしか存在できないので、ポリシーは、Cookie に対する実際のポリシーのうちの最も厳しいポリシーを表す必要があります。WebSEAL にとって、これは、例えば、応答の中でセッション Cookie は受け入れられたがフェイルオーバー Cookie は受け入れられなかった場合、ワーストケースの P3P ポリシーを、すべての Cookie について戻す必要があることを意味します。ワーストケースとは、ブラウザーが Cookie をリジェクトする原因となる条件の最小のセットとして定義されます。

WebSEAL は、以下の 4 つのタイプの Cookie をユーザー・エージェント (ブラウザー) に戻します。

- セッション Cookie
- フェイルオーバー Cookie
- e-community Cookie
- LTPA Cookie

e-community Cookie についてはポリシーを構成する必要はありません。Cookie の内容は、ユーザーが認証される Web サーバーのロケーションを指定することだけに限られます。この Cookie には、ユーザーを識別する情報は入りません。

セッション Cookie はセッション・データにリンクします。また、フェイルオーバー Cookie には、セッションを再構築するのに必要なセッション情報が入ります。セッション Cookie は、起点サーバー用であり、セッションが終了した後まで保存されることはなく、セッション・メンテナンスのプロセスを援助します。フェイルオーバー Cookie は、フェイルオーバー (複製された) サーバー用であり、セッショ

ンが終了した後まで保存されることはなく、また、セッション・メンテナンスのプロセスを援助します。したがって、セッション Cookie およびフェイルオーバー Cookie は同じ P3P ポリシーを持っています。したがって、Cookie 用の結合されたワーストケース・ポリシーはセッション Cookie ポリシーになります。

コンパクト・ポリシーの宣言

WebSEAL 構成ファイルには、World Wide Web Consortium Platform for Privacy Preferences 仕様書に明記されているコンパクト・ポリシーの XML 構文に一致する構成オプションのセットが用意されています。完全な仕様書には、下記の URL からアクセスできます。

<http://www.w3.org/TR/P3P/>

WebSEAL には、コンパクト・ポリシーの以下の XML エlement にマップする構成ファイル・エントリが用意されています。

- **access**

サイトが、さまざまな種類の情報へのアクセスを提供しているかどうかを示します。

- **categories**

Cookie に保管されている情報のタイプ。

- **disputes**

Cookie に含まれている情報についての紛争に関する一部の情報を、完全な P3P ポリシーに含めるかどうかを指定します。

- **non-identifiable**

データ (Web ログを含む) が収集されないか、データを収集する組織が、そのデータを無名にするかのどちらかを示します。

- **目的**

Web に関係のあるデータ処理の目的を示します。

- **recipients**

データが配布される先のサービス・プロバイダーおよびそのエージェントの先のリーガル・エンティティまたはドメイン。

- **remedies**

ポリシー不履行が発生した場合の救済方法。

- **retention**

有効になっている保存ポリシーのタイプ。

- **p3p-element**

圧縮ポリシーの他に P3P ヘッダーに追加するElement を指定します。このElement を使用して、完全 XML ポリシーへの参照を提供できます。

purpose (current を除く) および recipients (ours を除く) の値には、Cookie データの使用方法を説明する追加オプションがあります。このオプションによって、ユーザーがオプト・インまたはオプト・アウトを選択できるか否かを定義できます。

Junction ヘッダーの保存

WebSEAL を使用すると、junction 先アプリケーションの P3P ヘッダーを保存するか置き換えるかを指定できます。この機能は、P3P コンパクト・ポリシーの一部ではなく、WebSEAL の機能であることに注意してください。

構成ファイルのエントリは次のようになります。

```
[p3p-header]
preserve-p3p-policy = {yes|no}
```

デフォルトの設定は「no」です。つまり、junction 先サーバーの P3P ヘッダーは置き換えられることを意味します。

デフォルトでは、WebSEAL はバックエンド P3P ポリシー・ヘッダーを置き換え、バックエンド・サーバーによって設定されたより厳しいポリシーのために WebSEAL Cookie が除外されていないことを確認します。

デフォルト設定を使用しているとき、バックエンド・サーバーが設定する Cookie が、WebSEAL のコンパクト・ポリシーのために許可されない場合があります。この場合は、以下のオプションのいずれかを選択してください。

- preserve-p3p-policy = yes を設定し、WebSEAL を強制して、バックエンド・サーバーによって設定されたコンパクト・ポリシーを保存させる。
- WebSEAL コンパクト・ポリシーのヘッダーを変更し、ポリシーをより寛大にして、バックエンド Cookie が許可されるようにする。

WebSEAL がバックエンド・サーバーからの応答を処理すると、WebSEAL のアクションに、応答への Cookie の追加を含めることができます。この追加が起こるのは、junction Cookie を生成するために WebSEAL junction が作成されることです。これらの Cookie を使用して、junction 全体の URL をマップし、ブラウザとバックエンド・サーバーとの間の接続を確認できます。アドミニストレーターが、バックエンド・サーバーによって設定された (preserve-p3p-policy = yes) コンパクト・ポリシーを保存すると決めたら、アドミニストレーターは、コンパクト・ポリシーが WebSEAL junction Cookie の追加を受け入れるだけ寛大であることを確認する必要があります。コンパクト・ポリシーが junction Cookie の追加を許さないときは、ブラウザからの URL 要求は、バックエンド・サーバーの URL に正常に解決できません。

P3P ヘッダー内のデフォルト・コンパクト・ポリシー

WebSEAL は、Cookie が設定されるすべての応答に P3P ヘッダーを追加します。ヘッダーには P3P コンパクト・ポリシーが入っています。ポリシーは、応答内の Cookie に入っている情報に関するポリシーを記述した一連の条件です。

次の WebSEAL 構成ファイル・エントリは、デフォルトの P3P コンパクト・ポリシーを表しています。

```
[p3p-header]
access = none
purpose = current
purpose = other-purpose:opt-in
recipients = ours
retention = no-retention
categories = uniqueid
```

デフォルトのこの構成ファイル・エントリーから、次の内容の P3P ヘッダーが生成されます。

```
P3P: CP="NON CUR OTPi OUR NOR UNI"
```

次の表で、デフォルト・ポリシー・ヘッダー内の値について説明します。

表 6. P3P デフォルト・ヘッダーの値

用語	定義
NON	ユーザーは、Cookie 内の情報、あるいは Cookie のリンク先の情報へのアクセスを持っていません。
CUR	Cookie は、現行サービスの提供をヘルプします。現行サービスは、保護 Web サイトへのアクセスです。
OTPi	Cookie は、ユーザーがオプト・インを選択した別のサービスを提供します。
OUR	Web サイト自体が Cookie の唯一の受信者であり、また、Cookie のリンク先の情報です。
NOR	Cookie データも Cookie のリンク先のデータも、ユーザーがログアウトした後、またはユーザー・セッションが満了した後は保存されません。
UNI	Cookie は、セッション ID とユーザー名を使用することにより、ユーザーを表す固有 ID を使用します。

P3P ヘッダーの構成

このタスクについて

Web サーバーのセキュリティー・ソリューションの一環として WebSEAL サーバーをデプロイするアドミニストレーターは、その Web サイト用の P3P コンパクト・ポリシーを指定する必要があります。このステップでは、P3P 仕様書で定義されているプライバシー設定値のそれぞれについてポリシーを決定することが必要になります。WebSEAL では、Microsoft Internet Explorer 6 ブラウザーのデフォルトの設定値で受け入れられているデフォルト・ポリシーを提供しています。Web アドミニストレーターは、Cookie 内のユーザー・データを処理するためのサイト・ポリシーに一致するように、必要に応じてデフォルト・ポリシーを変更する必要があります。Web アドミニストレーターは Internet Explorer 6 でポリシーのテストを行い、WebSEAL Cookie が Internet Explorer 6 ブラウザーで引き続き受け入れられることを確認する必要があります。

Web アドミニストレーターは、サイト・ポリシーを定義する際に P3P 仕様書を参照する必要があります。

各構成エントリーには複数の値を使用できます。ただし、「yes」または「no」という値を必要とするエントリーは除きます。特定の構成エントリーが宣言されていない場合、そのエントリーのコンパクト・ポリシーにはインディケーターは追加されません。

WebSEAL で使用する P3P コンパクト・ポリシーを構成するには、以下のステップを実行します。

手順

- 1. 編集するために WebSEAL 構成ファイルをオープンします。[server] スタンザに進みます。
- 2. junction 先サーバーにある P3P ヘッダーを置き換えるか保存するかを決めます。値 [p3p-header] preserve-p3p-policy = {yes|no} を設定します。デフォルト値は「no」です。P3P ヘッダーを保存する場合は、これを「yes」に設定します。詳細については、129 ページの『Junction ヘッダーの保存』を参照してください。
- 3. [p3p-header] スタンザに進みます。Cookie 内の情報にアクセスするためにユーザーが持つアクセスを指定します。以下のエントリーの値を設定します。

[p3p-header]
access = {none|all|nonident|contact-and-other|ident-contact|other-ident}

デフォルト設定は、以下のとおりです。

[p3p-header]
access = none

表 7. access エントリー用にサポートされている値

値	説明
none	指定されたデータへのアクセスは行えません。
all	指定されたデータのすべてにアクセスできます。
nonident	Web サイトは、指定されたデータの収集を行いません。
contact-and-other	指定されたオンラインと物理的連絡先情報、および特定のその他の指定済みデータにアクセスできます。
ident-contact	指定されたオンラインと物理的連絡先情報にアクセスできます。例えば、ユーザーは、住所などにアクセスできます。
other-ident	指定されたその他の特定データにアクセスできます。例えば、ユーザーは、自分のオンライン・アカウント・チャージなどにアクセスできます。

- 4. Cookie に保管されている情報、あるいは、Cookie のリンク先の情報のタイプを指定します。以下のエントリーの値を設定します。

[p3p-header] categories = {physical|online
|uniqueid|purchase|financial|computer
|navigation| interactive|demographic
|content|state|political|health
|preference|location|
government|other-category}

デフォルト設定は、以下のとおりです。

[p3p-header]
categories = uniqueid

表 8. *categories* エントリー用にサポートされている値

値	説明
physical	個人が連絡されることを許す情報、あるいは物理的世界に存在している情報。例えば、電話番号または住所。
online	個人が連絡されることを許す情報、あるいはインターネットに存在している情報。
uniqueid	金融に関係しない ID で、個人を識別または認識する目的で発行されているもの。ただし、政府が発行した ID を除く。
purchase	製品またはサービスの購入によってアクティブに生成される情報 (支払いのメソッドに関する情報を含む)。
financial	個人の金融関係の情報で、アカウント状況および勘定残高、支払い、当座貸越ヒストリーなどのアクティビティー情報、および金融商品類の売買情報、クレジット・カード/デビット・カードの使用情報などを含む。
computer	ネットワークにアクセスするために個人が使用しているコンピューター・システムに関する情報。例えば、IP アドレス、ドメイン名、ブラウザのタイプ、またはオペレーティング・システム。
navigation	Web サイトをブラウズすることによってパッシブに生成されたデータ。例えば、ユーザーがどのページを参照したか、どれだけ長く参照したかなどのデータ。
interactive	そのサイトのサービス・プロバイダーとの明示的相互作用を反映するデータ、あるいはその相互作用からアクティブに生成されたデータ。例えば、検索エンジンへの照会、またはアカウント・アクティビティーのログなど。
demographic	個人の特性に関するデータ。例えば、性別、年齢、所得。
content	通信の本体に含まれているワードおよび表現。例えば、E メールテキスト、掲示板の掲載物、チャット・ルーム通信など。
state	ユーザーとのステートフル・セッションを維持するためのメカニズム、あるいは、前に特定のサイトを訪問した、または特定のコンテンツにアクセスしたユーザーを自動的に認識するメカニズム。例えば、HTTP Cookie。
political	宗教団体、労働組合、職業団体、政党などのグループのメンバーシップまたは関連。
health	個人の心身の健康状態、性的指向、健康管理サービスまたは製品の使用、購入、照会などに関する情報。
preference	個人の好みに関するデータ。例えば、好きな色、好きな音楽など。
location	個人の現住所を識別する、あるいは、住所変更をトラッキングするために使用できる情報。例えば、全地球測位システムによる位置データ。
government	個人を識別する目的で政府が発行した ID。
other-category	上記の定義では捕そくできないその他のタイプのデータ。

5. 完全 P3P ポリシーに、Cookie に入っている情報に関する論争についての情報を入れるかどうかを指定します。以下のエントリーの値を設定します。

```
[p3p-header]
disputes = {yes|no}
```


デフォルトでは、**disputes** エントリーは WebSEAL 構成ファイルには指定されていません。P3P 仕様書には、**dispute** エントリーが指定されない場合は、デフォルト値 **no** が自動的に割り当てられると明記されています。

表 9. *disputes* エントリー用にサポートされている値

値	説明
yes	完全 P3P ポリシーに、Cookie に入っている情報に関する論争についての情報を入れます。
no	完全 P3P ポリシーに、Cookie に入っている情報に関する論争についての情報を入れません。

6. ポリシー不履行が発生した場合の救済方法のタイプを指定します。以下のエントリーの値を設定します。

```
[p3p-header]
remedies = {correct|money|law}
```

デフォルト設定は、以下のとおりです。

```
[p3p-header]
remedies = correct
```

表 10. *remedies* エントリー用にサポートされている値

値	説明
correct	プライバシー・ポリシーに関連して発生したエラーまたは不当なアクションは、サービスによって救済されます。
money	サービス・プロバイダーがプライバシー・ポリシーに違反した場合は、サービス・プロバイダーが個人に、人間が理解できるプライバシー・ポリシーに指定されている金額または損害額を支払います。
law	ポリシー・ステートメントの不履行に対する救済方法は、人間が理解できる文書に記載されている法律に基づいて決定されます。

7. データ (Web ログを含む) が収集されないか、あるいは、データを収集する組織が、ユーザーを識別する情報をすべて無名にするかのどちらかを指定します。以下のエントリーの値を設定します。

```
[p3p-header]
non-identifiable = {yes|no}
```

non-identifiable エントリーは WebSEAL 構成ファイルには指定されていません。P3P 仕様書には、**non-identifiable** エントリーが指定されない場合は、デフォルト値が自動的に **no** に割り当てられると明記されています。

表 11. *non-identifiable* エントリー用にサポートされている値

値	説明
yes	収集されたデータはユーザーを識別します。
no	データ (Web ログを含む) が収集されないか、あるいは、収集された情報はユーザーを識別しません。

8. Cookie の中にある情報を指定します。以下のエントリーの値を設定します。


```
[p3p-header]
purpose = {current|admin|develop|tailoring|pseudo-analysis|pseudo-decision|
individual-analysis|individual-decision|contact|historical|
telemarketing|other-purpose} [:[opt-in|opt-out|always]]
```

デフォルト設定は、以下のとおりです。

```
[p3p-header]
purpose = current
```

表 12. *purpose* エントリー用にサポートされている値

値	説明
current	情報は、情報が提供されているアクティビティーを完了するためにサービス・プロバイダーによって使用されます。
admin	情報は、Web サイトとそのコンピューター・システムのテクニカル・サポートのために使用できます。
develop	情報は、サイト、サービス、製品、または市場を拡張、評価、あるいはその他の目的で検討するために使用できます。
tailoring	情報は、サイト (情報はサイトに 1 回訪問するためだけに使用されている) の内容または設計を調整あるいは変更するために使用できます。
pseudo-analysis	情報は、匿名 ID に結びつけられている特定の個人またはコンピューターのレコードを作成またはビルドする (ただし、識別されたデータをレコードに結びつけない) ために使用できます。このプロファイルは、リサーチ、分析、および報告書作成の目的で、個々人の習慣、興味、あるいはその他の特性を判別するために使用されます。
pseudo-decision	情報は、匿名 ID に結びつけられている特定の個人またはコンピューターのレコードを作成またはビルドする (ただし、識別されたデータをレコードに結びつけない) ために使用できます。このプロファイルは、個人に直接影響を与える決定を下す目的で、個々人の習慣、興味、あるいはその他の特性を判別するために使用されます。
individual-analysis	情報は、リサーチ、分析、および報告書作成の目的で、個々人の習慣、興味、あるいはその他の特性を判別し、これを識別されたデータに結びつけるために使用できます。
individual-decision	情報は、個人に直接影響を与える決定を下す目的で、個々人の習慣、興味、あるいはその他の特性を判別し、これを識別されたデータに結びつけるために使用できます。
contact	情報は、製品またはサービスの販売促進のために、通信チャネル (電話を除く) を介して個人に連絡をとるために使用できます。
historical	情報は、現存する法律またはポリシーで規定されている方法で、社会のヒストリーを保存する目的でアーカイブまたは保管できます。
telemarketing	情報は、製品またはサービスの販売促進のため、電話を使用して個人に連絡をとるために使用できます。
other-purpose	情報は、上記の定義では捕そくできないその他の方法で使用できます。

purpose で指定した値のすべてに (ただし値 **current** は除く)、オプションで **opt-in** ポリシーを指定することができます。構文は、**purpose** の値のすぐ後にコロン (:) を続け、その直後に **opt-in** ポリシーでサポートされている値のいずれかを指定します。例: [p3p-header] **purpose** = telemarketing:opt-in。

次の表にサポートされている値を示します。

表 13. *opt-in* または *opt-out* ポリシー用にサポートされている値

値	説明
opt-in	データは、ユーザーがこの目的のために使用することを積極的に要求した場合にのみ使用できます。
opt-out	データは、ユーザーが、この方法でデータを使用しないことを要求していない場合に、この目的に使用できます。
always	ユーザーは、ユーザーのデータをこの目的で使用するのを、オプト・インあるいはオプト・アウトできません。これはデフォルト値です。opt-in ポリシーが指定されていないときは、always ポリシーが適用されます。

9. Cookie 中の情報の受信者を指定します。以下のエントリーの値を設定します (受信者の値を 1 行で入力します)。

```
[p3p-header]
recipient = {ours|delivery|same|unrelated|public|other-recipient}
[:[opt-in|opt-out|always]]
```

デフォルト設定は、以下のとおりです。

```
[p3p-header]
recipient = ours
```

表 14. *recipient* エントリー用にサポートされている値

値	説明
ours	我々自身 (Ourselves) または我々のエージェントとして行動しているエンティティ (あるいはその両方)、または、我々がエージェントとして行動しているエンティティ。エージェントは、サービス・プロバイダーに代わってデータを処理するサード・パーティーです。
delivery	明記された目的の達成以外の目的でデータを使用することができる、デリバリー・サービスを実行するリーガル・エンティティ。
same	我々の施策を実行するリーガル・エンティティ。これらのリーガル・エンティティは、同等の施策方針のもとで、かれら自身の目的のためにデータを使用するリーガル・エンティティです。
unrelated	関連がないサード・パーティー。これらのリーガル・エンティティは、元のサービス・プロバイダーには不明の施策でデータを使用するリーガル・エンティティです。
public	公開フォーラム。これらは、掲示板、公開ディレクトリー、あるいは、市販の CD-ROM ディレクトリーなどの公開フォーラムです。
other-recipient	別の施策を実行するリーガル・エンティティ。これらのリーガル・エンティティは、元のサービス・プロバイダーの制約を受け、説明責任を持つリーガル・エンティティです。ただし、サービス・プロバイダーの施策で指定されていない方法でデータを使用できます。

recipient で指定した値のすべてに (ただし値 **ours** は除く)、オプションで **opt-in** ポリシーを指定することができます。構文は、**recipient** のすぐ後にコロンの (:) を続け、その直後に **opt-in** ポリシーでサポートされている値のいずれかを指定します。例: [p3p-header] recipient = delivery:opt-in 次の表にサポートされている値を示します。

表 15. Opt-in ポリシー値

値	説明
opt-in	データは、ユーザーがこの目的のために使用することを積極的に要求した場合にのみ使用できます。
opt-out	データは、ユーザーが、この方法でデータを使用しないことを要求していない場合に、この目的に使用できます。
always	ユーザーは、ユーザーのデータをこの目的で使用するのを、オプト・インあるいはオプト・アウトできません。 これはデフォルト値です。opt-in ポリシーが指定されていないときは、always ポリシーが適用されます。

10. Cookie の中にある情報を保存する期間を指定します。以下のエントリーの値を設定します。

```
[p3p-header]
retention = {no-retention|stated-purpose|legal-requirement|business-practices|
indefinitely}
```

デフォルト設定は、以下のとおりです。

```
[p3p-header]
retention = no-retention
```

表 16. retention エントリー用にサポートされている値

値	説明
no-retention	情報は、1 つのオンライン相互作用の実行中に利用するために必要な短い時間を超えて保存しません。
stated-purpose	情報は、明記された目的のために保存された後、できるだけ早い時点で廃棄されます。
legal-requirement	情報は明記された目的のために保存されますが、保存期間は、法的要件あるいは法的責任の理由により長期になります。
business-practices	情報は、サービス・プロバイダーの明記された業務施策に基づいて保存されます。
indefinitely	情報は、不定期間保存されます。

11. オプションで、完全 XML コンパクト・ポリシー・ファイルへの参照を指定します。以下のエントリーの値を指定します。[p3p-header] p3p-element = policyref=url_to_default_location_of_full_policy
デフォルトの WebSEAL 構成ファイルでは、このエントリーは存在していますがコメント化されているので、アクティブではありません。このデフォルト・エントリーは、すべての Web サイトの、完全ポリシーのデフォルト・ロケーションにあります。

```
[p3p-header] # p3p-element = policyref="/w3c/p3p.xml"
```

p3p-element が設定されていないときは、ブラウザーは、デフォルトでは /w3c/p3p.xml で完全ポリシーを探します。ブラウザーの一部には、**p3p-element** を参照せずに、直接 /w3c/p3p.xml に進むものがあることに注意してください。

注: 非認証アクセスが /w3c/p3p.xml に認可されていることを確かめてください。137 ページの『P3P 構成のトラブルシューティング』を参照してください。

カスタム P3P コンパクト・ポリシーの指定

このタスクについて

WebSEAL 構成ファイル中のエントリーに値を設定する代わりに、P3P ヘッダーの内容そのものを指定することができます。この構成は、例えば、ご使用のコンパクト・ポリシーのストリングが別のユーティリティーによって生成されていて、そのストリングを P3P ポリシーに使用したいときに便利です。

カスタム P3P コンパクト・ポリシーを指定するには、以下のステップを実行します。

手順

1. WebSEAL 構成ファイル内の事前定義ポリシー・エレメントを、コメント化または削除します。例えば、デフォルトの WebSEAL エントリーを以下のものに変更します。

```
[p3p-header]
#access =
#purpose =
#purpose =
#recipients =
#retention =
#categories =
```

2. ご使用のカスタム・コンパクト・ポリシーのストリングを **p3p-element** エントリーに追加します。

```
[p3p-header]
p3p-element = CP="your_series_of_compact_policy_abbreviations"
```

値は、いくつでも追加できます。値の順序は重要ではありません。

P3P 構成のトラブルシューティング

問題:

ブラウザーが、完全 P3P ポリシー・ファイルにアクセスできません。

ソリューション:

完全ポリシーが入っているファイルのロケーションを **p3p-element** スタンザ・エントリーを使用して指定すると、ブラウザーはそのファイルにアクセスしようとしてします。P3P 仕様書では、ブラウザーが、完全ポリシーを要求する Cookie をサブミットすることを必要としていません。Internet Explorer 6 では、完全ポリシーにアクセスするときに、セッション Cookie をサブミットしません。

したがって、完全ポリシーへのアクセスを、非認証ユーザーに認可する必要があります。ブラウザーが、ログイン・フォームまたは 401 エラーのどちらかを受け取ったときは、完全ポリシーに対するアクセス許可を変更して、非認証ユーザーがアクセスできるようにしてください。

第 6 章 ランタイム・セキュリティ・サービス外部許可サービス

このセクションは、以下のトピックで構成されています。

- 『ランタイム・セキュリティ・サービス外部許可サービスについて』
- 140 ページの『WebSEAL におけるランタイム・セキュリティ・サービス外部許可サービスの構成』
- 143 ページの『ランタイム・セキュリティ・サービス外部許可サービスのサンプル構成データ』

ランタイム・セキュリティ・サービス外部許可サービスについて

リスクベース・アクセスのランタイム・セキュリティ・サービス外部許可サービス (EAS) は、モジュラー許可サービス・プラグインです。EAS がある場合、IBM Security Access Manager for Web 許可を、独自の許可モデルへのアドオンとして使用できます。

ランタイム・セキュリティ・サービス EAS は、リスクベース・アクセスにポリシー適用ポイント (PEP) 機能を提供します。WebSEAL 要求の標準許可の一部としてリスクベース・アクセス決定を含むように、ランタイム・セキュリティ・サービス EAS を構成できます。

WebSEAL は、リスクベース・アクセスによって保護されているリソースへのアクセスに対する許可適用ポイントになります。リスクベース・アクセスについて詳しくは、IBM Tivoli Federated Identity Manager インフォメーション・センター (<http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/index.jsp>) で「リスクベース・アクセスのインストール、構成、および管理」ガイドを参照してください。

ランタイム・セキュリティ・サービス EAS プラグインは、IBM Tivoli Federated Identity Manager リスクベース・アクセス機能を使用します。EAS について詳しくは、IBM Security Access Manager for Web バージョン 7.0 インフォメーション・センター (http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome.html) の「*Authorization C API Developer Reference*」を参照してください。「*External authorization service plug-ins*」を検索してください。

ランタイム・セキュリティ・サービス EAS は、ポリシー決定ポイント (PDP) に送信する要求を構成します。この PDP は IBM Tivoli Federated Identity Manager です。PDP から受け取るポリシー決定に基づいて、EAS は以下のいずれかのアクションを実行します。

表 17. ランタイム・セキュリティ・サービス EAS アクセス決定

アクション	説明
許可	保護リソースへのアクセス権限を付与します。
義務を課して許可	ユーザーが 2 次チャレンジで正常に認証された後で、保護リソースへのアクセス権限を付与します。

表 17. ランタイム・セキュリティ・サービス EAS アクセス決定 (続き)

アクション	説明
拒否	保護リソースへのアクセスを拒否します。

PDP によって返されるポリシー決定を適用するには、WebSEAL ランタイム・セキュリティ・サービス EAS を構成する必要があります。

WebSEAL におけるランタイム・セキュリティ・サービス外部許可サービスの構成

ポリシー決定ポイント (PDP) によって返されるポリシー決定を施行するように、ランタイム・セキュリティ・サービス外部許可サービス (EAS) を構成します。

このタスクについて

この構成により、要求ごとに正しいデータがランタイム・セキュリティ・サービス EAS に確実に渡されるようになります。詳細については、139 ページの『ランタイム・セキュリティ・サービス外部許可サービスについて』を参照してください。

リスクベース・アクセス決定を WebSEAL 要求の標準許可の一部として組み込むには、以下の手順を実行してください。

手順

1. ランタイム・セキュリティ・サービス EAS に必要な HTTP ヘッダーと Cookie ヘッダーを構成します。

ランタイム・セキュリティ・サービス EAS には、要求からの許可決定データが必要です。以下の例に示すように、**[azn-decision-info]** スタンザで HTTP 要求エレメントとしてデータを指定します。これで、指定したデータがランタイム・セキュリティ・サービス EAS に渡されます。

```
[azn-decision-info]
# The following information is passed on to the
# policy decision point (PDP) for every request
# to access a resource that risk-based access protects
User-Agent = header:User-Agent
Accept-Language = header:Accept-Language
Accept = header:Accept
Accept-Encoding = header:Accept-Encoding
Accept-Charset = header:Accept-Charset
Cache-Control = header:Cache-Control
Connection = header:Connection
Pragma = header:Pragma
Missing = header:Missing
rspcode = header:rspcode
X-Requested-With = header:X-Requested-With
method = method
scheme = scheme
uri = uri
Host = header:Host
Content-Type = header:Content-Type
Transfer-Encoding = header:Transfer-Encoding
Authorization = header:Authorization
Subject-UUID = cookie:ac.uuid
```


重要: **[azn-decision-info]** スタンザの下で構成する各ヘッダーは、対応する属性構成をリスクベース・アクセス・データベースに持つ必要があります。エントリーで等号 (=) の後に指定する値は、**manageRbaRiskProfile** コマンドで構成する属性と一致する必要があります。エントリーでは大文字小文字が区別されるため、属性の大/小文字が **[azn-decision-info]** スタンザの下にある対応するエントリーと一致することを確認します。

ただし、Subject-UUID エントリーは例外です。**manageRbaRiskProfile** コマンドを使用して **cookie:ac.uuid** を属性として構成する必要はありません。Subject-UUID エントリーは、属性収集サービスがランタイム・セキュリティ・サービス EAS とともに機能するためにのみ必要です。

詳しくは、IBM Security Access Manager for Web バージョン 7.0 インフォメーション・センター (http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome.html) の「WebSEAL 管理ガイド」を参照してください。「*azn-decision-info*」を検索してください。

2. WebSEAL 構成ファイルの **[rtss-eas]** スタンザに、義務のマッピングと、ランタイム・セキュリティ・サービス EAS に固有のデータを構成します。
 - a. WebSEAL 構成ファイルを編集用を開きます。
 - b. **[rtss-eas]** スタンザを構成します。
 - c. **[rtss-cluster:<cluster>]** スタンザの **server** エントリーに、IBM Tivoli Federated Identity Manager で実行するランタイム・セキュリティ・サービスを指す URL を指定します。HTTPS URL を指定する場合、SSL エントリーを正しく構成するようにしてください。

以下のステップでは、サーバー・サイド SSL 認証の Secure Socket Layer (SSL) エントリーを構成するために使用できる方法の 1 つを説明します。

- 1) 発行者の証明書を、ランタイム・セキュリティ・サービスをホストする WebSphere Application Server インスタンスから、鍵データベース・ファイルにコピーします。
- 2) 鍵データベース・ファイル (pdsrv.kdb など) を指すように **ssl-keyfile** エントリーを構成します。
- 3) stash ファイル (pdsrv.sth など) を指すように **ssl-keyfile-stash** エントリーを構成します。

WebSEAL 用の SSL の構成について詳しくは、IBM Security Access Manager for Web バージョン 7.0 インフォメーション・センター (http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome.html) の「WebSEAL 管理ガイド」を参照してください。「鍵管理」を検索してください。

- d. **[obligations-levels-mapping]** スタンザの下で、PDP によって返される義務レベルと、WebSEAL の認証レベルとの間のマッピングを指定します。詳しくは、IBM Security Access Manager for Web バージョン 7.0 インフォメーション・センター (http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome.html) の「WebSEAL 管理ガイド」を参照してください。「認証レベルの指定」を検索してください。

3. ランタイム・セキュリティ・サービス EAS を呼び出すように保護オブジェクト・ポリシー (POP) を構成するには、以下のタスクを完了する必要があります。
 - a. POP を作成します。
 - b. WebSEAL をトリガーして EAS を呼び出すために必要な属性を作成します。
 - c. POP をオブジェクト MyJct に付加します。これは、リスクベース・アクセスによって保護される必要のあるリソースです。

EAS が呼び出されるのは、オブジェクトの有効な POP に **eas-trigger** という属性があり、値 **trigger_rba_eas** が関連付けられている場合です。

例えば、以下のコマンドを実行して、これらのタスクを完了します。

```
#pdadmin -a sec_master
Enter password: passwd0rd
pop create rba-pop
pop modify rba-pop set attribute eas-trigger trigger_rba_eas
pop attach /WebSEAL/localhost-default/MyJct rba-pop
server replicate
quit
```

注: リスクベース・アクセスによって保護されるリソースに対してのみ、ランタイム・セキュリティ・サービス EAS が使用されるように、オブジェクト・ツリーに POP を付加します。リスクベース POP をオブジェクト・スペース内のルート・オブジェクトに付加すると、その POP は管理タスクを含むすべての要求に使用されます。ランタイム・セキュリティ・サービス EAS へのこれらの呼び出しの大部分は必要がなく、パフォーマンス・スループットが低下する原因になる可能性があります。

4. WebSEAL を再始動して構成の変更を適用します。
5. pdadmin シェルを使用して、WebSEAL を再始動せずに、ランタイム・セキュリティ・サービス EAS に対するトレースとロギングを有効にします。

```
pdadmin > server task default-webseald-localhost
trace set pdweb.rtss 9 file path=rtss.log
```

注: この設定は、永続的なものではありません。WebSEAL を再始動すると、トレースは無効になります。

タスクの結果

これでランタイム・セキュリティ・サービス EAS は、PDP によって返されるリスクベース・アクセス・ポリシーの決定を施行するように構成されました。

次のタスク

属性収集サービスを構成します。詳しくは、IBM Tivoli Federated Identity Manager インフォメーション・センター (<http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/index.jsp>) の「リスクベース・アクセスのインストール、構成、および管理ガイド」を参照してください。

ランタイム・セキュリティ・サービス外部許可サービスのサンプル構成データ

サンプルの **[rtss-eas]** スタンザには、ポリシー決定を施行するためのランタイム・セキュリティ・サービス外部許可サービス (EAS) の構成詳細が含まれています。webseald-default.conf という名前のデフォルト WebSEAL 構成ファイルの **[rtss-eas]** スタンザの下に、サンプル・エントリーを追加します。

注: 以下のコード内のフォーマット、コメント、および改行は、PDF ファイルからコピーして貼り付けると、変更される可能性があります。コピーして貼り付けたコードを以下のコードと比較して、正しい構文に準拠していることを確認してください。

```
[rtss-eas]
# Specify the name of the IBM(r) Security Access Manager for Web
# trace component that the EAS uses

trace-component = pdweb.rtss

# Set this property to true if you want the EAS
# to first check with IBM(r) Security Access Manager for Web
# whether the user has permission to access the resource based
# on the ACL set.

apply-tam-native-policy = true

# Specify the context-id that is used in the requests that are
# sent by the EAS to the runtime security service.
# If the context-id parameter is not set, the WebSEAL server-name is used
# as the default value.
#
#context-id =
#
# Specify the audit logging configuration. This entry consists
# of an agent identifier that is followed by a comma-separated list
# of key-value pairs of attributes that are associated with the agent.
#
# For example, to configure the auditing of records to a file:
# audit-log-cfg = file path=/tmp/rtss-audit.log,flush=20,
#               rollover=2000000,buffer_size=8192,queue_size=48
# To send audit logs to STDOUT:
# audit-log-cfg = STDOUT
#
# If this attribute is missing or not configured, no audit
# events will be logged.

# audit-log-cfg =

# Specify the name of the runtime security services SOAP cluster
# that contains this runtime security services SOAP service.
# Also specify a corresponding [rtss-cluster:cluster]
# stanza with the definition of the cluster.

cluster-name = cluster1

[rtss-cluster:cluster1]
# Specify the definitions for a cluster of runtime security services
# SOAP servers in this stanza.

# Define the specifications of the server that you use to communicate
# with a single runtime security services SOAP server,
# which is a member of this cluster.
```

```

# Values for this entry are defined as:
#      {[0-9],}URL
# where the first digit (if present) represents the priority of the server
# in the cluster (9 being the highest, 0 being lowest). A priority of 9 is assumed
# if you do not specify a priority. The URL can be any
# well-formed HTTP or HTTPS URL.

# You can specify multiple server entries for failover and load balancing
# purposes. The complete set of these server entries defines the
# membership of the cluster for failover and load balancing.

server = 9,https://localhost:9443/rtss/authz/services/AuthzService

# Specify the maximum number of cached handles that are used when
# communicating with runtime security services SOAP.

handle-pool-size = 10

# Specify the length of time, in seconds, before an idle handle is removed
# from the handle pool cache.

handle-idle-timeout = 240

# Specify the length of time, in seconds, to wait for a response from
# runtime security services SOAP.

timeout = 240

# You can use the following optional configuration entries if
# the runtime security services SOAP server is configured to require
# basic authentication. If you leave these entries blank,
# the basic authentication header is not provided when communicating
# with the runtime security services SOAP server.

# Specify the name of the user for the basic authentication header.

basic-auth-user =

# Specify the password for the basic authentication header.
# Note: To obfuscate the value of a stanza/key,
# use the following pdadmin command:
# pdadmin> config modify keyvalue set -obfuscate
#      config-file stanza key value
# For example:
# pdadmin> config modify keyvalue set -obfuscate
#      /opt/pdweb/etc/webseald-default.conf
#      rtss-eas basic-auth-passwd passwd
# For more information about obfuscation, see Command Reference in
# the IBM Security Access Manager information center.
# Search for config modify in the pdadmin commands section.

basic-auth-passwd =
#
# The following SSL entries are optional and are only required if:
# 1. At least one server entry uses SSL (that is, contains an HTTPS
#    protocol specification in the URL).
# 2. A certificate is required other than that which is used by this server
#    when communicating with the policy server. For information about the
#    default certificate, see the [ssl] stanza.
#
# If these entries are required, but they are not specified in this stanza,
# WebSEAL uses the values in the [ssl] stanza by default.
#
#
# The name of the key database file that houses the client certificate for
# WebSEAL to use.
#

```

```

# ssl-keyfile =

#
# The name of the password stash file for the key database file.
#
# ssl-keyfile-stash =

#
# The label of the client certificate in the key database.
#
# ssl-keyfile-label =

#
# This configuration entry specifies the DN of the server (obtained from the
# server SSL certificate) that is accepted. If no entry is configured,
# all DNs are accepted. You can specify multiple domain names by including
# multiple ssl-valid-server-dn configuration entries.
#
# ssl-valid-server-dn =

#
# The entry controls whether FIPS communication is enabled with XACML SOAP.
# If no configuration entry is present, the global FIPS setting (as
# determined by the Security Access Manager policy server) takes effect.
#
# ssl-fips-enabled =

# Define the mappings between the obligation levels that the policy decision
# point (PDP) returns and the WebSEAL step-up authentication levels.
# The mapping must be one-to-one and the user must be permitted to authenticate
# only through the appropriate obligation mechanisms. These entries ensure that the
# EAS maps the obligations to the authentication levels and vice versa correctly.

[obligations-levels-mapping]
otp=3
consent=4

```

第 3 部 認証

第 7 章 認証の概要

この章では、WebSEAL 認証の基本概念について説明します。

この章で扱うトピックは以下のとおりです。

- 『認証の定義および目的』
- 『ユーザー要求内の情報』
- 150 ページの『クライアント識別および資格情報』
- 151 ページの『認証プロセス・フロー』
- 152 ページの『リソースへの認証アクセスと非認証アクセス』
- 154 ページの『サポートされる認証方式』
- 154 ページの『ユーザー・エージェントに基づく認証要求』

認証の定義および目的

認証とは、セキュア・ドメインにログインしようとする個別のプロセスまたはエンティティを識別するプロセスのことです。非認証ユーザーによる保護リソースの要求では、常に、認証試行が発生します。

- WebSEAL には、デフォルトで複数の認証方式が標準装備されています。

WebSEAL には、認証メカニズムをカスタマイズする柔軟性もあります。

- WebSEAL に対する認証が成功すると、Security Access Manager クライアント識別が作成されます。
- WebSEAL は、このクライアント識別を使用して、そのユーザーの資格情報を作成します。
- 許可サービスでは、各オブジェクトを管理する許可ポリシーを評価した後、この資格情報を使用して保護リソースに対するアクセスを許可したり拒否したりします。

ユーザー要求内の情報

WebSEAL は、認証時に、ユーザー要求に、以下の情報が付随しているか調べます。

- セッション鍵

セッション鍵は、クライアントと共に格納され、そのクライアントによって作成された WebSEAL への各要求と共に送信されるデータの一部です。セッション鍵は、WebSEAL によって使用され、一連の要求を同じクライアントからの要求として識別します。これにより、WebSEAL は、各要求へ認証を実行するオーバーヘッドを避けることができこれにより、WebSEAL は、要求ごとに認証を実行することによるオーバーヘッドを避けることができます。セッション鍵は、

WebSEAL サーバー・セッション・キャッシュに保管された関連セッション・データに対するロケーター索引です。セッション鍵は、WebSEALセッション ID ともいいます。

- **認証データ**

認証データは、ユーザー要求に含まれている情報であり、WebSEAL サーバーに対してユーザーを識別します。認証データのタイプの例として、クライアント・サイド証明書、パスワード、トークン・コードなどがあります。

WebSEAL は、ユーザー要求を受け取ると、最初に必ずセッション鍵を調べ、次に認証データを調べます。

クライアント識別および資格情報

認証すると、クライアント識別が作成されます。ユーザーの資格情報を作成するために、WebSEAL は、クライアント識別を必要とします。許可サービスでは、この資格情報を使用して、ユーザーから要求された保護リソースに対するアクセスを許可したり拒否したりします。

以下のプロセス・フローは、認証、クライアント識別、および資格情報の関係を説明しています。

1. WebSEAL は、非認証ユーザー用の非認証資格情報を常に作成します。

ACL に非認証ユーザーを特別に管理するルールを含めることができるため、非認証ユーザーは引き続きセキュア・ドメインに参加できます。

2. ユーザーが保護オブジェクトを要求し、ユーザーの認証が必要な場合、WebSEAL は、まず、ユーザー要求の認証データを調べます。

認証データには、ユーザーの物理的識別特性を表すパスワードおよび証明書などの方式固有の認証情報を含みます。

3. 認証が成功すると、**クライアント識別**が作成されます。

クライアント識別は、ユーザー名や作成される資格情報に追加される拡張属性情報を含むデータ構造です。

4. Security Access Manager は、クライアント識別情報を使用して、そのユーザーの**資格情報**を作成します。

Security Access Manager は、クライアント識別を、登録済みの Security Access Manager ユーザーと突き合わせ、このユーザーに適した資格情報を作成します。このアクションは、**資格情報の取得**と呼ばれます。

資格情報は、ユーザー名、グループ・メンバーシップ、およびユーザーのセッションと関連している特殊な拡張セキュリティ属性を含む複合構造です。資格情報は、特定のコンテキストの中でのユーザーを記述しています。資格情報は、このセッションの存続期間中のみ有効です。

許可サービスでは、各オブジェクトを管理する許可ポリシーを評価した後、この資格情報を使用して保護リソースに対するアクセスを許可したり拒否したりします。

資格情報取得が成功するのは、ユーザーが、Security Access Manager ユーザー・レジストリーに定義されたアカウントを持っている場合に限りです。

資格情報取得が失敗する（ユーザーが Security Access Manager ユーザー・レジストリーのメンバーでない）と、WebSEAL がエラーを戻します。

資格情報は、ユーザーについての情報を必要とするすべての Security Access Manager サービスで使用できます。資格情報を使用することによって、Security Access Manager は、許可、監査、および代行などの多くのサービスを実行することができます。

認証プロセス・フロー

以下の図は、外部認証インターフェース (EAI) が使用されていない場合の WebSEAL 認証の一般的なプロセス・フローを示しています。

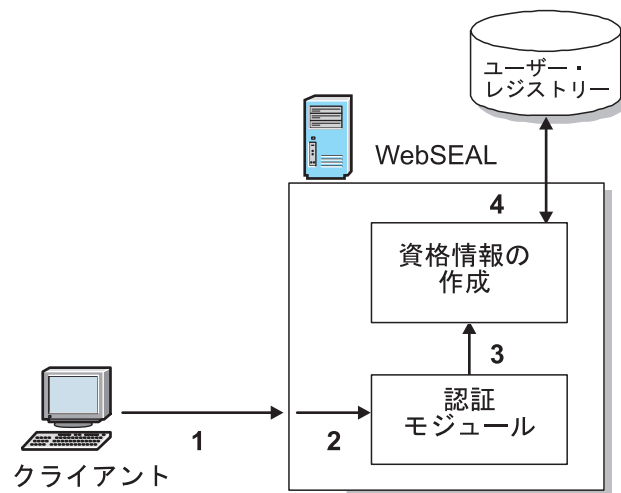


図 13. 認証プロセス・フロー

1. ユーザーは、保護オブジェクト・スペースのリソースに対する要求の間に WebSEAL に認証情報（例えば、パスワード、証明書、HTTP ヘッダーなど）を提示します。
2. WebSEAL は、そのタイプの認証情報用の構成済みの認証モジュールを呼び出します。
3. 認証モジュールは、認証情報を検証し、WebSEAL に ID を戻します。
4. WebSEAL はこの ID を使用して、ユーザー・レジストリーに保管されたそのユーザーのデータに基づいて、ユーザーの資格情報を作成します。この資格情報は、このユーザーからの要求を許可するかどうかの決定で使用されます。

注: 外部認証インターフェース (EAI) は外部認証をサポートします。

リソースへの認証アクセスと非認証アクセス

Security Access Manager 環境では、ユーザーの識別は認証のプロセスを通じて WebSEAL に対して証明されますが、WebSEAL は、HTTP と HTTPS を介して、認証済みユーザーと非認証ユーザーの両方からの要求を受け入れることができます。次いで WebSEAL は、許可サービスを使用して、保護リソースへのアクセスを許可または拒否することにより、セキュリティ・ポリシーを実施します。通常、ユーザーは、認証済み (Authenticated) または非認証 (Unauthenticated) としてセキュア・ドメインに参加できます。

どちらの場合も、Security Access Manager 認証サービスでは、セキュア・ドメイン内のリソースに対する要求を許可するかどうかを決定するために、ユーザー資格情報を必要とします。WebSEAL では、認証済みユーザーの資格情報と非認証ユーザーの資格情報は、取り扱いが異なります。

非認証ユーザー用の資格情報は、汎用パスポートで、ユーザーがセキュア・ドメインに参加し、非認証ユーザーが使用できるリソースのみにアクセスできるようにするものです。

認証済みユーザー用の資格情報は、固有のパスポートで、Security Access Manager ユーザー・レジストリーに所属している特定ユーザーを記述したものです。認証済みユーザーの資格情報には、ユーザー識別、グループ・メンバーシップ、および特殊な「拡張」セキュリティ属性が含まれています。

認証済みユーザーの要求プロセス

以下の条件は、認証済みユーザーの要求プロセスを示します。

- ユーザーが、WebSEAL で保護されているリソースに対する要求を出します。このリソースに対する保護では、ユーザーの認証が必要です。WebSEAL は、ユーザーに、ログインするようプロンプトを出します。
- 認証が成功するのは、ユーザーが Security Access Manager ユーザー・レジストリーのメンバーである場合のみです。
- このユーザー用の WebSEAL セッションおよび鍵が作成されます。
- このユーザーについてのレジストリーに含まれている情報 (グループ・メンバーシップなど) を基に、このユーザーの資格情報が作成されます。
- セッション鍵と資格情報、およびその他のデータは、1 つのエントリーとして、WebSEAL セッション・キャッシュに格納されます。
- WebSEAL は、この要求 (および同じセッション中の後続の要求) を処理するときに、資格情報をいつでも使用可能な状態にしておきます。
- 許可検査が必要になると、Security Access Manager 許可サービスが意思決定処理中に資格情報を使用します。
- ユーザーがログオフすると、そのユーザー用のキャッシュ・エントリーが削除され、セッションは終了します。

非認証ユーザーの要求プロセス

以下の条件は、非認証ユーザーの要求プロセスを示します。

- ユーザーが、WebSEAL で保護されているリソースに対する要求を出します。このリソースに対する保護では、ユーザーの認証は必要とされません。WebSEAL は、ユーザーにログインするようプロンプトを出すことはありません。
- WebSEAL は、このユーザー用の非認証資格情報を作成します。
- WebSEAL セッション・キャッシュには、エントリーは作成されません。
- 要求が、この資格情報によって保護 Web オブジェクトに送られます。
- 許可サービスがこのオブジェクトに対する ACL の非認証エントリーについての許可を検査し、要求された操作を許可または拒否します。このユーザーは、非認証タイプのユーザー・カテゴリーのための正しい許可が設定されているリソースにアクセスすることができます。
- このオブジェクトへのアクセスが正常に行われるためには、非認証 ACL エントリーに少なくとも読み取り (r) と全探索 (T) 許可が含まれていることが必要です。
- このユーザーが、非認証ユーザーが使用できないリソースへのアクセスを要求すると、WebSEAL は、ユーザーに、ログインするようプロンプトを出します。
- ログインに成功すると、このユーザーの状況は「認証済み」に変わります。
- ログインに成功しなかった場合は、403「禁止」メッセージが戻されます。ただし、このユーザーは、非認証ユーザーが使用可能なその他のリソースには、今後同様にアクセスできます。

SSL を介したアクセス条件

以下の条件は、SSL を介して WebSEAL にアクセスする非認証ユーザーに適用されます。

- 非認証ユーザーと WebSEAL の間の情報の交換は、認証ユーザーの場合と全く同じように暗号化される。
- 非認証ユーザーと WebSEAL との SSL 接続に必要なのは、サーバー・サイドの認証だけである。

ユーザーの強制ログイン

このタスクについて

要求されたオブジェクトを保護する ACL ポリシー内の非認証エントリーに適切な許可を正しく設定することによって、非認証ユーザーを強制的にログインさせることができます。

読み取り (r) および全探索 (T) 許可によって、オブジェクトへの非認証アクセスが許可されます。

非認証ユーザーを強制的にログインさせるには、オブジェクトを保護する ACL ポリシー内の非認証エントリーから読み取り (r) 許可を除去します。

ユーザーはログイン・プロンプト (基本認証またはフォーム) を受け取ります。

非認証 HTTPS の使用

以下のような、HTTPS を介した WebSEAL への非認証アクセスをサポートする多くの実践的なビジネス上の理由があります。

- 一部のアプリケーションは、個人的ログインを必要としませんが、住所やクレジット・カード番号などの機密情報を必要とします。例には、オンラインによる航空券などの商品の購入があります。
- 一部のアプリケーションでは、ユーザーが相手企業にアカウントを登録してからでないと、取引を先に進められないようになっています。この場合も、ネットワークを介して機密情報を提供する必要があります。

サポートされる認証方式

WebSEAL は、認証プロセスから独立して機能しますが、資格情報を使用して、セキユア・ドメインに参加するすべてのユーザーをモニターしています。

WebSEAL は、認証プロセスの結果の情報から、資格情報の獲得のために必要な識別情報を取得します。

次の表に、資格情報獲得のために WebSEAL でサポートされている認証方式がリストされています。WebSEAL は、クライアント要求を検査するときに、認証データを、この表の順序で検索します。

認証方式	サポートされる接続タイプ
1. フェイルオーバー Cookie	HTTP および HTTPS
2. LTPA Cookie	HTTP および HTTPS
3. CDSSO ID トークン	HTTP および HTTPS
4. クライアント・サイド証明書	HTTPS
5. フォーム認証 (ユーザー名およびパスワード)	HTTP および HTTPS
6. 基本認証 (ユーザー名およびパスワード)	HTTP および HTTPS
7. 外部認証インターフェース	HTTP および HTTPS

HTTP トランスポートと HTTPS トランスポートのどちらについても、各認証方式はそれぞれ独立して使用可能または使用不可能にすることができます。特定のトランスポートについて認証方式がどれも使用可能にされていない場合は、そのトランスポートを使用するクライアントに関しては認証プロセスは非アクティブになります。

ユーザー・エージェントに基づく認証要求

WebSEAL では、保護リソースを要求するクライアントのユーザー・エージェントに基づいて認証要求タイプを構成できるメカニズムが提供されます。このメカニズムにより、さまざまなクライアントの WebSEAL 認証方法に関する強固な統合ときめ細かい制御が可能になります。

auth-challenge-type 構成エントリーで指定される各認証タイプは、一連のルールで限定できます。これらのルールは、さまざまな認証タイプで含めるまたは除外するユーザー・エージェント・ストリングを定義します。

例: auth-challenge-type = [-msie*+ms*]ba, [+mozilla*; +msie]forms; eai

この構成例での WebSEAL の動作は次のとおりです。

- msie で始まるユーザー・エージェント・ストリングに基本認証要求を返しません
が、ms で始まるエージェントの基本認証要求を返します。
- mozilla または msie で始まるユーザー・エージェントにフォーム・ベースの認
証要求クライアントを返します。
- 任意のユーザー・エージェントに EAI 認証要求を返します。

ユーザー・エージェント・ストリング	認証要求
msie	フォーム、eai
ms_office_word	ba、eai
mozilla	フォーム、eai
chrome	eai

ルール構文

各認証要求タイプは、auth-challenge-type ストリングに一度だけ定義できます。
ルールは、セミコロンで区切られたさまざまなパターンを持つ大括弧に囲まれた認
証タイプの前になければなりません。プラス (+) またはマイナス (-) 文字は、その
ユーザー・エージェント・ストリングに要求タイプを含めるか除外するかをそれぞ
れ示します。

パターンには、英数字、スペース、ピリオド、および疑問符 (?) やアスタリスク (*)
などのワイルドカード文字を含めることができます。

WebSEAL がユーザー・エージェントに基づいてこれらのルールを評価する際に
は、パターンが現在のストリングと一致する最初のルールが適用されます。特定の
認証メカニズムと一致するその他のルールは無視されます。WebSEAL は、ルール
の定義順にこれらの評価を行います。

ルール・セットが定義されていない認証タイプは任意のユーザー・エージェント・
ストリングと一致します。

認証タイプを任意のユーザー・エージェント・ストリングと一致させない場合は、
[-*]ba などの負のワイルドカード・ストリングを使用して、特定の認証要求を指定
します。

注: ユーザー・エージェント機能に基づく認証要求をセキュリティーまたは適用手
段として使用することはできません。

第 8 章 認証方式

この章では、WebSEAL でサポートされる認証方式のコア・セットの構成方法について説明します。

認証が成功すると、ユーザーを表す Security Access Manager 識別が作成されます。WebSEAL は、この識別を使用して、ユーザーの資格情報を獲得します。資格情報によってユーザーはセキュア・ドメインに参加できます。また、資格情報は、保護リソースに対するアクセスを許可したり拒否したりするために許可サービスによって使用されます。

この章で扱うトピックは以下のとおりです。

- 『認証に関する用語』
- 158 ページの『ログアウトおよびパスワード変更の操作』
- 160 ページの『基本認証』
- 161 ページの『フォーム認証』
- 164 ページの『クライアント・サイド証明書認証』
- 174 ページの『Kerberos 認証』
- 177 ページの『LTPA 認証』

認証に関する用語

以下の用語は、この文書で認証に関して説明するのに使用します。

- **方式**

認証方式には、認証タイプのプロセスおよび戦略全体を記述します。認証方式の例を挙げますが、これだけではありません。

- ユーザー名/パスワード
- 証明書

常にとは限りませんが、通常認証方式は、ユーザーの ID を証明するために使用される特別なタイプのデータと 1 対 1 の関係を持ちます。

- **操作**

認証操作では、認証方式をサポートするあらゆるアクションを記述します。例:

- ユーザー名およびパスワードの認証中に LDAP 検索を実行する。
- ユーザー・パスワードを変更する。
- 新規パスワードが特定の基準を満たすことを検査する。
- 認証済み ID に属性を追加する。

ログアウトおよびパスワード変更の操作

Security Access Manager は、ログアウトおよびパスワード変更の操作を管理するために、認証ユーザーに以下のリソースを提供します。

- 『ログアウト: pkmslogout』
- 654 ページの 『pkmslogout-nomas を使用したログアウト』 ¥
- 『pkmslogout でのカスタム応答ページのコントロール』
- 159 ページの 『パスワードの変更: pkmspasswd』
- 159 ページの 『Windows での Active Directory のパスワード変更の問題』

ログアウト: pkmslogout

このタスクについて

一部の認証方式の場合、ユーザーは **pkmslogout** コマンドを使用して、現行セッションからログアウトすることができます。

pkmslogout コマンドは、基本認証、証明書、または IP アドレス認証のように要求ごとに認証データを提供する認証方式には適切ではありません。このような場合にログアウトするには、ブラウザーをクローズする必要があります。

手順

1. **pkmslogout** コマンドを実行して現行セッションからログアウトします。例:
`https://www.example.com/pkmslogout` この要求が実行されると、WebSEAL は、次のように WebSEAL 構成ファイルで定義されている適切なログアウト・フォームを返します。`[acct-mgt]logout = logout.html`
2. 特定の要件に合うように、該当する応答ページの内容 (`logout.html` など) を変更します。

pkmslogout でのカスタム応答ページのコントロール

pkmslogout コマンドを使用すると、デフォルトの HTML 応答ページ (`logout.html` など) をカスタム応答ページで置き換えることが可能になります。

カスタム応答ページは、**pkmslogout** URL に付加される照会ストリングを介して指定します。例:

`https://www.example.com/pkmslogout?filename=custom_logout_file`

ここで、`custom_logout_file` は、カスタム・ログアウト応答ページのファイル名です。このファイルは、LMI のデフォルトの HTML 応答フォーム (`logout.html` など) が含まれているのと同じ `management/lang` ディレクトリに配置されている必要があります。

カスタム応答ページ機能を使用すると、例えば、ユーザーが複数の異なるバックエンド・システムからログアウトするためにネットワーク体系で複数の異なる終了画面を必要とするときに、複数のログアウト応答ページを用意できます。

WebSEAL 構成ファイルの **[acct-mgt]** スタンザの **use-filename-for-pkmslogout** スタンザ・エントリーを介して、付加された照会ストリングがデフォルトの応答ページをオーバーライドすることができるかどうかをコントロールすることができます。

値「no」(デフォルト) を指定すると、照会ストリングの使用が無効になります。カスタム応答ページを指定する **pkmslogout** URL 内の照会ストリングは無視されます。デフォルト応答ページのみがログアウト時に使用されます。

値「yes」(デフォルト) を指定すると、照会ストリングの使用が有効になります。**pkmslogout** URL 内の照会ストリングがカスタム応答ページを指定する場合、デフォルト・ページの代わりに、そのカスタム・ページが使用されます。例:

```
[acct-mgt]
use-filename-for-pkmslogout = yes
```

333 ページの『古いセッション Cookie に対するカスタマイズ応答』も参照してください。

パスワードの変更: pkmspasswd

このタスクについて

このコマンドは、基本認証 (BA) またはフォーム認証を使用しているときに、ログイン・パスワードを変更するために使用できます。例:

```
https://www.example.com/pkmspasswd
```

タスクの結果

WebSEAL で BA を使用しているときに最大限のセキュリティを確保するために、このコマンドを実行すると BA クライアントに対し、次の動作が行われます。

1. パスワードが変更されます。
2. クライアント・ユーザーは、現行セッションからログアウトされます。
3. クライアントがさらに追加要求を出すと、ブラウザーはクライアントに BA プロンプトを表示します。
4. クライアントは、要求を続行するには再度ログインする必要があります。

このシナリオが適用されるのは、基本認証を使用しているクライアントのみです。

注: pkmspasswd 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

Windows での Active Directory のパスワード変更の問題

以下の問題は、Security Access Manager ユーザー・レジストリーとして Active Directory を使用するとき、Active Directory サーバーが Windows で実行しているときのパスワード変更で発生します。特定の Active Directory ポリシー設定によっては、パスワード変更が行われた後でも、Security Access Manager にログインするのに旧パスワードが使用されます。デフォルトでは、パスワードを変更してから約 1 時間、旧パスワードと新規パスワードを使用することができます。1 時間後に、旧パスワードの使用は停止になります。

Windows は、この動作を Active Directory に導入しました。発生内容に関する情報、および必要に応じてこの動作を無効にする手順については、Microsoft サポート技術情報 (KB) の記事 906305 を参照してください。

<http://support.microsoft.com/?id=906305>

基本認証

基本認証 (BA) は、認証メカニズムにユーザー名とパスワードを提供する標準的な方式です。BA は、HTTP プロトコルにより定義され、HTTP および HTTPS を介してインプリメントすることができます。

WebSEAL は、デフォルトでは、HTTPS を介した基本認証 (BA) 用に構成されています。

このセクションは、以下のトピックで構成されています。

- 『基本認証の使用可能化および使用不可』
- 161 ページの『レルム名の設定』

基本認証の使用可能化および使用不可

このタスクについて

基本認証方式を使用可能または使用不可にするには、WebSEAL 構成ファイルの **[ba]** スタンザにある **ba-auth** スタンザ・エントリーを使用します。

基本認証は、デフォルトで使用可能になっています。基本認証を構成するには、次のようにします。

手順

- WebSEAL サーバーを停止します。
- WebSEAL 構成ファイルを編集します。**[ba]** スタンザで、ご使用のネットワーク環境でサポートするプロトコルを指定します。サポートするプロトコルを、次の表に示します。

表 18. 基本認証の構成

サポートするプロトコル	構成ファイル・エントリー
HTTP	ba-auth = http
HTTPS	ba-auth = https
HTTP および HTTPS	ba-auth = both
基本認証の使用不可化	ba-auth = none

例えば、両方のプロトコルをサポートするには次のようにします。

```
[ba]
ba-auth = both
```

- WebSEAL サーバーを再始動します。

レルム名の設定

このタスクについて

レルム名は、ブラウザーがユーザーにログイン・データの入力を求めたときに現れるダイアログ・ボックスに表示されるテキストです。また、レルム名は、ユーザー・ログインが成功したときに、ユーザーを認証するレルムの名前です。

WebSEAL 構成ファイルの **[ba]** スタンザにある **basic-auth-realm** スタンザ・エントリーは、レルム名を設定します。

例えば、次のとおりです。

```
[ba]
basic-auth-realm = Access Manager
```

ダイアログ・ボックスには、例えば次のように表示されます。

Enter username for Access Manager at www.ibm.com:

フォーム認証

Security Access Manager は、標準の基本認証メカニズムに代わる方式として、フォーム認証方式を提供しています。この方式では、基本認証試行による標準のログイン・プロンプトの代わりに、Security Access Manager からカスタム HTML ログイン・フォームが生成されます。

フォーム・ベースのログインを使用すると、基本認証の場合とは異なり、ブラウザーが、ユーザー名とパスワードの情報をキャッシュに入れることはありません。

このセクションは、以下のトピックで構成されています。

- 『フォーム認証の使用可能および使用不可』
- 162 ページの『HTML 応答フォームのカスタマイズ』
- 162 ページの『WebSEAL へのログイン・フォーム・データの直接送付』

フォーム認証の使用可能および使用不可

このタスクについて

フォーム認証方式を使用可能または使用不可にするには、WebSEAL 構成ファイルの **[forms]** スタンザにある **forms-auth** スタンザ・エントリーを使用します。

デフォルトでは、フォーム認証は使用不可になっています。フォーム認証を構成するには、次のようにします。

手順

1. WebSEAL サーバーを停止します。
2. WebSEAL 構成ファイルを編集します。**[forms]** スタンザで、ご使用のネットワーク環境でサポートするプロトコルを指定します。 サポートするプロトコルを、次の表に示します。

表 19. フォーム認証の構成

サポートするプロトコル	構成ファイル・エントリー
HTTP	forms-auth = http
HTTPS	forms-auth = https
HTTP および HTTPS	forms-auth = both
フォーム認証の使用不可化 (デフォルト)	forms-auth = none

例えば、両方のプロトコルをサポートするには次のようにします。

```
[forms]
forms-auth = both
```

3. WebSEAL サーバーを再始動します。

HTML 応答フォームのカスタマイズ

フォーム認証では、login.html というカスタム・ログイン・フォームを使用する必要があります。

このタスクについて

LMI を使用してデフォルトの login.html フォームにアクセスすることができます。

このフォームの内容と設計はカスタマイズできます。

カスタマイズできる HTML フォームについての詳細は、79 ページの『静的 HTML サーバー応答ページ』を参照してください。

WebSEAL へのログイン・フォーム・データの直接送付

このタスクについて

WebSEAL によるプロンプト表示なしで WebSEAL にフォーム 認証を実行することができます。

以下の手順で、ユーザーが WebSEAL からログイン・フォームの入力を要求される典型的な WebSEAL ログインが発生するイベントについて説明します。

手順

1. ユーザーが保護リソースを要求します。
2. WebSEAL がユーザーの要求をキャッシュに入れます。
3. WebSEAL は、ユーザーにログイン・フォームを戻します。
4. ユーザーがログイン・フォームのフィールドを入力し (ユーザー名およびパスワードを入力)、「送信」ボタンをクリックします。
5. 「送信」ボタンによって、/pkmslogin.form への POST 要求が起動されます。
要求の本文には、フォーム・フィールド・データが含まれています。

注: pkmslogin.form 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

6. WebSEAL はユーザーを認証し、認証が成功すると、優先順序に従ってユーザーを以下の 3 つの場所のいずれかにリダイレクトします。
 - a. 構成済みの場合、**[acct-mgt]** スタンザ内の **login-redirect-page** エントリーで指定された場所。229 ページの『認証後の自動リダイレクト』を参照してください。
 - b. ユーザーが最初に要求したリソース (わかっている場合)。
 - c. 汎用 `login_success.html` ページ。79 ページの『静的 HTML サーバー応答ページ』を参照してください。

タスクの結果

アプリケーション統合インプリメンテーションによっては、保護リソースに対する初期要求を作成せずに、または WebSEAL からログインを求めるプロンプトの表示なしに、直接ログインが必要な場合があります。このような直接ログインは、`/pkmslogin.form` に対して直接 POST 要求を使用することで実行できます。

以下の手順で、直接ログイン中に発生するイベントを説明します。

1. クライアントは、要求の本文に適切なフォーム・フィールド・データを持つ POST 要求を `/pkmslogin.form` に送信します。
2. WebSEAL はユーザーを認証し、認証が成功すると、優先順序に従ってユーザーを以下の 2 つの場所のいずれかにリダイレクトします。
 - a. 構成済みの場合、**[acct-mgt]** スタンザ内の **login-redirect-page** エントリーで指定された場所。

229 ページの『認証後の自動リダイレクト』を参照してください。
 - b. 汎用 `login_success.html` ページ。

79 ページの『静的 HTML サーバー応答ページ』を参照してください。

POST データの形式は、以下の規則に従う必要があります。

- POST を `/pkmslogin.form` に対して実行すること。
- POST 要求の本文に、以下の 3 つのフィールドのフィールド・データが含まれること。
 - **username**
 - **パスワード**
 - **login-form-type**
- **login-form-type** の値がフォーム・ログイン用の「`pwd`」であること。
- **content-length** ヘッダーに、生成された要求の本文の長さが指定されていること。

例 (telnet を使用):

```
prompt> telnet webseal.example.com 80
Connected to webseal.example.com.
Escape character is '^]'.
POST /pkmslogin.form HTTP/1.1
host: webseal.webseal.com
content-length: 56
```

```
username=testuser&password=my0passwd&login-form-type=pwd
```

クライアント・サイド証明書認証

このセクションは、以下のトピックで構成されています。

- 『クライアント・サイド証明書認証モード』
- 167 ページの『証明書認証の構成タスクの要約』
- 167 ページの『証明書認証の使用可能化』
- 168 ページの『証明書認証メカニズムの構成』
- 170 ページの『証明書ログイン・エラー・ページ』
- 171 ページの『証明書ログイン・フォーム』
- 171 ページの『セッション・トラッキング用の SSL セッション ID の使用不可化』
- 172 ページの『証明書 SSL ID キャッシュの使用可能化と構成』
- 172 ページの『証明書 SSL ID キャッシュのタイムアウトの設定』
- 173 ページの『誤ったプロトコル用のエラー・ページ』
- 173 ページの『証明書認証の使用不可化』
- 174 ページの『証明書 SSL ID キャッシュの使用不可化』
- 174 ページの『証明書認証に関する技術上の注意点』

クライアント・サイド証明書認証モード

クライアント・サイド証明書認証を使用することにより、ユーザーは、クライアント・サイド・デジタル証明書を使用して、Security Access Manager セキュア・ドメイン内で使用する認証済み ID を要求できるようになります。認証が成功すると、WebSEAL は、ユーザー用の資格情報を作成するために使用する Security Access Manager ID を取得します。資格情報は、ユーザーに認可されるアクセス許可と権限を指定します。

デフォルトでは、クライアント・サイド証明書認証は使用不可になっています。

WebSEAL は、クライアント・サイド証明書認証を、下記の 3 つのモードでサポートしています。アドミニストレーターは、構成を行うときに、該当するモードを指定する必要があります。以下のセクションで、各モードについて説明します。

- 『必須証明書認証モード』
- 165 ページの『オプション証明書認証モード』
- 165 ページの『遅延証明書認証モード』

必須証明書認証モード

必須証明書認証モードでは、WebSEAL は、最初の HTTPS 要求でクライアント・サイド証明書が常に必要です。

ユーザーがリソースへのアクセスを SSL を介して要求すると、WebSEAL はそのサーバー・サイド証明書を提供し、これを使用して、ユーザーは SSL セッションを確立します。次に、WebSEAL は、ユーザーに、クライアント・サイド証明書を提示するよう要求します。

ユーザーが有効な証明書を提示しない場合、そのユーザーとの SSL 接続はクローズされ、クライアント・サイド証明書認証は試行されません。

注: 有効であるためには、証明書のデータが破損しておらず、証明書自体が証明書失効リスト (CRL) で失効済みとされていない必要があります。

有効な証明書が提示されているのに証明書の識別名 (DN) の認証または許可が失敗した場合、接続が確立され、非認証セッションが作成されます。保護リソースへのアクセスは許可されません。

オプション証明書認証モード

このモードでは、WebSEAL は、最初の HTTPS 要求でクライアント・サイド証明書を要求しますが、それが必要なわけではありません。

ユーザーがリソースへのアクセスを SSL を介して要求すると、WebSEAL はそのサーバー・サイド証明書を提供し、これを使用して、ユーザーは SSL セッションを確立します。次に、WebSEAL は、ユーザーに、クライアント・サイド証明書を提示するよう要求します。ユーザーがクライアント・サイド証明書を提示した場合、WebSEAL はその証明書を使用して、証明書ベースの認証セッションを開始します。ユーザーがクライアント・サイド証明書を提示しなかった場合、WebSEAL は SSL セッションが続行することを許しますが、ユーザーは Security Access Manager に対して非認証状態のままになります。

遅延証明書認証モード

このモードでは、WebSEAL は、ユーザーが証明書ベースの認証を必要とする保護リソースにアクセスしようとするまで、クライアント・サイド証明書認証の目的でクライアント・サイド証明書を要求することはありません。

ユーザーがリソースへのアクセスを SSL を介して要求すると、WebSEAL はそのサーバー・サイド証明書を提供し、これを使用して、ユーザーは SSL セッションを確立します。WebSEAL は、要求されたリソースのセキュリティー・ポリシーを検査して、証明書認証が必要かどうかを判別します。セキュリティー・ポリシーは、保護リソースに付加されているアクセス・コントロール・リスト (ACL) または保護オブジェクト・ポリシー (POP) の内容の中で説明されています。

セキュリティー・ポリシーで証明書認証が必要とされていない場合、WebSEAL は、クライアント・サイド・デジタル証明書を要求しません。

セキュリティー・ポリシーで、証明書認証が必要である としている場合は、WebSEAL はログイン・フォームを戻します。ユーザーは、このフォーム内のボタンをクリックして、証明書の交換を開始します。

このモードでは、SSL セッションが再折衝される (その結果、新規の SSL セッション ID が作成される) ため、SSL セッション ID を使用してユーザー・セッション・アクティビティーを追跡することはできません。既存 SSL セッションのすべての接続はクローズされます。

ユーザーが証明書認証が必要なリソースを要求した時点のユーザーの認証状況に基づいて、2 つのシナリオで遅延証明書認証が使用されます。どちらのシナリオで

も、証明書を使用して認証を行う必要性が確立されるまで、ユーザーは、WebSEAL サーバーとの交換を何回でも行うことができます。

2 つのシナリオには以下が含まれます。

- **ユーザーが非認証である**

このシナリオでは、ユーザーが、認証を必要とするリソースにアクセスしようとしなければ、ユーザーは非認証状態のままになっています。ユーザーが ACL のために認証が必要なリソースにアクセスしようすると、WebSEAL から証明書ログイン・フォームが表示され、ユーザーは (このフォーム上のボタンをクリックして) 証明書の転送を開始することができます。

WebSEAL は、非認証ユーザーのエントリーをセッション・キャッシュの中に保存しますが、GSKit にある新しい SSL ID を入手します。古い SSL セッション ID は廃棄されます。ユーザーが正常に認証されると、WebSEAL は、セッション・キャッシュ・データにある古い非認証ユーザーの資格情報を、新しいユーザー資格情報で置き換えます。これでユーザーは認証され、(ACL のために) 認証が必要なリソースへのアクセスを要求できます。

- **ユーザーは、別の認証方式を使用して既に認証されている**

このシナリオ (認証強度ポリシーまたはステップアップ認証と呼ばれます) では、ユーザーは、以前の WebSEAL との交換で Security Access Manager に対する認証が必要でした。前の認証は、例えば、フォーム認証などの別の認証方式で行われました。

その後、ユーザーは、リソースにアクセスするにはクライアント・サイド証明書認証が必要な保護オブジェクト・ポリシー (POP) によって保護されているリソースにアクセスしようとします。WebSEAL は、現行の WebSEAL 認証強度ポリシー構成を調べて、使用可能になっている認証方式のランキングを判別します。(認証強度ポリシーは、認証方式を、弱い強度から強い強度の順の階層にランク付けしています。)

証明書認証が、ユーザーの現行認証方式より上位にランク付けされていると、WebSEAL は、証明書ログイン・ボタンを含むステップアップ・ログイン・フォームをユーザーに提供します。ユーザーはボタンをクリックして、証明書の交換を開始できます。ユーザーが証明書を使用して正常に認証されると、現行セッション中は、ユーザーの認証強度レベルが上がります。

WebSEAL は、ユーザーのエントリーをセッション・キャッシュの中に保存していますが、GSKit にある新しい SSL セッション ID を入手します。古い SSL セッション ID は廃棄されます。WebSEAL は、古いユーザー資格情報 (ユーザーの前の認証方式に基づいている) を新規ユーザー資格情報に置き換えます。

認証強度ポリシーを使用することにより、ユーザーは、セッション中に、異なる認証レベルの間を移動できます。証明書認証は、保護オブジェクト・リソースにアクセスするために、ユーザーが認証レベルを上げる (ステップアップする) 必要があるときに使用できる認証レベルの 1 つです。

ユーザーが証明書認証レベルに移動できるようにするには、アドミニストレータ
ーは、WebSEAL 構成ファイルを変更して、証明書認証を、サポートされる認証
強度のリストに組み込む必要があります。

認証強度ポリシーの構成手順については、200 ページの『認証強度ポリシー（ス
テップアップ）』を参照してください。

証明書認証の構成タスクの要約

すべての証明書認証モードは、構成タスクの共通セットを共有しています。遅延証
明書認証モードには、追加タスクが必要です。

サポートされているすべてのモードでクライアント・サイド証明書認証を使用可能
にするには、以下のタスクを実行します。

1. 『証明書認証の使用可能化』
2. 168 ページの『証明書認証メカニズムの構成』
3. 170 ページの『証明書ログイン・エラー・ページ』

遅延証明書認証モードを使用可能にするときは、以下の追加タスクを実行します。

1. 171 ページの『証明書ログイン・フォーム』
2. 171 ページの『セッション・トラッキング用の SSL セッション ID の使用不可
化』
3. 172 ページの『証明書 SSL ID キャッシュの使用可能化と構成』
4. 172 ページの『証明書 SSL ID キャッシュのタイムアウトの設定』
5. 173 ページの『誤ったプロトコル用のエラー・ページ』

注: 新しい構成設定値を活動化するためには、WebSEAL サーバーを停止して再始動
する必要があります。

クライアント・サイド証明書認証を使用不可にする（構成解除する）には、以下のタ
スクを実行します。

- 173 ページの『証明書認証の使用不可化』
- 174 ページの『証明書 SSL ID キャッシュの使用不可化』

証明書認証に関する技術上の注意点

- 174 ページの『証明書認証に関する技術上の注意点』

証明書認証用の WebSEAL 構成ファイルの設定値は、「*IBM Security Web Gateway
Appliance: Web* リバース・プロキシ・スタンザ・リファレンス」に要約されてい
ます。

証明書認証の使用可能化

このタスクについて

デフォルトでは、証明書認証は使用不可になっています。証明書認証を使用可能に
するには、次のようにします。

手順

WebSEAL 構成ファイルを編集します。**[certificate]** スタンザで、クライアント・サイド証明書認証要求の処理方法を WebSEAL に指示する **accept-client-certs** スタンザ・エントリーに値を指定します。次の表に、有効な値を示します。

表 20. 証明書認証の構成

構成	説明
accept-client-certs = optional	クライアントは、オプションで、証明書ベースの認証を使用できます。 WebSEAL は、クライアントに、X.509 証明書の提供を求めます。ユーザーが証明書を提供している場合、証明書ベースの認証が使用されます。
accept-client-certs = required	クライアントは、証明書ベースの認証を使用することが必須です。 WebSEAL は、クライアントに、X.509 証明書の提供を求めます。ユーザーが証明書を提示しない場合、WebSEAL は接続を許可しません。
accept-client-certs = prompt_as_needed	ユーザーは、セッション開始時に、証明書で認証を受けることが必須ではありません。ユーザーは、あとで、証明書認証を開始できます。 この設定は、 <i>遅延証明書認証</i> モードを使用可能にします。

例えば、クライアント・サイド証明書を求めるプロンプトをユーザーに出すのを、証明書認証が必要なリソースにユーザーが出会ったときだけにしたい場合は、次のように入力します。

```
[certificate]
accept-client-certs = prompt_as_needed
```

この設定は、証明書認証の認証強度ポリシー (ステップアップ) をインプリメントするときに使用されます。

証明書認証メカニズムの構成

外部認証インターフェース (EAI) プロトコルを使用して、WebSEAL に代わって認証を処理するように junction Web アプリケーションを構成できます。

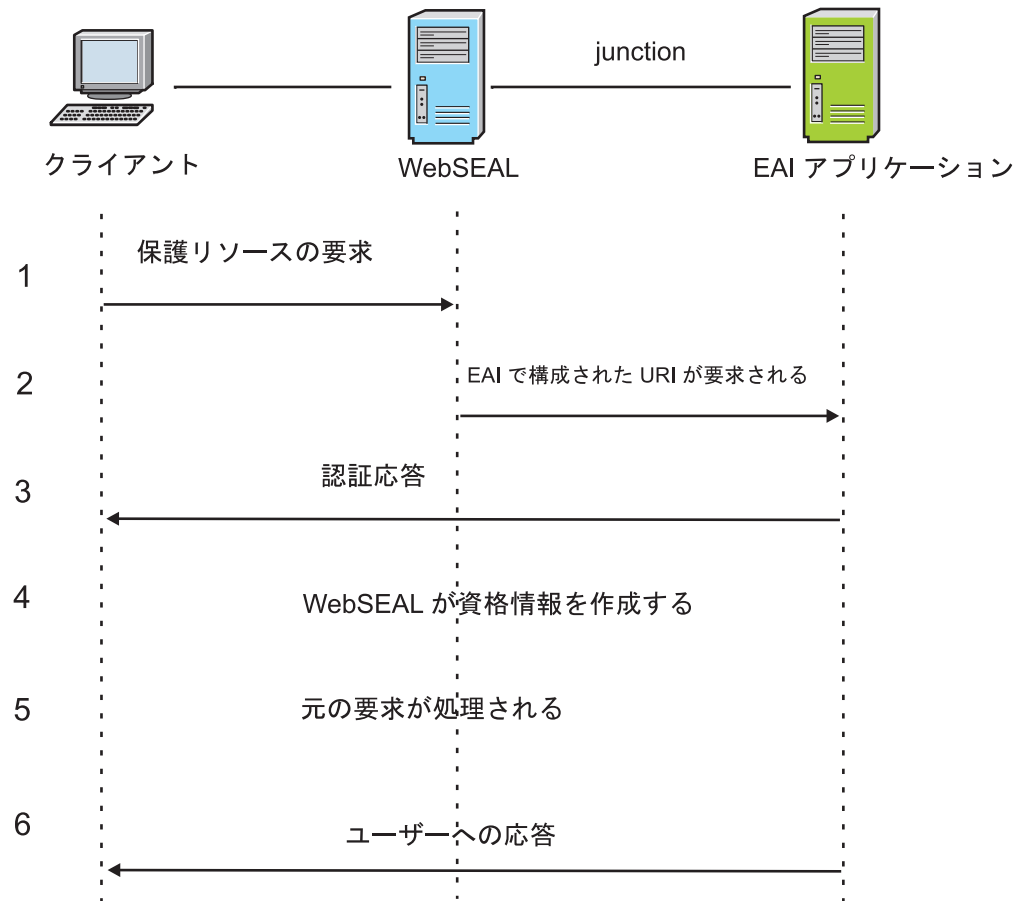
- 『EAI 証明書認証』。
- 169 ページの『EAI 証明書認証の構成』。

注: EAI の詳細については、263 ページの『第 13 章 外部認証インターフェース』を参照してください。

EAI 証明書認証

外部アプリケーションを使用してクライアント証明書を認証することもできます。この外部アプリケーションは EAI プロトコルを使用して、ユーザー資格情報を生成するときに WebSEAL が使用する認証データを提供します。

以下の図は、認証操作のプロセス・フローを中心に説明したものです。



1. WebSEAL によって保護されているリソースへの要求が実行されます。
WebSEAL は **accept-client-certs** 構成エントリーの設定に基づき、クライアント証明書を折衝します。
2. WebSEAL はサブリクエストを作成します。作成されたサブリクエストは、次に構成済み EAI アプリケーションに送信されます。EAI アプリケーションの URI は **eai-uri** 構成エントリーを使用して構成されます。
3. EAI アプリケーションは (クライアント証明書データに基づいて) ユーザーを認証し、WebSEAL がユーザーの資格情報を正しく作成できるよう、必要な EAI ヘッダーを提供します。認証が失敗した場合、EAI は認証データを戻しません。これは、WebSEAL に対して、認証エラーが発生したことを示します。この時点で WebSEAL は認証エラー・ページを生成し、これをクライアントに戻します。
4. WebSEAL は認証データを使用して、ユーザーの資格情報を作成します。
5. ユーザーが正しく認証されたため、WebSEAL は元の要求の処理を続行します。
6. 元の要求への応答がクライアントに渡されます。

EAI 証明書認証の構成

このタスクについて

外部認証メカニズムを構成するには、以下のステップを実行します。

手順

1. 証明書認証が使用可能になっていることを確認してください。167 ページの『証明書認証の使用可能化』を参照してください。
2. **[certificate]** スタンザの中に、認証を実行するために呼び出される URI を **eai-uri** スタンザ・エントリーの値として指定します。これは、WebSEAL サーバーのルート Web スペースに対して相対的な URL である必要があります。
「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の構成エントリーを参照してください。
3. **[certificate]** スタンザの中に、EAI アプリケーションに渡されるクライアント証明書データ・エレメントを **eai-data** スタンザ・エントリーの値として指定します。これは **eai-data = data: header_name** の形式である必要があります。複数の **eai-data** 構成エントリーを含めることによって、複数のクライアント証明書データを EAI アプリケーションに渡すことができます。受け入れ可能なデータ値などの詳細については、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の構成エントリーを参照してください。

次のタスク

EAI プロトコルについて詳しくは、以下のセクションを参照してください。

1. 269 ページの『認証データの HTTP ヘッダー名』
2. 270 ページの『特殊 HTTP ヘッダーからの認証データの抽出』
3. 271 ページの『資格情報の生成方法』
4. 273 ページの『外部認証アプリケーションの作成方法』

注: クライアント証明書を認証するために外部アプリケーションを使用するとき、複数ステップ認証は許可されません。また、外部認証アプリケーションは非認証ユーザーに対して利用可能である必要はありません。

5. 275 ページの『外部認証インターフェース HTTP ヘッダーのリファレンス』
6. 277 ページの『外部認証インターフェースによる認証後リダイレクト』
7. 277 ページの『外部認証インターフェースによるセッション処理』
8. 277 ページの『外部認証インターフェースによる認証強度レベル』
9. 278 ページの『外部認証インターフェースによる再認証』
10. 279 ページの『クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定』
11. 281 ページの『クライアント固有のセッション・キャッシュ・エントリー非アクティブ・タイムアウト値の設定』

証明書ログイン・エラー・ページ

アドミニストレータは、デフォルトのエラー・ページを使用する、エラー・メッセージをカスタマイズする、あるいは、まったく別のカスタマイズされたエラー・ページを指定するを選択できます。典型的には、アドミニストレータは、デフォルト・ページを使用するが、エラー・メッセージの内容をカスタマイズします。

WebSEAL は、エラー・メッセージが入っているデフォルトの HTML ページを戻します。ユーザーがクライアント・サイド証明書認証を使用して正常に認証されな

った場合に、このページが表示されます。具体的には、このエラー・ページは証明書が有効であるが、Security Access Manager ユーザーに一致しないときに戻されます。

このページは、失効した証明書が提示されたときは戻されません。証明書の失効は、SSL によって処理されます。失効した証明書が提示された場合、SSL 接続がすぐにクローズされ、(WebSEAL エラー・ページではなく) ブラウザーのエラー・ページが表示されます。

新規の HTML エラー・ページを作成することを選択したアドミニストレーターは、WebSEAL 構成ファイルを編集して、新規ページのロケーションを示す必要があります。

デフォルトの WebSEAL 構成ファイルのエントリーは次のようになっています。

```
[acct-mgt]
cert-failure = certfailure.html
```

証明書ログイン・フォーム

WebSEAL には、ログイン・フォームが含まれる HTML ページが用意されており、遅延証明書認証が必要となった場合に、ユーザーに対して表示されます。

アドミニストレーターは、デフォルトのログイン・フォームを使用する、ログイン・フォームをカスタマイズする、あるいは、まったく別のカスタマイズされたログイン・ページを指定するを選択できます。典型的には、アドミニストレーターは、デフォルト・ファイルを使用するが、フォームの内容をカスタマイズします。

新規の HTML ファイルを作成することを選択したアドミニストレーターは、WebSEAL 構成ファイルを編集して、新規ファイルのロケーションを示す必要があります。

デフォルトの WebSEAL 構成ファイルのエントリーは次のようになっています。

```
[acct-mgt]
certificate-login = certlogin.html
```

セッション・トラッキング用の SSL セッション ID の使用不可化

このタスクについて

この構成ステップは、遅延証明書認証が使用可能になっている場合にのみ適用されます。

手順

セッション状態をトラッキングする SSL セッション ID の使用を使用不可にします。WebSEAL 構成ファイル内の **ssl-id-sessions** スタンザ・エントリーの値が、デフォルトの「no」であることを確認します。

```
[session]
ssl-id-sessions = no
```

注: この場合、SSL ID は、ユーザー・セッションを維持するために使用できません。これは、証明書を提示するようにユーザーにプロンプトが出されると、ユーザーの SSL ID が変更されるためです。ssl-id-sessions が「yes」に設定されている場合、WebSEAL は開始時にエラー・メッセージを出してシャットダウンします。

証明書 SSL ID キャッシュの使用可能化と構成

このタスクについて

この構成ステップは、遅延証明書認証が使用可能になっている場合にのみ適用されます。

キャッシュを構成するには、次のステップを実行します。

手順

1. 証明書認証が使用可能になっていることを確認してください。

167 ページの『証明書認証の使用可能化』を参照してください。

2. キャッシュに入れることができるエントリーの最大数を指定します。WebSEAL 構成ファイルを編集します。[certificate] スタンザで、**cert-cache-max-entries** に値を割り当てます。以下に例を示します。

```
[certificate]
cert-cache-max-entries = 1024
```

この値は、同時証明書認証の最大数に対応します。デフォルト値は、SSL ID キャッシュに入るエントリーのデフォルト数の 4 分の 1 です。(ほとんどの SSL セッションは、証明書ログインを必要としません。あるいは、セッションに一度だけ証明書認証が必要です。) SSL ID キャッシュに入れるエントリーの数を、次のように、[ssl] スタンザに設定します。例:

```
[ssl]
ssl-max-entries = 4096
```

したがって、**cert-cache-max-entries** のデフォルト値は 1024 になります。これは、**ssl-max-entries** のデフォルト値である 4096 の 4 分の 1 です。

注: WebSEAL に対するほとんどのユーザー要求は SSL 接続を介して行われます。SSL 接続を介した、証明書がないすべての要求は、キャッシュを検査する必要があります。キャッシュ・サイズをより小さくして保持すると、パフォーマンスを大幅に向上させることができます。

証明書 SSL ID キャッシュのタイムアウトの設定

このタスクについて

この構成ステップは、遅延証明書認証が使用可能になっている場合にのみ適用されます。

以下のステップを実行してください。

手順

1. 証明書認証が使用可能になっていることを確認してください。
167 ページの『証明書認証の使用可能化』を参照してください。
2. WebSEAL 構成ファイルを編集します。**[certificate]** スタンザで、**cert-cache-timeout** の値を、必要に応じて調整します。以下に例を示します。

```
[certificate]
cert-cache-timeout = 120
```

この値は、エントリーがキャッシュ内に存続できる最大存続時間 (秒数) です。変更する必要がない限り、デフォルト値を使用してください。デフォルト値を変更する、考えられる理由には次のものがあります。

- メモリーに制限があるシステムの場合、有効期限を短くする必要があります。
- ユーザーが証明書転送を開始するときと、ユーザーが実際に証明書をサブミットするときとの間に大きな時間のずれがある場合は、存続時間値を増やす必要があります。
- 証明書認証の処理がない場合は、値を小さくしたほうが、キャッシュが早く空になります。キャッシュをクリーニングすることによって、システム・メモリーが解放されます。

誤ったプロトコル用のエラー・ページ

この構成ステップは、遅延証明書認証が使用可能になっている場合にのみ適用されます。

WebSEAL には、エラー・メッセージが入っているデフォルトの HTML ページが用意されていて、認証済みユーザーが、認証強度レベルを証明書認証に HTTP セッションから上げようとする、このページが表示されます。認証レベルを証明書認証に上げようとするユーザーは、HTTPS プロトコルを使用する必要があります。

アドミニストレーターは、デフォルトのエラー・ページを使用する、エラー・メッセージをカスタマイズする、あるいは、まったく別のカスタマイズされたエラー・ページを指定するを選択できます。典型的には、アドミニストレーターは、デフォルト・ページを使用するが、エラー・メッセージの内容をカスタマイズします。

新規の HTML エラー・ページを作成することを選択したアドミニストレーターは、WebSEAL 構成ファイルを編集して、新規ページのロケーションを示す必要があります。

デフォルトの WebSEAL 構成ファイルのエントリーは次のようになっています。

```
[acct-mgt]
cert-stepup-http = certstepuphttp.html
```

証明書認証の使用不可化

このタスクについて

証明書認証を使用不可にするには、次のようにします。

手順

1. WebSEAL サーバーを停止します。
2. WebSEAL 構成ファイルを編集します。**[certificate]** スタンザで、次のような *key = value* ペアを指定します。

```
[certificate]  
accept-client-certs = never
```

3. WebSEAL サーバーを再始動します。

証明書 SSL ID キャッシュの使用不可化

このタスクについて

証明書 SSL ID キャッシュは、遅延証明書認証、または証明書認証への認証強度ステップアップでのみ使用されます。

キャッシュは、証明書認証の構成設定に基づいて自動的に使用不可にされます。

手順

キャッシュが使用不可になっていることを確認するには、**[certificate]** スタンザの中の **accept-client-certs** の値を調べます。値が以下のいずれかになっていることを確認します。

- required
- optional
- never

値が、**prompt_as_needed** になっていない ことを確認します。

証明書認証に関する技術上の注意点

すべての証明書構成で、クライアント・サイド証明書を提示できるのはブラウザー・セッションごとに 1 度だけです。

クライアント・サイド証明書の交換が失敗した場合、WebSEAL では、クライアント・サイド証明書を再度提示する前にブラウザー・セッションを再始動する必要があります。

Kerberos 認証

IBM Security Web Gateway Appliance は、Windows デスクトップ・シングル・サインオンを実行できるようにするために Windows クライアントで使用する Kerberos 認証を内部でサポートしません。ただし、アプライアンスの代わりに Kerberos 認証を処理するように、Junction Web サーバーを構成することができます。

Junction Web サーバーは、外部認証インターフェース (EAI) アプリケーションとして機能し、Kerberos 認証を完了します。この Web サーバー (*Kerberos* オーセンティケーター と呼ばれる) は、その後認証されたユーザー ID を EAI ヘッダー内のアプライアンスに戻します。EAI の詳細については、263 ページの『第 13 章 外部認証インターフェース』を参照してください。

この章で扱うトピックは以下のとおりです。

- 『Kerberos 認証の構成』.
- 176 ページの『制限』.

Kerberos 認証の構成

Windows デスクトップのシングル・サインオンを実現するには、IBM Security Web Gateway Appliance に代わってクライアントを認証するように Kerberos オーセンティケーターを構成する必要があります。

このタスクについて

実際の認証を完了して認証済みの ID をアプライアンスに返すように junction Web サーバーを構成することができます。

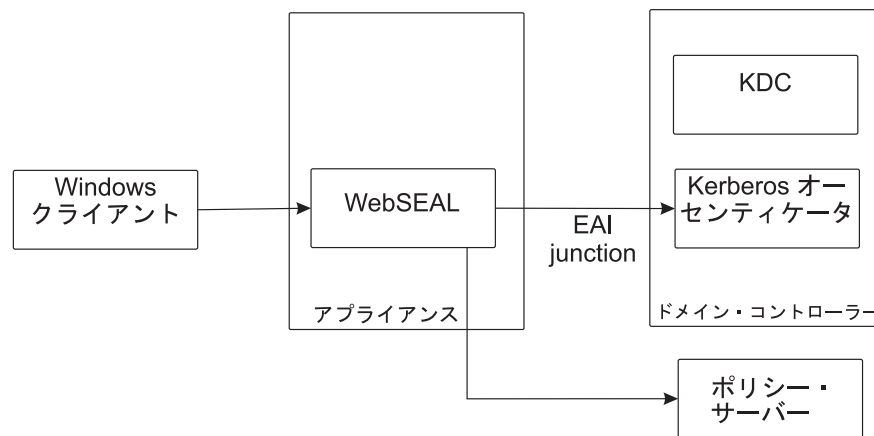


図 14. 外部 Kerberos 認証

アプライアンスに代わって認証を実行するように外部 Kerberos オーセンティケーターを構成するには、以下のステップを実行します。ステップごとに例を示しています。全体として、この例では、Windows デスクトップのシングル・サインオンをサポートする 1 つの構成例について説明しています。

手順

1. ポリシー・サーバーをインストールし、ユーザー・レジストリーを構成します。例として Active Directory を採用します。
2. Kerberos 認証をサポートする Web サーバーを構成します。この Web サーバーが Kerberos オーセンティケーターになります。例えば、ドメイン・コントローラーに WebSEAL をインストールして Kerberos 認証を構成します。

この WebSEAL 構成について詳しくは、「*IBM Security Access Manager for Web: WebSEAL 管理ガイド*」で「Windows デスクトップ・シングル・サインオン」を検索してください。

3. Kerberos オーセンティケーターで外部認証インターフェース (EAI) アプリケーションを構成します。例えば、EAI として機能する単純なコモン・ゲートウェ

イ・インターフェース (CGI) を作成します。この CGI は EAI 応答を作成し、**am-eai-user-id** ヘッダー・フィールドを認証済みユーザーの名前に設定します。

この時点で、Kerberos オーセンティケーターの構成を検証できます。Windows クライアントをドメインに追加します。このクライアントから WebSEAL サーバーにアクセスしたときに Windows デスクトップのシングル・サインオンが行われることを確認します。

ドメイン・コントローラーにネットワーク・プロトコル・アナライザー (Wireshark など) をインストールして、ネットワーク・トラフィックをモニターおよび検証することができます。

4. 認証に外部 Kerberos オーセンティケーターを使用するようにアプライアンスの WebSEAL を構成します。例えば、以下のステップを実行します。
 - a. Kerberos オーセンティケーターに対する junction を作成します。
 - b. CGI スクリプトを EAI アプリケーションとして構成します。
 - c. **strip-www-authenticate-headers** 構成エントリーを **no** に設定します。

strip-www-authenticate-headers 構成エントリーが **yes** に設定されていると、WebSEAL が junction サーバーの応答から **Negotiate www-authenticate** ヘッダーおよび **NTLM www-authenticate** ヘッダーを削除してしまいます。そのため、この値を **no** に設定して、これらの **www-authenticate** ヘッダーが junction サーバーの応答に保持されるようにする必要があります。

この構成エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

この時点で、アプライアンスで Windows デスクトップのシングル・サインオンが使用可能かどうかを確認することができます。

Windows クライアントからアプライアンスの WebSEAL サーバーを経由して EAI アプリケーションに要求を送信します。シングル・サインオンが行われます。つまり、そのユーザーは、認証済みユーザーとしてアプライアンスの WebSEAL サーバーにアクセスできます。ここでも、ネットワーク・プロトコル・アナライザーを使用してネットワーク・トラフィックをモニターおよび検証することができます。

制限

外部の Kerberos オーセンティケーターを使用する場合、アプライアンスは Kerberos 認証のみをサポートすることができます。NTLM 認証をサポートすることはできません。

Windows NTLM インプリメンテーションでは、認証プロセスの複数の段階で同じ接続を使用する必要があります。WebSEAL は、認証プロセス全体で使用できるように同じ接続を常に提供することはできません。

したがって、NTLM 認証のみをサポートするサーバーを Kerberos オーセンティケーターとして使用することはできません。Kerberos 認証をサポートするサーバーを Kerberos オーセンティケーターとして使用する必要があります。

注: Microsoft Internet Information Services (IIS) では、NTLM 認証をデフォルトで使用しています。

LTPA 認証

Security Access Manager では、クライアントから受け取った LTPA Cookie を使用した認証をサポートしています。このセクションは、以下のトピックで構成されています。

- 『LTPA 認証の概要』
- 178 ページの 『LTPA 認証の使用可能化』
- 178 ページの 『鍵ファイル情報』
- 179 ページの 『クライアントの Cookie 名の指定』
- 179 ページの 『Junction の Cookie 名の指定』
- 180 ページの 『LTPA トークンの存続時間の制御』
- 180 ページの 『LTPA 認証の使用不可化』

LTPA 認証の概要

Cookie ベースの LTPA (Lightweight Third-Party Authentication) 認証メカニズムのサポートは、様々な IBM サーバーで提供されています。これらのサーバーには、WebSphere および DataPower® などがあります。これらの 1 つ以上のサーバーにシングル・サインオン・ソリューションを実現するには、LTPA 認証をサポートするように WebSEAL を構成します。

WebSphere または DataPower 用の認証トークンとしての役割を果たすこの LTPA Cookie には、ユーザー ID、鍵およびトークン・データ、バッファー長、および有効期限に関する情報が含まれています。これらの情報は、WebSEAL と他の LTPA 対応サーバーとの間で共用される、パスワード保護された秘密鍵を使用して暗号化されます。

WebSEAL で保護されているリソースを非認証ユーザーが要求すると、LTPA Cookie が使用可能かどうかはまず判別されます。LTPA Cookie が使用可能な場合、Cookie の内容が検証されます。Cookie が正常な場合、Cookie に含まれているユーザー名および有効期限時刻に基づいて新規セッションが作成されます。LTPA Cookie が使用可能でない場合、WebSEAL は構成済みの他の認証メカニズムを使用してユーザーの認証を続行します。認証操作が完了すると、新規の LTPA Cookie が HTTP 応答に挿入されてクライアントに渡され、LTPA に対応した他の認証サーバーで使用されます。

WebSEAL は LTPA バージョン 2 (LtpaToken2) の Cookie のみサポートします。LtpaToken2 は、以前のバージョンのトークンよりも強い暗号化を含み、複数の属性をトークンに追加することができます。このトークンは、認証 ID および追加情報 (元のログイン・サーバーと接続するために使用される属性や、固有性の判別で ID 以外も考慮に入れる場合に検索に使用する固有のキャッシュ鍵など) を含みます。

LtpaToken2 は、WebSphere Application Server バージョン 5.1.0.2 (z/OS の場合) およびバージョン 5.1.1 (分散の場合) 以降用に生成されます。

LTPA 認証の使用可能化

ltpa-auth スタンザ・エントリーは、WebSEAL 構成ファイルの [ltpa] スタンザにあります。これは、LTPA 認証方式を使用可能または使用不可にします。

このタスクについて

デフォルトでは、LTPA 認証は使用不可になっています。LTPA 認証を構成するには、以下のステップを実行します。

手順

1. WebSEAL サーバーを停止します。
2. WebSEAL 構成ファイルを編集します。[ltpa] スタンザで、ご使用のネットワーク環境でサポートするプロトコルを指定します。サポートするプロトコルを、次の表に示します。

表 21. LTPA 認証の構成

サポートするプロトコル	構成ファイル・エントリー
HTTP	ltpa-auth = http
HTTPS	ltpa-auth = https
HTTP および HTTPS	ltpa-auth = both
HTTP ヘッダーおよび Cookie 認証の使用不可化 (デフォルト)	ltpa-auth = none

例えば、両方のプロトコルをサポートするには [ltpa] ltpa-auth = both とします。

3. [ltpa] スタンザに含まれているエントリーをカスタマイズします。
4. WebSEAL サーバーを再始動します。

鍵ファイル情報

LTPA トークンは、パスワード保護された秘密鍵によって暗号化されます。鍵そのものは WebSphere によって生成され、鍵ファイルに格納されます。この鍵ファイルは平文の鍵によってパスワード保護されます。

WebSEAL によって使用される鍵ファイルの名前は、[ltpa] スタンザ内の **keyfile** 構成エントリーで定義されます。ファイルには、WebSEAL バイナリーを実行するユーザーに対する読み取りアクセスが許可されている必要があります。

[ltpa] スタンザ内の **keyfile-password** 構成エントリーは、鍵ファイルの保護に使用されるパスワードを定義します。パスワードが重要である場合、代わりに WebSEAL の難読化されたデータベース内の対応する構成エントリーに保存することもできます。

このパスワードを管理するには、ローカル管理インターフェース (LMI) を使用します。

クライアントの Cookie 名の指定

WebSEAL がクライアントに送出する LTPA トークンを含む Cookie の名前を構成することができます。

デフォルトでは、WebSEAL は **Ltpatoken2** という名前の Cookie を使用して LTPA トークンを含めます。WebSphere および DataPower はどちらも、デフォルトではこの名前を予期します。LTPA トークンが含まれている Cookie の名前をカスタマイズするには、**[ltpa]** スタンザ内の **cookie-name** 構成エントリーの値を変更します。

例:

```
[ltpa] cookie-name = Ltpatoken2
```

Junction の Cookie 名の指定

Junction Web サーバーの LTPA トークンを含む Cookie の名前を構成することができます。

WebSphere Application Server および WebSEAL は、以下に示す同じデフォルト値を LTPA Cookie 名に使用します。

- LTPA トークン用の **LtpaToken**。
- LTPA バージョン 2 のトークン用の **LtpaToken2**。

WebSEAL からバックエンド上の Junction を介して送信される LTPA Cookie の名前を構成するには、**[ltpa]** スタンザ内の **jct-ltpa-cookie-name** エントリーを使用します。このアイテムは、グローバルに構成することも、Junction 単位で構成することもできます。

すべての Junction を介して使用する WebSEAL の Cookie 名を設定するには、**[ltpa]** スタンザ内のエントリーを構成します。例:

```
[ltpa]
jct-ltpa-cookie-name = myGlobalLTPAcookie
```

特定の junction に固有の Cookie 名を設定するには、**[ltpa:/jct]** スタンザ内のエントリーを構成します。

説明:

jct バックエンド・サーバーへの Junction の名前。

例:

```
[ltpa:/jct]
jct-ltpa-cookie-name = myLTPACookieForJct
```

WebSEAL でカスタム LTPA Cookie 名を使用する場合は、シングル・サインオンを実現するために、WebSphere でも同じ Cookie 名を構成する必要があります。

jct-ltpa-cookie-name エントリーを構成しない場合、WebSEAL はデフォルトの Cookie 名を使用します。

LTPA トークンの存続時間の制御

LTPA Cookie の存続時間は、デフォルトで、トークンの作成に使用されたセッションの存続時間に設定されます。よりきめ細かい方法で設定するには、**[ltpa]** スタンザで **update-cookie** 構成エントリーを変更することができます。このエントリーは、新しい存続時間タイムアウトを使用してトークンの更新頻度を制御します。

注: この構成エントリーは、WebSEAL がクライアントに対して発行する LTPA Cookie に影響します。**[ltpa]** スタンザ内の **cookie-name** 構成エントリーによって指定されるのは、その Cookie の存続時間です。

- デフォルト値の **-1** は、トークンが更新されることはなく、トークンの存続時間が最大セッション存続時間と等しいことを示します。
- 値 **0** は、トークンの存続時間が要求ごとに更新されることを示します。この構成は非アクティブ・タイムアウトと同等の機能をトークンに提供します。
- 正数は、トークンの更新から更新までの経過時間 (秒) を示します。この構成は、精度があまり高くなく、非アクティブ・タイムアウトと同等の機能をトークンに提供します。

ご使用の環境でこの構成エントリーを有効にするかどうかは、慎重に考慮してください。LTPA トークンを作成して HTTP 応答に追加することのコストが、トークンの非アクティブ・タイムアウトを実現することによって得られる利益を上回る可能性もあります。

LTPA 認証の使用不可化

このタスクについて

LTPA 認証を使用不可にするには、以下のステップを実行します。

手順

1. WebSEAL サーバーを停止します。
2. WebSEAL 構成ファイルを編集し、**ltpa-auth** を **none** に設定します。

```
[ltpa] ltpa-auth = none
```
3. WebSEAL サーバーを再始動します。

タスクの結果

注: デフォルトでは、LTPA 認証は使用不可になっています。

第 9 章 拡張認証方式

この章では、拡張 WebSEAL 認証の機能について説明します。

この章で扱うトピックは以下のとおりです。

- 『多重プロキシ・エージェント (MPA)』
- 185 ページの『ユーザー切り替え認証』
- 278 ページの『外部認証インターフェースによる再認証』
- 200 ページの『認証強度ポリシー (ステップアップ)』
- 213 ページの『外部認証インターフェース』
- 213 ページの『クライアント証明書ユーザー・マッピング』

多重プロキシ・エージェント (MPA)

このセクションは、以下のトピックで構成されています。

- 『多重プロキシ・エージェント (MPA) の概要』
- 182 ページの『有効なセッション・データ・タイプと認証方式』
- 183 ページの『MPA および複数クライアントの認証プロセス・フロー』
- 184 ページの『MPA 認証の使用可能および使用不可』
- 184 ページの『MPA 用ユーザー・アカウントの作成』
- 184 ページの『webseal-mpa-servers グループへの MPA アカウントの追加』
- 184 ページの『MPA 認証に関する制限』

多重プロキシ・エージェント (MPA) の概要

Security Access Manager は、多重方式プロキシ・エージェント (MPA) を使用するネットワークを保護するためのソリューションを提供します。

標準プロキシ・エージェント (SPA) は、SSL または HTTP におけるクライアントとオリジン・サーバーの間のクライアント別セッションをサポートするゲートウェイです。WebSEAL は、これらのクライアント別セッションに通常の SSL または HTTP 認証を適用することができます。

多重方式プロキシ・エージェント (MPA) は、マルチクライアント・アクセスに対応するゲートウェイです。これらのゲートウェイは、クライアントが Wireless Access Protocol (WAP) を使用してアクセスする際には、WAP ゲートウェイとも呼ばれます。ゲートウェイは、オリジン・サーバーへの単一の認証済みチャネルを確立し、このチャネルをすべてのクライアント要求および応答用の「トンネル」として使用します。

WebSEAL から見ると、このチャネルを経由した情報は、最初は、1 つのクライアントからの複数の要求として認識されます。WebSEAL は、MPA サーバーの認証と

各個別クライアントの追加認証とを区別する必要があります。

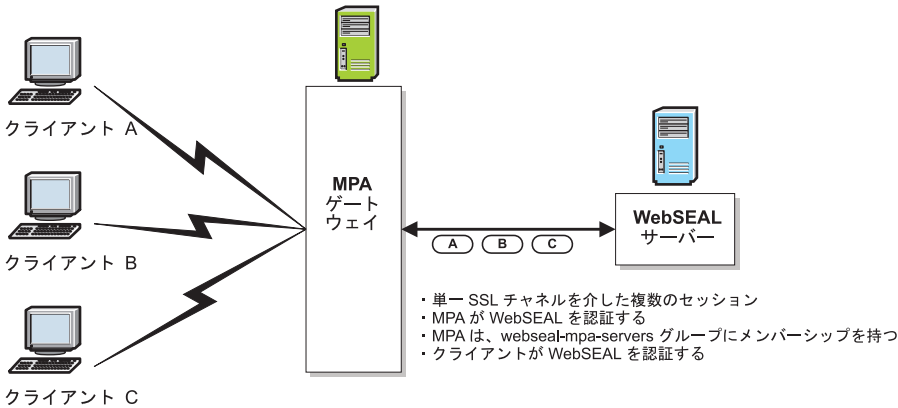


図 15. MPA ゲートウェイを介した通信

WebSEAL は、MPA 用の認証済みセッションを維持するので、同時に、各クライアントごとに別々のセッションを維持する必要があります。したがって、MPA 用に使用されるセッション・データと認証方式は、クライアントが使用する認証方式およびセッション・データとは異なる別個のものであることが必要です。

有効なセッション・データ・タイプと認証方式

セッションが多重プロキシ・エージェント (MPA) 用であるかクライアント用であるかに応じて、さまざまなセッション・データ・タイプおよび認証方式が有効です。

以下の表は、MPA とクライアントの有効なセッション・タイプを示しています。

有効なセッション・タイプ		
セッション・タイプ	MPA 対 WebSEAL	クライアント対 WebSEAL
SSL セッション ID	はい	無効
HTTP ヘッダー	はい	はい
IP アドレス	はい	無効
セッション Cookie	はい	はい

MPA が WebSEAL に対して使用するセッション・データ・タイプは、クライアントが WebSEAL に対して使用するセッション・データ・タイプとは別個のものであることが必要です。例えば、MPA がセッション・データ・タイプとしてセッション Cookie を使用している場合、クライアントは HTTP ヘッダー・セッション・データ・タイプを使用する必要があります。

- ・ クライアントは、セッション・データ・タイプとして SSL セッションを使用することはできません。
- ・ MPA サポートが使用可能にされている場合は、**ssl-id-sessions** の機能が変化します。通常、**ssl-id-sessions=yes** の場合は、HTTPS クライアントのセッション

を維持するために、SSL セッション ID のみが使用されます。MPA が SSL セッション ID を使用してセッションを維持し、クライアントが別の方式を使用してセッションを維持できるようにする必要があるときは、この制限はなくなります。329 ページの『有効なセッション鍵データ・タイプ』も参照してください。

以下の表は、MPA とクライアントの有効な認証方式を示しています。

有効な認証タイプ		
認証タイプ	MPA 対 WebSEAL	クライアント対 WebSEAL
基本認証	はい	はい
フォーム認証	はい	はい
証明書	はい	無効
外部認証インターフェース	はい	無効

MPA が WebSEAL に対して使用する認証方式は、クライアントが WebSEAL に対して使用する認証方式とは別個のものであることが必要です。例えば、MPA が基本認証を使用している場合、クライアントは、フォーム認証を使用する必要があります。

- クライアントは、証明書および外部認証インターフェース認証方式を使用することはできません。
- 通常、特定のトランスポートについてフォーム認証が使用可能な場合、そのトランスポートについては基本認証は自動的に使用不可にされます。MPA サポートが使用可能にされている場合は、この制限はなくなります。その結果、例えば、MPA はフォーム を使用したログインが、クライアントは同じトランスポートの基本認証を使用したログインができるようになります。

MPA および複数クライアントの認証プロセス・フロー

1. WebSEAL アドミニストレータは、以下の事前準備の構成を実行します。
 - 多重プロキシ・エージェント (MPA) のサポートを使用可能にします。
 - 特定の MPA ゲートウェイについて Security Access Manager アカウントを作成します。
 - この MPA アカウントを **webseal-mpa-servers** グループに追加します。
2. クライアントが MPA ゲートウェイに接続されます。
3. ゲートウェイが要求を HTTP 要求に変換します。
4. ゲートウェイによりクライアントの認証を行います。
5. ゲートウェイが、クライアント要求を持つ WebSEAL との接続を確立します。
6. MPA は WebSEAL に対して認証し (クライアントとは異なる方式を使用)、MPA 用の識別 (既に WebSEAL アカウントを持っている) が導き出されます。
7. WebSEAL が、**webseal-mpa-servers** グループの MPA のメンバーシップを検査します。
8. MPA について資格情報を作成し、それに対して、キャッシュ内に特定の MPA タイプとしてのフラグを立てます。

この MPA 資格情報は、今後の各クライアント要求に伴いますが、これらの要求の許可検査には使用されません。

9. ここで、WebSEAL は、要求の所有者をさらに識別する必要があります。

MPA は、ログイン・プロンプトの適正なルーティングを行うための、複数のクライアントを区別することができます。

10. クライアントは、MPA 用に使用される認証タイプとは異なる方式を使用して、ログインし、認証します。
11. WebSEAL は、クライアント認証データに基づいて資格情報を作成します。
12. 各クライアントが使用するセッション・データ・タイプは、MPA が使用するセッション・データ・タイプとは異なるものでなければなりません。
13. 許可サービスは、ユーザーの資格情報とオブジェクトの ACL 許可に基づいて、保護オブジェクトへのアクセスを許可あるいは拒否します。

MPA 認証の使用可能および使用不可

このタスクについて

MPA 認証を使用可能または使用不可にするには、WebSEAL 構成ファイルの `[mpa]` スタンザにある `mpa` スタンザ・エントリを使用します。

手順

- MPA 認証方式を使用可能にするには、「yes」を入力します。
- MPA 認証方式を使用不可にするには、「no」を入力します。

例

```
[mpa]
mpa = yes
```

MPA 用ユーザー・アカウントの作成

ユーザー・アカウントの作成については、「*IBM Security Access Manager for Web 管理ガイド*」を参照してください。

webseal-mpa-servers グループへの MPA アカウントの追加

グループの管理については、「*IBM Security Access Manager for Web 管理ガイド*」を参照してください。

MPA 認証に関する制限

- Security Access Manager がサポートしているのは、1 つの WebSEAL サーバーにつき 1 つの MPA だけです。
- ステップアップ認証構成では、MPA 認証はサポートされません。
- `use-same-session = yes` の場合は、MPA はサポートされません。

ユーザー切り替え認証

このセクションは、以下のトピックで構成されています。

- 『ユーザー切り替え機能の概要』
- 188 ページの『ユーザー切り替え認証の構成』
- 191 ページの『ユーザー切り替えの使用』

ユーザー切り替え機能の概要

WebSEAL のユーザー切り替え機能を使用すると、特定のアドミニストレーターが、Security Access Manager セキュア・ドメインのメンバーであるユーザーの ID を使用して操作することができます。ユーザーの ID を使用できる機能は、ヘルプ・デスク環境でアドミニストレーターが問題のトラブルシューティングと診断を行う際に役立ちます。また、リソースへのユーザーのアクセスをテストし、アプリケーション統合テストを行うときにも、ユーザー切り替えを使用できます。

ユーザー切り替えインプリメンテーションは、UNIX 環境における `su` コマンドに似ています。WebSEAL 環境で、アドミニストレーターはユーザーの資格情報を獲得し、その実ユーザーとまったく同じ能力を得て、リソースおよびバックエンド・アプリケーションと相互作用します。

アドミニストレーターは、特殊な HTML フォームを使用して、ユーザー切り替え情報を指定します。WebSEAL はこのフォームを処理し、特殊認証メカニズムを呼び出して、ユーザーのパスワードを要求せずに指定されたユーザーの資格情報を戻します。

ユーザー切り替えのプロセス・フローは以下の手順のとおりです。

1. アドミニストレーターは WebSEAL の認証を受けます。WebSEAL はアドミニストレーターのセッションを確立し、アドミニストレーターのエントリーを WebSEAL のセッション・キャッシュの中に作成します。

セッション・キャッシュ・エントリーには、キャッシュ・データ構造が入っています。このデータ構造には、アドミニストレーターの資格情報が保管されます。ユーザー切り替えのプロセス・フロー中に、キャッシュ・データが操作されます。

WebSEAL セッション・キャッシュについて詳しくは、287 ページの『WebSEAL セッション・キャッシュ構造』を参照してください。

2. アドミニストレーターは、事前構成済みのユーザー切り替え HTML フォームを要求し、フォームに記入します。フォーム上に、アドミニストレーターは、以下のことを指定します。
 - アドミニストレーターが代わりに実行するユーザーの ID と名前。
 - 宛先 URL。
 - 認証方式。

このアクションの結果として、POST 要求が `/pkmsu.form` に送られます。

WebSEAL で使用可能にする前に、ユーザー切り替え HTML フォームの内容を変更することができます。189 ページの『ユーザー切り替え HTML フォームの構成』を参照してください。

また、フォームの機能を拡張することもできます。190 ページの『追加入力フォームの設計』を参照してください。

注: `pkmsu.form` 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

3. WebSEAL は、以下の検査を実行して、ユーザー切り替え要求を認可するかどうかを決定します。

- a. WebSEAL は、Security Access Manager **su-admins** グループのメンバーシップを調べて、アドミニストレーターがユーザー切り替え機能を起動する許可を持っているかどうかを判別します。

ユーザー切り替え認証の使用を要求するアドミニストレーターは、**su-admins** グループのメンバーでなければなりません。このグループのメンバーシップを構成してからでなければ、ユーザー切り替えは使用できません。詳細については、188 ページの『ユーザー・アクセス権限の構成』を参照してください。

- b. WebSEAL は、Security Access Manager **su-admins > securitygroup > su-excluded** グループのメンバーシップを調べて、ユーザー切り替えフォームに指定されているユーザー ID が、これらのいずれかのグループのメンバーではないことを確認します。

これらのグループのいずれかに所属するユーザー ID は、ユーザー切り替え機能からアクセスできません。WebSEAL アドミニストレーターがこれらのグループのメンバーシップを構成してからでなければ、アドミニストレーターはユーザー切り替え機能を使用できません。構成手順とこれらのグループの詳細については、188 ページの『ユーザー・アクセス権限の構成』を参照してください。

4. ユーザー切り替え要求の許可を決定すると、WebSEAL は該当するユーザー切り替えモジュールを呼び出して特別なユーザー切り替え認証を実行します。

WebSEAL は、さまざまな認証メカニズムをサポートします。それぞれの認証メカニズムには、対応するユーザー切り替え認証メカニズムがあります。WebSEAL には、特別なユーザー切り替え機能を持つ組み込みモジュールが用意されています。

5. 指定されたユーザーの認証が成功すると、ユーザー切り替えモジュールは、ユーザー・パスワードの入力を求めずに、ユーザーの有効な資格情報を戻します。
6. WebSEAL は、WebSEAL セッション・キャッシュに入っている該当エントリーの内容を、以下のことを実行することによって操作します。
 - a. アドミニストレーターの WebSEAL セッション・キャッシュ・データを除去し、それを、別のロケーションに保管する。
 - b. アドミニストレーターのキャッシュ・データの代わりに、ユーザーの資格情報などの切り替え先ユーザーのキャッシュ・データを挿入する。

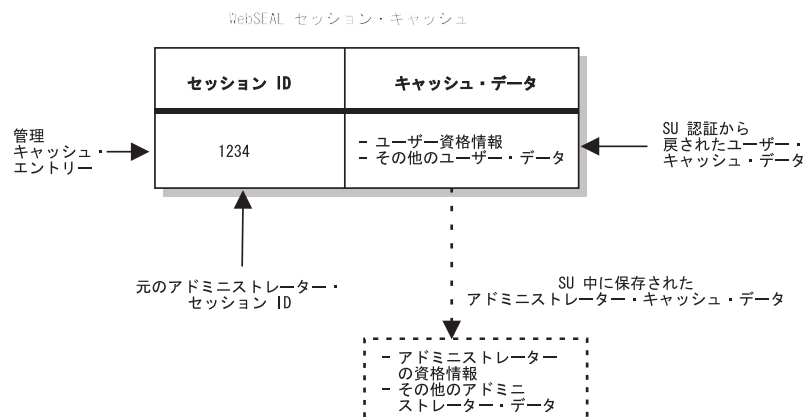


図 16. ユーザー切り替え中のアドミニストレーターおよびユーザー・キャッシュ・データのスイッチング

7. WebSEAL は、ユーザー切り替えフォームに指定されている宛先 URL へのリダイレクトを、ブラウザに送ります。

ユーザーの資格情報を使用して要求が正常に処理されます。

8. アドミニストレーターは、さらに他の要求を続行することができます。これらの要求についての許可決定は、すべて、このユーザーの資格情報に基づいて行われます。

ユーザー切り替え機能を使用するとき、アドミニストレーターは、他のアプリケーションとセッションを確立し、管理する必要があります。これらのセッションは、新しいユーザーの ID を使用して確立する必要があります。これを可能にするために、新しいユーザー資格情報には、新しいユーザー・セッション ID も入っています。このユーザー・セッション ID は、例えば、他の Web リソースにアクセスして使用するユーザー能力をトラブルシューティングするときに使用できます。

WebSEAL セッション・キャッシュについて詳しくは、295 ページの『WebSEAL セッション・キャッシュの構成』および 287 ページの『WebSEAL セッション・キャッシュ構造』を参照してください。

9. アドミニストレーターは、標準の Security Access Manager `/pkmslogout` ユーティリティを使用して、ユーザー切り替えセッションを終了します。ログアウトが正常に行われると、以下のことが行われます。
 - a. ユーザーのキャッシュ・データが削除される。
 - b. アドミニストレーターのオリジナルのキャッシュ・データ (および資格情報) が復元される。
 - c. アドミニストレーターは、ユーザー切り替えフォームを要求した元のページに戻る。

以後、許可サービスでは、すべての要求に対して、アドミニストレーターのオリジナルの資格情報が使用されます。

ユーザー切り替え認証の構成

WebSEAL アドミニストレーターがいくつかの構成ステップを実行してからでなければ、アドミニストレーターはユーザー切り替え機能を使用することはできません。ユーザー切り替えを構成するには、以下のそれぞれのセクションの手順を実行します。

1. 『ユーザー・アクセス権限の構成』
2. 189 ページの『ユーザー切り替え HTML フォームの構成』

このステップはオプションです。

3. 190 ページの『追加入力フォームの設計』

このステップはオプションです。

4. 191 ページの『WebSEAL の停止と再始動』

ユーザー・アクセス権限の構成

このタスクについて

WebSEAL のインストールの際に、WebSEAL 構成プロセスによって、ユーザー切り替え機能で使用されるいくつかのグループが自動的に作成されます。WebSEAL アドミニストレーターは、ユーザー切り替えの能力を、ユーザーをグループに追加することによってコントロールします。

ユーザー・アクセスを構成するには、以下のステップを実行します。

手順

1. ユーザーを **su-admins** グループに追加する。
ユーザー切り替え機能を使用するには、ユーザーは、**su-admins** という特殊な管理グループのメンバーでなければなりません。このグループは、WebSEAL サーバーのインストールの際に、デフォルトで自動的に作成されます。デフォルトでは、このグループに入っているユーザーはありません。WebSEAL アドミニストレーターが、ユーザーを手動でこのグループに追加しなければなりません。通常、管理ユーザーだけがこのグループに追加されます。
su-admins にメンバーシップを認可されたユーザーは、その他のほとんどのユーザー ID にユーザー切り替えができますが、**su-admins** グループのメンバーでもある他のユーザーの ID にはユーザー切り替えすることはできません。したがって、あるアドミニストレーターが、**su-admins** に追加されることによってユーザー切り替え特権を付与されると、そのアドミニストレーターのアカウントは、ユーザー切り替え特権を取得した他のユーザーによるアクセスから保護されます。
2. ユーザーを **su-excluded** グループに追加する。
このグループには、ユーザー切り替え機能を使用してアクセスしてはならない ID を持つユーザーの名前が入ります。WebSEAL のインストールの際に、WebSEAL 構成プロセスによって、このグループが自動的に作成されます。デフォルトでは、このグループに入っているユーザーはありません。WebSEAL アドミニストレーターは、通常、このグループに、管理グループ **su-admins** のメンバーではないユーザーであるが、ユーザー切り替え機能によるアクセスをブロックしなければならないユーザーの名前を追加します。

タスクの結果

ユーザー切り替えが使用されると、WebSEAL は、**securitygroup** という Security Access Manager グループのメンバーシップも検査します。このグループには、Security Access Manager 管理ユーザー **sec_master** の名前、およびユーザー切り替え機能によるアクセスから除外しなければならないいくつかの WebSEAL プロセスが入ります。

securitygroup グループは、WebSEAL サーバーのインストールの際に、デフォルトで自動的に作成されます。インストールの際に、以下の ID がこのグループに自動的に追加されます。

- **sec_master** — Security Access Manager アドミニストレーター
- **acld** -- Security Access Manager Authorization server デモン
- **webseald** — WebSEAL デモン

WebSEAL アドミニストレーターは、**securitygroup** グループに、ユーザーを追加してはなりません。ユーザー切り替えへのユーザー・アクセスをコントロールするには、**su-admins** または **su-excluded** のどちらかを使用します。

ユーザー切り替え HTML フォームの構成

WebSEAL には、アドミニストレーターがユーザー切り替え機能を使用するためにアクセスするデフォルトの HTML フォームが用意されています。このデフォルト・フォームは変更せずに使用できます。必要に応じて、フォームを編集して、外観と機能をカスタマイズすることができます。

このタスクについて

このステップはオプションです。

デフォルト・フォームの名前は **switchuser.html** です。このファイルの名前は変更できます。

LMI を使用して、**management/lang** ディレクトリーにあるこのファイルにアクセスできます。**lang** ディレクトリーの値は、ロケール固有の値です。例えば、米国英語ロケールの場合の **lang** ディレクトリーには「C」という名前が付いています。

フォームのコンテンツ

このフォームには以下のものに対する要求が含まれています。

- ユーザー名

アドミニストレーターがアクセスする必要がある資格情報を持っているユーザーの名前。

- 宛先 URL

このページは、ユーザー切り替え操作が成功した後で表示されます。

- 認証方式

この認証方式スタンザ・エントリーは、WebSEAL がユーザー資格情報を作成するために、どの認証メカニズムを使用するかを指定します。

これらのエントリーはすべて必須です。WebSEAL は、サブミットされたフォームにすべての必須データが存在することを確認します。欠落しているデータがある場合は、フォームは、データ欠落を示すメッセージと共にアドミニストレーターに戻されます。必要なデータがすべて存在していると、WebSEAL は、ユーザー切り替えフォーム・データにあるデータを、`/pkmsu.form` アクション URL にサブミットします。

注: このフォームを起動できるのは、**su-admins** グループのメンバーだけです。このファイルに対しては ACL は必要ありません。WebSEAL は、内部にハードコーディングされたグループ・メンバーシップ・チェックを行います。グループ・メンバーシップ・チェックが失敗すると、WebSEAL は **404「見つかりません」** エラーを戻します。

HTML フォームのカスタマイズ

ユーザー切り替えフォームをカスタマイズするには、編集するためにフォームをオープンし、以下のステップを実行します。

手順

1. 宛先 URL のロケーションと内容を指定します。
この URL は、該当のホーム・ページまたは成功したユーザー切り替えの確認ページを含む隠し入力として構成できます。
2. 認証方式を指定します。
このフィールドは隠し入力として構成できます。認証方式の有効な値には以下があります。
`su-ba su-forms su-certificate su-http-request su-cdsso`
このリストのメソッドは、WebSEAL 構成ファイルに指定されている認証メカニズムに直接マップします。ただし、**su-ba** > **su-forms** メソッドは、両方とも、**su-password** 認証メカニズムにマップすることに注意してください。基本認証 (ba) およびフォーム認証 (forms) は両方とも **su-password** 認証モジュールを使用します。WebSEAL デプロイメントは、フォーム認証をサポートせずに基本認証をサポートできることに注意してください。したがって、それぞれの認証タイプ (**su-ba** > **su-forms**) ごとに、別々の構成値が維持されています。

追加入力フォームの設計

このタスクについて

このステップはオプションです。

`/pkmsu.form` にサブミットされるデータを検証または処理するための追加フォームを設計できます。これらのフォームを使用して、アドミニストレーターがユーザー切り替えフォームにエントリーの一部を取り込む際に役立てることができます。

次に、いくつかの例を示します。

- アドミニストレーターが、さまざまな宛先 URL を使用することに決め、これらをユーザー ID に基づいてアクセスできるとします。これらの URL のリストを作成し提示する別のフォームを作成し、このリストから、アドミニストレーターが必要なエントリーを選択できるようにします。

- 別のプログラム (例えば、CGI スクリプト) を呼び出すフォームを作成し、ユーザー切り替えが使用できるユーザー ID のリストを指定します。このリストを使用すると、アドミニストレーターは、あるユーザー ID に、ユーザー切り替えを使用してアクセスできるかどうかを判別できます。
- ユーザー切り替えが使用できないユーザー ID のリストを示すフォームを作成できます。このリストは、su-excluded グループおよび securitygroup グループのメンバーシップを基にするとします。

WebSEAL の停止と再始動

このタスクについて

構成の新規変更を活動化するには、WebSEAL を停止し、再始動する必要があります。これによって、WebSEAL は、188 ページの『ユーザー・アクセス権限の構成』で説明した、WebSEAL 構成ファイルに指定した新しい値を使用できるようになります。

WebSEAL サーバーを停止し再始動するメソッドについては、21 ページの『WebSEAL インスタンス管理』に説明があります。

ユーザー切り替えの使用

このタスクについて

前のセクションの構成ステップが完了すると、WebSEAL アドミニストレーターはユーザー切り替え機能を使用できます。

ユーザー切り替え機能を使用するには、次のステップを実行します。

手順

1. ユーザー切り替え機能にアクセスできる許可を持ったユーザーとしてログインします。

この機能は、通常、アドミニストレーターによってアクセスされます。ユーザーは、**su-admins** グループのメンバーでなければなりません。

2. ユーザー切り替え HTML フォームを要求します。

デフォルトのファイル名は `switchuser.html` です。このファイルの詳細については、189 ページの『ユーザー切り替え HTML フォームの構成』を参照してください。

3. フォーム上に、以下のことを指定します。

- アドミニストレーターが代わりに実行するユーザーの ID と名前。
- 宛先 URL。
- 認証方式。

このアクションの結果として、POST 要求が `/pkmssu.form` に送られます。

WebSEAL は、ユーザー切り替えフォームに指定されている宛先 URL へのリダイレクトを、ブラウザに送ります。ユーザーの資格情報を使用して要求が処理され、URL がアクセスされます。

注: pkmsu.form 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

4. 必要に応じて、他の要求をします。

これらの要求についての許可決定は、すべて、このユーザーの資格情報に基づいて行われます。

5. 終了したら、標準の Security Access Manager **/pkmslogout** ユーティリティーを使用して、ユーザー切り替えセッションを終了します。

タスクの結果

ユーザー切り替え機能の作動について詳しくは、185 ページの『ユーザー切り替え機能の概要』を参照してください。

ユーザー切り替えの追加機能のサポート

このセクションでは、再認証、ユーザー・セッション管理、および監査などの追加機能に対するユーザー切り替えのサポートについて説明します。

セッション・キャッシュ・タイムアウトのサポート

構成済みの WebSEAL セッション・キャッシュの非アクティブ・タイムアウト値および存続時間タイムアウト値の機能は、ユーザー切り替え操作の影響を受けません。非アクティブ・タイマーおよび存続時間タイマーは、アドミニストレーターのセッション・キャッシュ・エントリーに関連付けられるもので、ユーザー切り替え操作中に変化するキャッシュ・データには関係ありません。

アドミニストレーターが「切り替え先」ユーザーとして要求を実行している間は、非アクティブ・タイマーはリセットされたままになっています。アドミニストレーターが切り替えユーザー・セッションを終了した後は、再確立されたアドミニストレーター・セッションに対して、非アクティブ・タイマーが再び有効になります。

存続時間値は、ユーザー切り替え操作のために延長されることはありません。したがって、ユーザー切り替え操作中に、セッション・キャッシュ・エントリーの存続時間タイムアウトが発生する可能性があります。このタイムアウトが生じると、セッション・キャッシュは削除され、アドミニストレーターはログオフされます。この場合、アドミニストレーターは、再認証を行い、ユーザー切り替え操作を再度開始する必要があります。

再認証のサポート

ユーザー切り替え操作では、WebSEAL 再認証機能が認識されます。

ユーザー切り替え操作中に再認証が必要になった場合は、アドミニストレーターは「切り替え先」ユーザーとして認証する必要があります。

注: アドミニストレーターは、再認証を正しく行うには、「切り替え先」ユーザーのパスワードを知っていなければなりません。

ユーザー・セッション管理に対するサポート

ユーザー切り替え操作は、ユーザー・セッション管理をサポートしています。

アドミニストレーターは、固有のユーザー・セッション ID を持っています。さらに、ユーザー切り替え操作中は、「切り替え先」ユーザー用の固有のユーザー・セッション ID が存在します。単一ユーザー・セッション終了タスクおよび全ユーザー・セッション終了タスクは、意図したとおりに実行されます。

tag-value のサポート

ユーザー切り替え機能は、カスタム認証モジュールがよく使用する tag-value 機能を認識しサポートしています。

監査のサポート

ユーザー切り替え操作中も、アドミニストレーターを監査することができます。ユーザー切り替え機能は、アドミニストレーターを識別する拡張属性を「切り替え先」ユーザー資格情報に追加します。資格情報に格納されるこの拡張属性は、**tagvalue_su-admin** と呼ばれます。

```
tagvalue_su-admin = su-admin-name
```

どの監査メカニズムでも、この拡張属性を使用できます。

再認証

このセクションは、以下のトピックで構成されています。

- 『再認証の概念』
- 195 ページの『セキュリティ・ポリシーに基づく再認証』
- 195 ページの『再認証 POP: 作成と適用』
- 195 ページの『セッション非アクティブに基づく再認証』
- 196 ページの『セッション非アクティブに基づく再認証の使用可能化』
- 196 ページの『セッション・キャッシュ・エントリーの存続時間値のリセット』
- 197 ページの『セッション・キャッシュ・エントリーの存続時間値の延長』
- 198 ページの『セッション存続時間満了に伴うセッション除去の回避』
- 199 ページの『ログイン失敗ポリシー制限でのユーザー・セッションの削除』
- 200 ページの『再認証用のログイン・フォームのカスタマイズ』

再認証の概念

保護リソースにアクセスしようとしているユーザーが、セッションの開始時に認証された最初のユーザーと同一人物であることを確認するために、Security Access Manager WebSEAL がユーザーに再ログイン (再認証) を要求するように設定することができます。強制再認証により、セキュア・ドメイン内の機密リソースに対する保護をさらに強化することができます。

再認証は以下を使用してアクティブにすることができます。

- 保護オブジェクトに対する保護オブジェクト・ポリシー (POP)。
- WebSEAL セッション・キャッシュ・エントリーの非アクティブ・タイムアウト値の満了。

再認証は以下の WebSEAL 認証方式によりサポートされています。

- フォーム (ユーザー名およびパスワード) 認証
- 外部認証インターフェース

さらに、再認証をサポートするカスタム・ユーザー名およびパスワード・モジュールを作成することもできます。

再認証では、ユーザーが最初にセキュア・ドメインにログインしていること、およびそのユーザー用の有効なセッション (資格情報) が存在していることが前提となります。WebSEAL 構成ファイルの **[reauthentication]** スタンザ内の **reauth-at-any-level** オプションによって、WebSEAL が再認証操作を処理する方法が以下のように決定されます。

- このオプションの値が **no** の場合、ユーザーはその既存の資格情報を生成したときと同じ ID、認証方式、認証レベルを使用してログインする必要があります。WebSEAL は、再認証時に、資格情報を含めたそのユーザーのオリジナルのセッション情報を保存します。したがって、再認証時に資格情報が取り替えられることはありません。
- このオプションの値が **yes** の場合、ユーザーは同じ ID を使用する必要がありますが、ユーザーによって現在保持されているものと異なる認証方式または認証レベルを使用して認証できます。この場合、ユーザーの資格情報は、ユーザーのセッションの存続時間中に 1 回以上変更される可能性があり、さらに正常に再認証されたときに更新されます。これにより起きうるいくつかの結果については、注意深く検討する必要があります。確立されたセッション中に資格情報が変更されると、許可決定や監査などの資格情報属性を使用する操作は、セッション中に異なる結果を返す場合があります。一貫性のあるユーザー・エクスペリエンスを確保し、監査記録において、こうしたタイプの変更を考慮するよう、注意が必要です。

WebSEAL は、再認証時に、再認証を求めるプロンプトを出した要求もキャッシュに入れます。再認証が成功すると、キャッシュ・データを使用して要求が再作成されます。233 ページの『サーバー・サイド要求キャッシング』を参照してください。

再認証が失敗した場合は、WebSEAL は再度ログイン・プロンプトを戻します。再認証は成功したが、目的のリソースについての ACL 検査が失敗した場合は、403 エラー (「禁止」) が戻され、ユーザーは、要求したリソースへのアクセスを拒否されます。

どちらの場合も、ユーザーはログオフされることはありません (**max-login-failures** ポリシー制限に達した場合を除きます)。ユーザーは、まだ有効性を維持している資格情報を使用して、再認証プロセスを終了させることも (別の URL を要求する)、また、セキュア・ドメインに参加したまま、再認証を必要としない他のリソースにアクセスすることもできます。

再認証の試行が **max-login-failures** ポリシー制限に達した場合に、WebSEAL がユーザーのセッション・キャッシュ・エントリーを除去し、ユーザーをログアウトさせなければならない構成オプションがあります。

別の構成オプションは、WebSEAL セッション・キャッシュ・エントリーの存続時間タイマーのリセットに使用可能です。さらに、猶予期間を構成することによって、セッション・キャッシュ・エントリーの存続時間タイムアウトが生じる前に、再認証プロセスが完了するための十分な時間が与えられるようにすることもできます。

セキュリティ・ポリシーに基づく再認証

セキュリティ・ポリシーに基づく再認証は、POP 内の特定の拡張属性により活動化されます。この拡張属性は、要求されたリソース・オブジェクトを保護します。POP はオブジェクトに直接付随させることもでき、また、オブジェクトが親オブジェクトから POP 条件を継承することもできます。

再認証 POP: 作成と適用

セキュリティ・ポリシーに基づく強制再認証を構成するには、「reauth」という名前の特殊拡張属性を指定した保護オブジェクト・ポリシー (POP) を作成します。この POP は、強制再認証が提供する追加の保護を必要とする任意のオブジェクトに付加することができます。

この POP が付随しているオブジェクトのすべての子も、POP 条件を継承するという点に注意してください。つまり、これらの子オブジェクトが要求された場合も、それぞれ再認証が必要になります。

pdadmin pop create、**pdadmin pop modify**、および **pdadmin pop attach** コマンドを使用して、再認証 POP を作成および適用します。次の例では、**reauth** 拡張属性を持つ「secure」という名前の POP が作成され、その POP がオブジェクト (budget.html) に付加されます。

```
pdadmin> pop create secure
pdadmin> pop modify secure set attribute reauth true
pdadmin> pop attach /WebSEAL/hostA/junction/budget.html secure
```

budget.html にアクセスしようとするユーザーは、既存の資格情報を生成したときと同じ識別および認証方式を使用して再認証を行うよう要求されます。

リソースを要求しているユーザーが非認証ユーザーである場合は、POP はそのユーザーに認証を行うよう要求します。初期ログインが成功した後は、このリソースに対する再認証は不要です。

pdadmin pop コマンドについて詳しくは、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

セッション非アクティブに基づく再認証

セッション非アクティブに基づく再認証は、構成スタンザ・エントリーにより使用可能にされ、セッション・キャッシュ・エントリーの非アクティブ・タイムアウト値の満了と同時にアクティブになります。

通常、ユーザーのセッションは、セッション非アクティブ値およびセッション存続時間値により規制されます。WebSEAL が、セッション非アクティブ値に基づいて

再認証を行うように構成されている場合は、セッション非アクティブ・タイムアウト値が満了するたびに、ユーザーのセッション・キャッシュ・エントリーに「フラグ」が立てられます。セッション・キャッシュ・エントリー (ユーザーの資格情報が入っている) は、削除されることはありません。ユーザーは、無保護リソースへのアクセスを続けることができます。しかし、このユーザーが保護リソースを要求した場合は、WebSEAL はログイン・プロンプトを送ります。再認証が成功すると、非アクティブ・セッションの「フラグ」は削除され、非アクティブ・タイマーはリセットされます。

再認証が失敗した場合は、WebSEAL は再度ログイン・プロンプトを戻します。セッション・キャッシュ・エントリーはフラグが立てられたままになり、ユーザーは、セッション・キャッシュ・エントリー存続時間が過ぎるまで、無保護リソースを要求し続けることができます。

ユーザー・セッションを終了させる条件は、他に 2 つあります。それは、ユーザーが明示的にログアウトした場合と、アドミニストレーターがユーザー・セッションを打ち切った場合です。679 ページの『ユーザー・セッションの終了』を参照してください。

セッション非アクティブに基づく再認証の使用可能化

非アクティブ・セッションをセッション・キャッシュから削除せずに「フラグ」を立てるように WebSEAL を構成するには、webseald.conf 構成ファイルの **[reauthentication]** スタンザで、**reauth-for-inactive** スタンザ・エントリーを「yes」に設定します。

```
[reauthentication]
reauth-for-inactive = yes
```

このスタンザ・エントリーのデフォルト値は「no」です。

Session Management Server (SMS) のセッション非アクティブに基づく再認証の使用可能化については、370 ページの『SMS の最終アクセス時間更新頻度の調整』を参照してください。

セッション・キャッシュ・エントリーの存続時間値のリセット

ユーザーのセッション・キャッシュ・エントリーには、限られた存続時間が設定されています。この値を指定するには、webseald.conf 構成ファイルの **[session]** スタンザ内の **timeout** スタンザ・エントリーを使用します。デフォルト値 (秒) は、3600 (1 時間) です。

```
[session]
timeout = 3600
```

セッションがアクティブか非アクティブかに関係なく、存続時間の値に達した時点でセッション・キャッシュ・エントリーは除去され、同時にユーザーはログオフされます。

ただし、再認証が発生するたびにセッション・キャッシュ・エントリーの存続時間がリセットされるように、構成することができます。この構成では、ユーザー・セ

セッションの最大存続時間値は単一のものでなくなります。つまり、再認証が発生するたびに、セッション・キャッシュ・エントリーの存続時間値はリセットされます。

セッション・キャッシュ・エントリー存続時間のリセットを構成するには、`webseald.conf` 構成ファイルの **[reauthentication]** スタンザ内の **reauth-reset-lifetime** スタンザ・エントリーを使用します。

```
[reauthentication]
reauth-reset-lifetime = yes
```

デフォルト値は「no」です。

セッション・キャッシュ・エントリーの存続時間値の延長

ユーザーが再認証を行っている間に、セッション・キャッシュ・エントリーの存続時間値が満了してしまうことがあります。この状況が生じるのは次のような場合です。

- ・ ユーザーが、再認証 POP により保護されているリソースを要求している。
- ・ ユーザーのセッション・キャッシュ・エントリー存続時間値の満了が間近に迫っている。

再認証ログイン・フォームがユーザーに送られてから、記入済みのログイン・フォームが戻されるまでの間に、セッション・キャッシュ・エントリーの存続時間が満了することがあります。セッション・キャッシュ・エントリー存続時間値が満了すると、セッション・キャッシュ・エントリーは削除されます。したがって、ログイン・フォームが WebSEAL に戻されたときには、既にそのユーザー用のセッションは存在していません。しかも、キャッシュ内のユーザー要求データもすべて失われています。

再認証中にセッション・キャッシュ・エントリーの存続時間が満了した場合に備えて、セッション・キャッシュ・エントリー存続時間値の延長、つまり「猶予期間」を設定できます。この時間延長 (秒数) を指定するには、`webseald.conf` 構成ファイルの **[reauthentication]** スタンザ内の **reauth-extend-lifetime** スタンザ・エントリーを使用します。例えば次のように入力します (5 分を指定する場合)。

```
[reauthentication]
reauth-extend-lifetime = 300
```

デフォルト値は「0」で、これはセッション・キャッシュ・エントリーのタイムアウト値を延長しないことを意味します。

reauth-extend-lifetime スタンザ・エントリーは、既存のセッション・キャッシュ・エントリーを持ち、再認証が必要とされるユーザーに適用されます。例:

- ・ POP セキュリティー・ポリシーに従って再認証を行うユーザー
- ・ セッション・キャッシュが非アクティブであることが原因で再認証を行うユーザー
- ・ ステップアップ認証を行うユーザー

reauth-extend-lifetime オプションは、**reauth-reset-lifetime=yes** オプションと併用するためのものです。

セッション存続時間満了に伴うセッション除去の回避

通常、セッション・キャッシュ・エントリー存続時間値はセッションの最大長を決定します。ユーザーが、セッション存続期間中最後までアクティブな状態を維持している場合があります。この場合に、セッション存続期間が過ぎると、アクティビティに関係なく、セッション・キャッシュ・エントリーは削除され、ユーザーはログオフされます。

このようにセッションが突然終了するのを防ぐには、セッション・タイムアウト値が満了したときにユーザーが再認証できるように、WebSEAL を構成することができます。再認証が成功すると、セッション・キャッシュ・エントリーの存続時間値はリセットされます。

WebSEAL では、以下の条件が満たされているときに、満了したセッション存続時間値をリセットできます。

- 非アクティブ・ポリシーに基づく再認証が使用可能にされている (reauth-for-inactive=yes)。
- セッション存続時間値 (timeout) が満了している。
- セッション存続時間の時間延長 (「猶予期間」) が使用可能にされており、妥当な値に設定されている (例えば、reauth-extend-lifetime=300)。
- ユーザーが、時間延長 (「猶予期間」) が満了する前に保護リソースを要求することにより、再認証プロンプトをアクティブにしている。

(WebSEAL では、セッション存続時間終了イベントに対して、時間延長を何度も繰り返すことはできません。)

- セッション・キャッシュ存続時間のリセットが真に設定されている (reauth-reset-lifetime=yes)。

セッション存続時間が満了すると、WebSEAL は上記の条件をチェックします。すべての条件が満たされていれば、存続時間タイムアウトは **reauth-extend-lifetime** の値だけ延長され、ユーザーのセッション・キャッシュ・エントリーには延長の「フラグ」が立てられます。セッション・キャッシュ・エントリー (ユーザーの資格情報が入っている) は削除されず、ユーザーは無保護リソースへのアクセスを続行することができます。ユーザーが保護リソースを要求した場合は、WebSEAL は、ユーザーに、再認証を受けるようプロンプトを出します。

reauth-extend-lifetime 値は、ユーザーに再認証プロンプトを起動するための時間的余裕が十分与えられるように、妥当な値に設定してください。ユーザーが「猶予期間」内に保護オブジェクトにアクセスしない場合は、再認証プロセスはアクティブにされないという点に注意してください。その場合は、**reauth-extend-lifetime** 値が満了し、その結果、セッション・キャッシュ・エントリーは削除されます。

ただし、再認証ポリシーは、主として保護リソースに関するサービスを提供するアプリケーションを保護するようにインプリメントされているのが普通です。アクティブ・ユーザーが再認証プロセスをトリガーし、セッション存続時間値をリセットできるようにするには、時間延長 (「猶予期間」) は 5 分から 10 分程度で十分です。

ログイン失敗ポリシー制限でのユーザー・セッションの削除

再認証の試行が失敗した場合、WebSEAL はログイン・プロンプトを再度戻します。まだ有効なセッションおよび資格情報を保有しているため、ユーザーは (別の URL を要求することによって) 再認証プロセスを終了し、再認証の必要がない他のリソースにアクセスすることによって、セキュア・ドメインに参加し続けることができます。

ただし、ユーザーが再認証の試行を失敗し続けると、再認証プロセスはログイン失敗ポリシー (**max-login-failures**) の影響を受けます。再認証試行の失敗回数が **max-login-failures** 制限に達するか超えた場合、WebSEAL は、**terminate-on-reauth-lockout** 構成に従って応答します。

terminate-on-reauth-lockout スタンザ・エントリーは、WebSEAL 構成ファイルの **[reauthentication]** スタンザにあります。このスタンザ・エントリーの目的は、**max-login-failures** ポリシー制限に達した際にユーザーのセッション・キャッシュ・エントリーを完全に削除するかどうかを制御することです。

デフォルト設定は「yes」です。再認証中にログイン失敗回数の最大数 (**max-login-failures** ポリシーで指定) に達した際、ユーザーは元のセッションからログアウトされ、ユーザーのセッション・キャッシュ・エントリーは削除されます。以下に例を示します。

```
[reauthentication]
terminate-on-reauth-lockout = yes
```

ユーザーはもう有効なセッションおよび資格情報を保有していません。ユーザーは引き続き無保護リソースにはアクセスできますが、保護リソースに対する要求を行うためには再度ログインする必要があります。

バージョン 6.0 より前のバージョンの WebSEAL への後方互換性を保つには、**terminate-on-reauth-lockout** スタンザ・エントリーに「no」の値を入力します。

```
[reauthentication]
terminate-on-reauth-lockout = no
```

「no」に設定すると、ユーザーはログアウトされず、最初のログイン・セッションがそのまま有効になります。ユーザーは、**reauth** POP によって保護されていない他のリソースに引き続きアクセスできます。

再認証中にログイン失敗回数の最大数 (**max-login-failures** ポリシーで指定) に達すると、ユーザーは **disable-time-interval** ポリシー設定で指定したようにリソースへのアクセスからロックアウトされ、**late-lockout-notification** 構成設定の指定に従ってロックアウトされたことを通知されます。

terminate-on-reauth-lockout の両方の値については、ユーザーへの特定の応答は **disable-time-interval** および **late-lockout-notification** 設定によって決まります。

disable-time-interval ポリシーが秒数に設定されている場合、アカウントが一時的にロックアウトされていることを示すエラー・メッセージが表示されます。

disable-time-interval ポリシーが「disable」に設定されている場合、アカウントが使用不可でアドミニストレーターがそのアカウントをリセット（アンロック）する必要があります。あることを示すエラー・メッセージが表示されます。

ログイン失敗ポリシーのメカニズムの詳細については、239 ページの『ログイン失敗ポリシー（「スリー・ストライク」ログイン・ポリシー）』を参照してください。

再認証用のログイン・フォームのカスタマイズ

WebSEAL は、フォーム認証方式認証方式の両方について、再認証をサポートしています。

デフォルトのフォーム認証では、login.html ページを使用して、クライアントに対してユーザー名およびパスワード情報が要求されます（79 ページの『静的 HTML サーバー応答ページ』を参照）。このデフォルトのログイン・ページは、再認証のときにも使用されます。

USERNAME マクロを使用することにより、再認証時に、これらのログイン・ページ内のユーザー名フィールドに自動的に名前が入るようにすることもできます（87 ページの『HTML 応答ページのカスタマイズのためのマクロ・リソース』を参照）。この場合、ユーザーはパスワード（パスコード）フィールドのみを入力すれば済みます。

例えば、login.html ページ内の次の行を変更します。

```
<TD><INPUT NAME="username" SIZE="15"></TD>
```

上記の行に、次のように USERNAME マクロを組み込みます。

```
<TD><INPUT NAME="username" SIZE="15" VALUE="%USERNAME%"></TD>
```

初期ログイン（非認証ユーザー）の時点では、USERNAME マクロの値は空で、ログイン・ページに表示されるユーザー名テキスト・フィールドには、エントリーに「unknown」が表示されます。

再認証するクライアントの場合、USERNAME マクロにはそのクライアント・ユーザー名の値が含まれています。そして、ログイン・ページ上のユーザー名テキスト・フィールドには、自動的にユーザーの名前が表示されます。

認証強度ポリシー（ステップアップ）

このセクションは、以下のトピックで構成されています。

- 201 ページの『認証強度の概念』
- 202 ページの『認証強度構成タスクの要約』
- 203 ページの『認証強度ポリシーの確立』
- 203 ページの『認証レベルの指定』
- 205 ページの『認証強度ログイン・フォームの指定』
- 206 ページの『保護オブジェクト・ポリシーの作成』
- 207 ページの『ネットワーク・ベースのアクセス制限の指定』
- 210 ページの『保護リソースへの保護オブジェクト・ポリシーの付加』

- 211 ページの『複数の認証レベル間でのユーザー ID マッチングの実行』
- 212 ページの『非認証ユーザーへのログイン応答の制御』

認証強度の概念

WebSEAL は、多くの認証方式をサポートしています。認証方式には、基本認証、フォーム認証、証明書認証などがあります。WebSEAL サーバーにアクセスするクライアントは、いずれも、非認証 または証明書 などの認証状態を持っています。これらは、WebSEAL が最後にクライアントを認証したときのメソッドを示しています。

WebSEAL には、アドミニストレーターが、いくつかのサポートされている認証方式にランキングまたはレベルを割り当てることができるようにする機能が用意されています。アドミニストレーターは、それぞれの認証方式に、最低から最高まででランク付けする番号付きリストを定義できます。この階層ランキングは、個々の WebSEAL デプロイメントに合わせて任意に調整できます。

各種の認証方式間には、絶対ランキングはありません。ある認証方式が、別の認証方式に比べて本質的に優れているとか強力であるということではありません。つまり、ランキングは、特定の Security Access Manager WebSEAL 保護オブジェクト・ネーム・スペースで使用するそれぞれの認証方式の相対レベルを、アドミニストレーターが定義するために使用する単なる方法にすぎません。レベルの割り当てを支配する唯一のルールは、非認証 レベルが常にその他のすべての認証済みレベルより低いということです。

この認証レベルのセットを使用して、認証強度ポリシーをインプリメントすることができます。認証強度は、場合により、ステップアップ認証 と呼ばれます。ただし、このステップアップ認証は、フォーム認証あるいは証明書認証のような固有の認証方式ではないことに注意してください。これは、ユーザーが、現在使用している認証方式を別の認証方式に変更できるようにするために定義されたプロセスです。

認証方式を変更するという概念は、WebSEAL 保護オブジェクト・ネーム・スペースの中にある一定のリソースに保護を追加する方法として便利な方法です。例えば、ユーザーは、証明書認証を使用してログインし、Security Access Manager セキュリティーで保護されている多くのリソースにアクセスできます。ユーザーが、より高いレベルのアクセスが必要であるとマークされている、より機密度が高いリソースにアクセスしようとする、ユーザーにプロンプトが出されて、別の認証レベルにログインするよう求められます。

ユーザーが保護オブジェクトにアクセスしようとして認証強度を活動化するとき、ユーザーがまずログアウトしなくてもよいことに注意してください。代わりに、ユーザーに対してログイン・プロンプトが表示され、より高いレベルに再びログインするだけですみます。

ユーザーは、1 つの認証セッションで、複数回認証強度を変更できます。制御用 POP に指定されている認証レベルが、ユーザーがどのレベルで認証される必要があるかを制御します。

以下の認証方式に、認証レベルを割り当てることができます。

- 非認証
- パスワード認証

パスワード認証は、フォーム認証に限定されます。基本認証は、ステップアップ認証レベルとしてサポートされません。

- 証明書認証
- 外部認証インターフェース

認証強度は、証明書認証を除いて、HTTP および HTTPS のどちらでもサポートされています。証明書認証は SSL 接続を介してのみ有効であるので、HTTP を介して証明書認証にステップアップすることはできません。証明書認証が必要なオブジェクトが HTTP を介して要求されると、WebSEAL 構成ファイルの **[acnt-mgt]** スタンザ内の **cert-stepup-http** スタンザ・エントリーによって指定されているように、エラー・ページが出されます。

アドミニストレーターは、標準 Security Access Manager 保護オブジェクト・ポリシー (POP) を宣言し、そのポリシーをリソース・オブジェクトに付加することによって、認証レベルを保護リソースに適用します。認証強度ポリシーは、IP エンドポイント認証方式 という POP 属性の中に設定され保管されます。この属性は、整数値を使用して認証レベルを表します。最低レベルである非認証 は常時 0 です。各レベルは、整数索引値を、レベルが割り当てられている認証方式の合計数に達するまで増分します。

WebSEAL によってクライアントが初めて認証されると、使用された初期認証方式が、拡張属性としてクライアントの資格情報の中に保管されます。Security Access Manager 許可サービスは、資格情報の中の認証方式 (レベル) と、要求されたリソースの認証レベル (POP に指定されている) とを比較します。POP のレベルが資格情報のレベルを超えると、ユーザーは認証強度レベルを上げて認証するように要求されます。

また、IP エンドポイント認証方式属性をオプションで使用して、アクセス要求を出したクライアントのネットワーク・アドレスに基づいて、リソースへのアクセスを制限することもできます。アクセスの制限は、個々のネットワーク・アドレス (IP アドレス)、あるいは、ある範囲のネットワーク・アドレスのどちらに基づいても行えます。

WebSEAL は、以下のアルゴリズムを使用して、POP 内の条件を処理します。

1. POP の IP エンドポイント認証方式ポリシーを検査します。
2. ACL 許可を検査する。
3. POP に関する時刻ポリシーを検査する。
4. POP の監査レベル・ポリシーを検査します。

認証強度構成タスクの要約

認証強度レベルを構成するには、以下のセクションのそれぞれの手順を実行してください。

1. 203 ページの『認証強度ポリシーの確立』
2. 203 ページの『認証レベルの指定』

3. 205 ページの『認証強度ログイン・フォームの指定』
4. 206 ページの『保護オブジェクト・ポリシーの作成』
5. 207 ページの『ネットワーク・ベースのアクセス制限の指定』
6. 210 ページの『保護リソースへの保護オブジェクト・ポリシーの付加』
7. 211 ページの『複数の認証レベル間でのユーザー ID マッチングの実行』
8. 212 ページの『非認証ユーザーへのログイン応答の制御』
9. 212 ページの『より高いレベルへの認証のステップアップ』

認証強度ポリシーの確立

このタスクについて

このセクションでは、WebSEAL 構成ファイルに認証強度設定値を指定する前にとるべき計画ステップを実行します。

以下のステップを実行してください。

手順

1. 特定の認証方式を使用して認証されたユーザーにのみアクセスが制限される保護オブジェクトのリストを編集します。それぞれの保護オブジェクトごとに、適用される認証方式を指定します。
2. WebSEAL サーバー・システムでアクティブになる (使用可能になる) すべての認証メカニズムのリストを編集します。
3. アクティブ認証メカニズムの階層 (ランキング) を決定します。最も弱いメカニズムから最も強いメカニズムまでの順番をつけます。
4. 認証強度レベルのステップアップの際、ユーザー ID を、ステップアップした認証レベルでの ID も同じ ID にすべきかを決定します。
5. 要求クライアントのネットワーク・アドレスに基づいてアクセスを制限する必要がある保護リソースがあるかを決定します。
6. WebSEAL サーバーを停止します。

認証レベルの指定

このタスクについて

認証レベルを指定するには、以下のステップを実行します。

手順

1. WebSEAL 構成ファイル内の **[authentication-levels]** スタンザを編集します。認証レベルのステップアップに使用するそれぞれの認証方式ごとに、1 つのエントリーをスタンザに追加します。次の表には、サポートされている認証方式の説明があります。

表 22. 認証強度用にサポートされている認証方式

認証方式	構成ファイル・エントリー
なし	level = unauthenticated
フォーム認証	level = password

表 22. 認証強度用にサポートされている認証方式 (続き)

認証方式	構成ファイル・エントリー
証明書認証	level = ssl
外部認証インターフェース	level = ext-auth-interface

デフォルト・エントリーは、以下のとおりです。

```
[authentication-levels]
level = unauthenticated
level = password
```

エントリー level = unauthenticated は、常にリストの先頭になければなりません。その他のエントリーの順序は、どのような順序でもかまいません。例えば、証明書認証の認証強度レベルを最高レベルで使用可能にするには、完了したスタンザのエントリーは次のようになります。

```
[authentication-levels]
level = unauthenticated
level = password
level = ssl
```

2. **[authentication-levels]** にリストされているそれぞれの認証方式が使用可能になっていることを確認します。認証方式が使用可能になっているかどうかを判別するには、WebSEAL 構成ファイル内の該当のエントリーを検査します。必要なエントリーを検討し、認証構成手順にアクセスするには、以下のセクションを参照してください。

- 160 ページの『基本認証の使用可能化および使用不可』
- 161 ページの『フォーム認証の使用可能および使用不可』
- 167 ページの『証明書認証の使用可能化』

注: 基本認証は、デフォルトで使用可能になっています。

複数の認証レベルの使用

特定の認証メカニズムに複数の認証レベルを関連付けることができます。

認証メカニズムでは、正常な認証の結果得られた認証レベルを、属性として資格情報に直接設定することができます。その場合、認証レベルは **[authentication-levels]** スタンザ内の認証メカニズムの配置によって資格情報に設定されたレベルをオーバーライドします。この方式を使用して、結果として得られる Security Access Manager 資格情報に設定される認証レベルを指定できます。この方式は、特定の認証メカニズムに複数のレベルを関連付ける場合に使用します。

認証メカニズムを **[authentication-levels]** スタンザに複数回入力する理由には、次のものがあります。

1. ユーザーのステップアップが必要なときにユーザーに表示されるプロンプトを制御するため。
2. 構成されたすべての POP レベルを処理できる方法が存在するというポリシー・サーバー要件を満たすため。

このことは、外部認証インターフェース (EAI) サーバーを使用して認証を実行するときに考慮事項となる可能性があります。EAI に複数の認証レベルを処理させるの

が一般的です。EAI サーバーは、特権属性証明書 (PAC) を戻す場合はその属性として、ユーザー ID を戻す場合は拡張属性ヘッダーとして、認証レベルを戻すことができます。これにより、EAI 認証メカニズムを、**[authentication-levels]** スタンザ内の複数の行に指定できます。例えば、認証するユーザーをレベル 2 およびレベル 3 で処理するように EAI サーバーが構成される一方で、ユーザーをレベル 1 で認証するためにフォーム認証が使用される場合、**[authentication-levels]** スタンザには以下のエントリーが含まれます。

```
[authentication-levels]
#-----
# STEP UP
#-----
# authentication levels
#
# Syntax:
# level = <method-name>
#
# Valid method names are:
#   unauthenticated
#   password
#   ssl
#   ext-auth-interface
#
level = unauthenticated
level = password
level = ext-auth-interface
level = ext-auth-interface
```

認証強度ログイン・フォームの指定

このタスクについて

認証クライアントが保護リソースにアクセスしようとし、より高い認証強度レベルへの再認証が必要になったとき、WebSEAL は特殊 HTML フォームを提示します。クライアントはこのフォームを使用して、必要な認証のタイプで必要とされる情報を提供します。

WebSEAL はデフォルト・ログイン・フォームを提供します。アドミニストレータは、デフォルトのログイン・フォームを使用することも、ローカルの WebSEAL デプロイメントに合うようにフォームをカスタマイズすることもできます。

デフォルトのログイン・フォームのロケーションは、WebSEAL 構成ファイルの中で指定されます。

```
[acct-mgt]
stepup-login = stepuplogin.html
```

以下のステップを実行してください。

手順

1. 認証強度のログイン・フォームの名前を指定します。

フォームのデフォルト・ロケーションを使用するには、WebSEAL 構成ファイルのスタンザ・エントリー **stepup-login** に、デフォルト値 **stepuplogin.html** が入っていることを確認してください。

2. オプションで、認証強度ログイン・フォームの内容をカスタマイズします。

このファイルには、%TEXT% シーケンスの形式のマクロが入っていて、このマクロは適切な値に置き換えられます。この置換は、WebSEAL のテンプレート・ファイル処理機能内で行われますが、これによって、正しくフォーマットされた、サポートされる認証方式に、このフォームを使用できるようになります。また、エラー・メッセージおよび認証方式名などのその他の情報を、ユーザー用のフォームに提供することもできます。

マクロの使用について詳しくは、87 ページの『HTML 応答ページのカスタマイズのためのマクロ・リソース』を参照してください。

3. WebSEAL サーバーを再始動します。

タスクの結果

認証強度レベルの構成はこれで完了しました。

保護オブジェクト・ポリシーの作成

このタスクについて

以下のステップを実行してください。

手順

1. POP を作成します。例えば、次のように、**pdadmin** を使用して **test** という名前の新規 POP を作成します。

```
pdadmin> pop create test
```

2. 次のようにして、新規 POP の内容を表示します。

```
pdadmin> pop show test
```

新規 POP には、次のような新しい設定値が入っています。

```
pdadmin> pop show test
Protected object policy: test
Description:
Warning: no
Audit level: none
Quality of protection: none
Time of day access: sun, mon, tue, wed, thu, fri, sat:
anytime:local
IP Endpoint Authentication Method Policy
Any Other Network 0
```

3. POP の中の、属性 IP エンドポイント認証方式ポリシーのデフォルト値をメモしておきます。

```
...
...
IP Endpoint Authentication Method Policy
Any Other Network 0
...
```

IP エンドポイント認証方式ポリシー属性は、2 つの異なる属性を指定するために使用されます。

- 認証強度レベル。

デフォルト値は 0 です。

- ネットワーク・ベースのアクセス・ポリシー。

デフォルト値は Any Other Network です。

4. **pdadmin pop modify** を使用して IP エンドポイント認証方式ポリシー属性を変更し、203 ページの『認証強度ポリシーの確立』で指定されるリソースに適用する認証強度レベルを指定します。構文は次のようになります。

```
pdadmin> pop modify pop-name set ipauth anyothernw level-index
```

値 *level-index* は整数です。デフォルト値は 0 です。デフォルト値は、認証強度レベル *unauthenticated* にマップされます。

必要な認証強度レベルに対応する索引を指定します。正しい *level-index* を判別するには、WebSEAL 構成ファイルの **[authentication-level]** スタンザを調べてください。

```
例:[authentication-levels]
level = unauthenticated
level = password
level = ssl
```

次の表に、上記のエントリーの索引値がリストされています。

表 23. 認証強度レベルの整数値の例

認証方式	索引値
unauthenticated (非認証)	0
password	1
ssl	2

例えば、パスワード認証強度レベル (索引値 1) を *test* POP に追加するには、
pdadmin> pop modify test set ipauth anyothernw 1 と入力します。変更を確認するには、次のようにして POP を表示します。

```
pdadmin> pop show test
Protected object policy: test
Description: Test POP
Warning: no
Audit level: none
Quality of protection: none
Time of day access: sun, mon, tue, wed, thu, fri, sat:
anytime:local
IP Endpoint Authentication Method Policy
Any Other Network 1
```

注: この例では、有効な索引値は 0、1、2 だけです。その他の索引値が構成されている場合、クライアントが、その POP が付加されているオブジェクトを要求するたびに、WebSEAL はエラー・ページを表示します。

ネットワーク・ベースのアクセス制限の指定

このタスクについて

Security Access Manager は、指定したネットワーク・アドレスから発信されたクライアント要求に認証強度レベルの適用を使用可能にする、オプションの POP 構成設定をサポートします。ネットワーク・アドレスは、単一 IP アドレスでも、一定範囲の IP アドレスとしても、どちらでも定義できます。

注: ほとんどのデプロイメントでは、ユーザー・アクセスは、POP 内で、IP アドレスを基にして制限されていません。したがって、ほとんどのデプロイメントでは、この構成セクションはスキップしてかまいません。

pdadmin pop modify set ipauth コマンドは、IP アドレスを指定するために使用されます。これは、認証レベルを指定するために使用される、同じ **pdadmin** コマンドであることに注意してください。

pdadmin pop modify set ipauth のデフォルトの使用法では、ネットワーク・ベースのアクセス制限を課すことはありません。この使用法では、IP エンドポイント認証方式ポリシー属性の値としてコマンド行引数 **anyothernw** を指定します。この設定は、リクエストの IP アドレスに関係なくすべてのユーザー・アクセスに適用され、指定したレベルで、すべてのユーザーが認証されることを必要とします。

構文は次のようになります。

```
pdadmin> pop modify pop-name set ipauth anyothernw level_index
```

例えば、206 ページの『保護オブジェクト・ポリシーの作成』において、下記のコマンドでは、すべてのユーザーが認証レベル 1 で認証されることを必要とする POP が作成されましたが、ネットワーク・ベースのアクセス要件を課していませんでした。

```
pdadmin> pop modify test set ipauth anyothernw 1
```

手順

以下のネットワーク・ベースのアクセス制限が適用できます。

- 要求クライアントの IP アドレスが IP アドレスの定義済み範囲内にあるときに、特定の認証強度レベルを必要とする。

構文:

```
pdadmin> pop modify pop_name set ipauth add network netmask level_index
```

pdadmin pop modify set ipauth add コマンドが、IP エンドポイント認証方式属性の中に、ネットワーク・アドレスと必要な認証レベルの両方を指定していることに注意してください。

例えば、IP アドレスの範囲 9.1.2.[0-255] のユーザーが認証強度レベル 1 を使用することを必要とするには、次のようにします。

```
pdadmin> pop modify test set ipauth add 9.1.2.1 255.255.255.0 1
```

ネットマスク用に指定した値が、影響を受けるネットワーク・アドレスの範囲を決定していることに注意してください。ネットマスク内の数値 0 が、そのサブネットのすべての IP アドレスを意味するワイルドカードを役をします。次の例を参照してください。

表 24. ネットワーク範囲を指定するネットマスクの使用 (IPv4)

使用法の例		
IPv4 アドレス	ネットマスク	影響を受けるネットワーク範囲
9.1.2.3	255.255.255.0	9.1.2.[0-255]
9.1.2.3	255.255.0.0	9.1.[0-255].[0-255]

表 24. ネットワーク範囲を指定するネットマスクの使用 (IPv4) (続き)

使用法の例		
IPv4 アドレス	ネットマスク	影響を受けるネットワーク範囲
9.1.2.3	255.0.0.0	9.[0-255].[0-255].[0-255]

表 25. ネットワーク範囲を指定するネットマスクの使用 (IPv6)

使用法の例		
IPv6 アドレス	ネットマスク	影響を受けるネットワーク範囲
fec0::1	fff0::	fec[0-f]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]
fec0:ffff::1	ffff:fff0::	fec0:fff[0-f]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]:[0-ffff]

- 1 つの特定の IP アドレスからの要求が、指定した 1 つの認証強度レベルを使用することを必要とする。

例えば、IP アドレス 9.1.2.3 からの要求が認証強度レベル 1 を使用することを必要とするには、次のようにします。

```
pdadmin> pop modify test set ipauth add 9.1.2.3 255.255.255.255 1
```

サブネット 9.1.2.x 上のすべての IP アドレスからの要求が認証強度レベル 1 を使用することを必要とするには、次のようにします。

```
pdadmin> pop modify test set ipauth add 9.1.2.3 255.255.255.0 1
```

- ある範囲のネットワーク・アドレスからのすべての要求による認証強度レベルのステップアップの使用を使用不可にする。

構文は次のようになります。

```
pdadmin> pop modify pop_name set ipauth remove network netmask
```

例えば、サブネット 9.1.2.x 上の範囲の IP アドレスからのすべての要求を使用不可にするには、次のようにします。

```
pdadmin> pop modify test set ipauth remove 9.1.2.1 255.255.255.0
```

- 保護リソースへのアクセスを、認証強度レベルに関係なく、IP アドレスまたはある範囲の IP アドレスに基づいてのみ許可する。

この制限は、IP アドレス (単数または複数) を指定し、さらに、認証レベル 0 (ゼロ) を割り当てることによって実行できます。例えば、認証強度レベルに関係なく、IP アドレス 9.1.2.3 からの要求を許可するには、次のようにします。

```
pdadmin> pop modify test set ipauth add 9.1.2.3 255.255.255.255 0
```

同様に、認証強度レベルに関係なく、サブネット 9.1.2.x 上のすべての IP アドレスからの要求を許可するには、次のようにします。

```
pdadmin> pop modify test set ipauth add 9.1.2.3 255.255.255.0 0
```

- 認証強度レベルに関係なく、IP アドレスまたはある範囲の IP アドレスに基づいてのみ、アクセスを拒否する。

この制限は、最終パラメーターとしてキー・ワード forbidden を使用することによって実行されます。

例えば、IP アドレス 9.1.2.3 にあるクライアントだけを保護リソースにアクセスさせないように制限するには、次のようにします。

```
pdadmin> pop modify test set ipauth 9.1.2.3 255.255.255.255 forbidden
```

同様に、サブネット 9.1.2.x 上のすべての IP アドレスからの要求を、このリソースにアクセスさせないように制限するには、次のようにします。

```
pdadmin> pop modify test set ipauth 9.1.2.3 255.255.255.0 forbidden
```

- 前の `pop modify set ipauth add` コマンドによって IP アドレスが使用可能にされていない限り、すべての IP アドレスからの要求を保護オブジェクトにアクセスさせないようにする。

例えば、上記のケースで、一定の範囲の IP アドレスは、次のようにして、認証強度レベル 1 を使用して保護リソースにアクセスすることが必要でした。

```
pdadmin> pop modify test set ipauth add 9.1.2.3 255.255.255.0 1
```

これに加えて、アドミニストレーターは、次の **pdadmin** コマンドで、認証強度レベルに関係なく、その他のすべての IP アドレスからの要求を拒否するように指定できます。

```
pdadmin> pop modify test set ipauth anyothernw forbidden
```

オプション **anyothernw** は、その他のすべてのネットワーク・アドレス (*any other network address*) を意味し、オプション **forbidden** は、拒否ポリシーを実行します。

保護リソースへの保護オブジェクト・ポリシーの付加

このタスクについて

保護オブジェクト・ポリシー (POP) を定義し作成したら、その POP が適用される保護リソースに付加しなければなりません。POP を付加する構文は次のようになります。

```
pdadmin pop attach object_name pop_name
```

例えば、WebSEAL デプロイメントの認証ポリシーは、以下のように定義することができます。

- デプロイメントは、フォーム認証と証明書認証を使用する。フォーム認証は最初の認証強度レベル (1) とし、証明書認証は 2 番目の (強度がより強い) 認証レベル 1 (2) とする。
- ユーザーは、下記の保護リソース (WebSEAL junction) にアクセスするには、フォーム認証以上の認証を使用して認証を受けなければならない。

```
/WebSEAL/hostA/junction
```

- ユーザーは、下記の保護リソース (アプリケーション) にアクセスするには、証明書認証を使用して認証を受けなければならない。

```
/WebSEAL/hostA/junction/applicationA
```

このポリシーをインプリメントするには、次の構成ステップを実行する必要があります。

手順

1. WebSEAL 構成ファイルを変更して、フォーム認証に認証強度 1 を認可し、さらに、証明書認証に強度 2 を認可します。

```
[authentication-levels]
level = unauthenticated
level = password
level = ssl
```

2. 認証レベル 1 (フォーム認証) 用の POP を作成します。

```
pdadmin> pop create test1
pdadmin> pop modify test1 set ipauth anyothernw 1
```

3. 認証レベル 2 (証明書認証) 用の POP を作成します。

```
pdadmin> pop create test2
pdadmin> pop modify test2 set ipauth anyothernw 2
```

4. POP test1 を /WebSEAL/hostA/junction に付加します。

```
pdadmin> pop attach /WebSEAL/hostA/junction test1
```

5. POP test2 を /WebSEAL/hostA/junction/application に付加します。

```
pdadmin> pop attach /WebSEAL/hostA/junction/applicationA test2
```

タスクの結果

注: POP の管理について詳しくは、「*IBM Security Access Manager for Web 管理ガイド*」を参照してください。**pdadmin** 構文については、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

複数の認証レベル間でのユーザー ID マッチングの実行

このタスクについて

WebSEAL はデフォルトで、認証強度 (ステップアップ) 操作を実行するユーザー ID が、最初の認証操作を実行するのに使用されるユーザー ID と一致することを必要とします。

WebSEAL は、新規のユーザー資格情報の中のユーザー ID が、元の資格情報の中のユーザー ID に一致しているか確認します。2 つのユーザー ID が一致しない場合、WebSEAL は、認証ステップアップを拒否し、エラーをログに記録し、ユーザーにエラー・ページを戻します。

手順

この機能はデフォルトで使用可能になっています。

この機能を使用不可にするには、WebSEAL 構成ファイルを編集し、次のように **verify-step-up-user** の値を **no** に設定します。

```
[step-up]
verify-step-up-user = yes
```

非認証ユーザーへのログイン応答の制御

このタスクについて

ステップアップ認証 POP 属性によって保護されているオブジェクトを要求する非認証ユーザーへのログイン・プロンプト応答を制御することができます。

デフォルトでは、WebSEAL は POP で必要な特定の認証レベルのログイン・プロンプトのみを表示します。WebSEAL 構成ファイルの **[step-up]** スタンザにある **show-all-auth-prompts** スタンザ・エントリーがこの応答を制御します。デフォルト値は「no」です。

```
[step-up]
show-all-auth-prompts = no
```

手順

前バージョンの WebSEAL では、非認証ユーザーに対して、1 つのログイン・ページで、複数のログイン・プロンプト (使用可能な認証方式ごとに 1 つ) が提示されていました。この前バージョンの動作をサポートするには、次のように、**show-all-auth-prompts** スタンザ・エントリーの値を「yes」に設定します。

```
[step-up]
show-all-auth-prompts = yes
```

注: **show-all-auth-prompts** 機能は、オブジェクト上の POP によってのみ起動されます。非認証ユーザーが、オブジェクト上に POP がないという理由で認証を求められた場合、**show-all-auth-prompts** の機能は使用されません。

より高いレベルへの認証のステップアップ

このタスクについて

POP で指定されたレベルより高いレベルで構成された認証メカニズムを受け入れるように WebSEAL を構成することができます。この構成では、ユーザーが直接高いレベルで認証できます。

手順

- ステップアップ操作中に上位の認証レベルを受け入れるには、**step-up-at-higher-level** スタンザ・エントリーの値を「yes」に設定する必要があります。

```
[step-up]
step-up-at-higher-level = yes
```

- ステップアップ操作中に上位の認証レベルを却下するには、**step-up-at-higher-level** スタンザ・エントリーの値を「no」に設定します。

```
[step-up]
step-up-at-higher-level = no
```

タスクの結果

この構成エントリーを構成しない場合のデフォルト値は「no」です。つまりデフォルトでは、WebSEAL は上位の認証レベルを受け入れません。

外部認証インターフェース

Security Access Manager は、WebSEAL 認証プロセス機能の拡張を可能にする外部認証インターフェースを提供します。外部認証インターフェースによって、サーブド・パーティー・システムは、WebSEAL および Web サーバー・プラグインに認証済み ID を提供できます。その後、ID 情報は資格情報の生成に使用されます。

この拡張認証機能は、Web セキュリティー外部認証 C API が提供する既存のカスタム認証モジュール機能と似ています。ただし、外部認証インターフェースを使用すると、認証モジュール・インターフェース経由ではなく HTTP 応答ヘッダーでユーザー ID を提供することができます。

外部認証インターフェースの認証の構成については、263 ページの『第 13 章 外部認証インターフェース』を参照してください。

クライアント証明書ユーザー・マッピング

このセクションでは、クライアント証明書ユーザー・マッピング拡張機能の概要を示し、主要な概念およびコンポーネントについて説明し、この機能を Security Access Manager 環境内にデプロイする方法の詳細を示します。

概要

WebSEAL は、クロスドメイン認証サービス (CDAS) を使用してユーザーを認証し、Security Access Manager ユーザー ID を提供します。

クライアント証明書ユーザー・マッピング CDAS は、WebSEAL がクライアント証明書の詳細情報を使用できるようにするためのメカニズムで、これにより WebSEAL は、対応する Security Access Manager ユーザー ID を判別することができます。クライアント証明書のマッピングを管理するルールは、XSL スタイルの表記で定義されます。

CDAS は、Security Access Manager がサポートするすべてのユーザー・レジストリーをサポートします。

ルール評価により、LDAP 検索ストリングを返すことができます。LDAP 検索フィルターのストリング表記は、RFC 2254 に記載されている形式に一致する必要があります。

ルールの例

新しい CDAS によって、証明書に含まれている属性を Security Access Manager ユーザー ID にマッピングする際の柔軟性がさらに高まります。

次のリストは、この CDAS によってサポートされるマッピング機能の一部を示しています。

1. 発行者 DN が X で、サブジェクト DN が Y なら、Security Access Manager DN も Y。

2. 証明書自体は **inetOrgPerson** エントリーの **userCertificate** 属性に保存され、ユーザー・レジストリー内の Base64 エンコード・バージョンの証明書について検索が実行されます。
3. 証明書から発行者 DN およびサブジェクト DN を取得し、これらを以下に示すように結合します。

```
<certDN>subjectName</certDN><issuerDN>issuerName</issuerDN>
```

次に、この値を持つエントリーを次の属性から検索します。

```
ibm-certificateSubjectAndIssuer
```

4. 発行者 DN が X の場合、**subjectAltName** は **inetOrgPerson** エントリーの DN と同じです。
5. 発行者 DN が X の場合、**serialNumber** は **inetOrgPerson** の **secCertSerialNumber** 属性にマップします。
6. 発行者 DN が X の場合、**subjectDN** フィールドの cn は、**inetOrgPerson** エントリーの cn にマップします。
7. 発行者 DN が X の場合、**subjectDN** は **inetOrgPerson** エントリーの **secCertDN** にマップします。

証明書ユーザー・マッピング・ルールの言語

Extensible Style Language (XSL) は、ルールを指定するために使用する言語で、Extensible Markup Language (XML) は、ルールへの入力を形成するデータに使用する言語です。XML と XSL を組み合わせることにより、ルール評価プログラムへの入力およびルール自体の両方を表現する、プラットフォームに依存しない方法が提供されます。

XML には、テキスト形式で、構造化された標準方法で複素数データ・タイプを表現する機能も用意されています。このテキスト形式により、XML データを処理するためのルールを、プラットフォームおよびプログラミング言語特性を考慮する必要なく書き込むことができます。

XSL は、ユーザーの必要性に応じて単純または複雑なタスクを実行するために使用できる機能スタイルシート言語です。XSL は、XML データを分析して評価する継承機能を所有し、これはデータ表記の標準になりつつあります。XSL は、XPath などの他の XSL 系の標準を基礎として構築され、証明書ユーザー・マッピング・ルールの中核をなす式言語です。

ユーザー・マッピング・ルールをインプリメントするには、XSL ルールに多くの制約を課す必要があります。この制約には、ルール評価の出力が単純テキストであり、その出力は結果ストリングの既知のセットのいずれかに準拠するという要件などがあります。ユーザー・マッピング・ルールの形式および制約の詳細については、218 ページの『ルールの形式および制約』を参照してください。

UMI XML 文書モデル

Universal Management Infrastructure XML 文書モデル (または UMI XML モデル) は、ユーザー・マッピング・ルールのインプリメンテーションによって XSL/XML モデルに加える制約のセットです。この制約によって、インターフェースは、単純でありながらも証明書目的の機能を持つことができます。このモデルでは、証明書

ルールが、事前定義された XML 文書形式内で機能するよう制約を課します。この形式では、最上位の XML 文書要素はすべてのルールについて同じです。ルール評価プログラムによって証明書属性からインポートされる XML UMI は、証明書によってそのデータが使用できるようになる前にこの XML 文書に挿入される必要があります。同様に、ルールの定義プロセスを単純化するには、証明書ルールは UMI XML モデルの範囲内で運用される必要があります。

UMI XML モデルでは、XML 文書に以下の最上位 XML エlementが含まれている必要があります、このElementの内部に、特定のルール評価のためのすべてのターゲット UMI が挿入されます。この XMLUMI Elementは、ルール評価プロセスの一部としてユーザー・マッピング・エンジンによって自動的に作成されます。

```
<XMLUMI>
<!--XML formatted UMI are inserted here.    -->
</XMLUMI>
```

この制限の結果、証明書ユーザー・マッピング・ルールで使用するデータへの XPath には、モデル内の特定のデータ・Elementにアクセスするために、接頭部 /XMLUMI が含まれている必要があります。例えば、UMI 項目 stsuuser:STSUniversalUser が文書に追加される場合、XML オブジェクト stsuuser:STSUniversalUser 内に含まれるデータにアクセスするには、XPath /XMLUMI/stsuuser:STSUniversalUser を指定する必要があります。

XPath は、構造化 XML データ・オブジェクトの階層内にある特定の子Elementへのパスです。これは、ハード・ディスクのディレクトリー・パスが特定のファイルへのアクセスに使用されるのと同じです。XPath 指定は文書のルート（この場合は /XMLUMI）から始まって、このルートからその子Elementを通り、参照されている特定のElementまでトレースします。例えば、216 ページの『XML 証明書モデル』内の資格付与の例 stsuuser:STSUniversalUser を参照として使用するとき、次の XPath を使用して /XMLUMI/stsuuser:STSUniversalUser の Version Elementにアクセスします。

```
"/XMLUMI/stsuuser:STSUniversalUser/stsuuser:AttributeList/stsuuser:
Attribute[@name='Version']/stsuuser:Value"
```

この例のような XPath は、属性ベースのユーザー・マッピングの判断を下すのに必要な UMI データ値にユーザー・マッピング・ルールがアクセスするための方法です。

すべてのデータ・Elementは UMI XML モデルの範囲内で機能するように制限されているため、ユーザー・マッピング・ルールも、モデル内の XPath に基づいて運用するか、XPath に一致するように制限する必要があります。したがって、XSL テンプレート突き合わせステートメントも、UMI XML 文書内の /XMLUMI から始まる XPath を突き合わせるように制限されます。追加情報については、218 ページの『ルールの形式および制約』を参照してください。

コンテナおよび XML UMI コンテナ名

データがリソース・マネージャーから要求されるとき、戻されたすべての XML データの細分性は、情報の単一コンテナのレベルにあります。コンテナは、通常最小のデータ・Elementでもあります（例えば、請求目的と考えられるElement）。この規則は UMI XML モデルにも適用されます。ユーザー・マッピング・ル

ール内で使用される UMI は、XML データのコンテナとしても定義および操作されます。例えば、『XML 証明書モデル』で定義されている `stsuuser:STSUniversalUser` XML オブジェクトは UMI コンテナの一例です。

UMI の項目の定義内の最上位エレメントは、UMI の項目のコンテナ名 として参照されます。証明書ユーザー・マッピング・ルールを定義するとき、任意の UMI コンテナ内のデータの XML 定義への XPath は、データ・エレメントの XPath 指定内で `/XMLUMI` に続く最初のエレメントとして、コンテナの名前を使用して常に参照される必要があります。

ここで、UMI 項目 `stsuuser:STSUniversalUser` の例に戻ります。
`stsuuser:STSUniversalUser` コンテナ内のエレメントにアクセスするには、XPath 指定に接頭部 `stsuuser:STSUniversalUser` を付ける必要があります。例えば、
`"/stsuuser:STSUniversalUser/stsuuser:AttributeList/
stsuuser:Attribute[@name='SerialNumber']/stsuuser:Value"` は `SerialNumber` 値を示します。証明書ユーザー・マッピング・ルール内でこの情報にアクセスするには、この XPath は XML ターゲット UMI 入力文書の最上位エレメント、つまり `XMLUMI` も接頭部として付ける必要があります (例: `"/XMLUMI/
stsuuser:STSUniversalUser/stsuuser:AttributeList/
stsuuser:Attribute[@name='SerialNumber']/stsuuser:Value"`)。

テンプレート突き合わせステートメントを使用して、パス全体を完全に指定する必要性をなくすることができます。ルール・ファイルの例には、テンプレート突き合わせステートメント `/XMLUMI/stsuuser:STSUniversalUser/stsuuser:AttributeList` が含まれています。これは、証明書のすべての属性が接頭部を付けずに指定可能であることを意味します。例えば、`"/XMLUMI/stsuuser:STSUniversalUser/
stsuuser:AttributeList/stsuuser:Attribute[@name='SerialNumber']/
stsuuser:Value"` は `"stsuuser:Attribute[@name='SerialNumber']/
stsuuser:Value"` と同じです。追加情報については、218 ページの『ルールの形式および制約』を参照してください。

XML 証明書モデル

次の UMI XML 文書は、証明書ユーザー・マッピング・ルールの評価中にルール評価プログラムから XSL プロセッサに渡されるデータを示しています。

この文書には、`stsuuser` という 1 つのコンテナが含まれています。コンテナ `stsuuser:STSUniversalUser` の属性値は XML で定義されます。

証明書評価プログラムは、UMI XML 文書が作成されるときに、自動的にすべてのデータを XML 最上位ノード宣言 `XMLUMI` の下に含めます。つまり、この最上位レベル・エレメントは明確にするために追加されます。

XML 文書は、クライアント証明書内部で使用可能な属性に基づき、CDAS によって自動的に作成されます。ユーザー・マッピング・ルール評価プログラムによって鑑定ルーチンに渡される XML 文書は、次のとおりです。

```
<XMLUMI>
<stsuuser:STSUniversalUser xmlns:stsuuser="urn:ibm:names:ITFIM:1.0:stsuuser">
<stsuuser:Principal>
<stsuuser:Attribute name="name">
<stsuuser:Value>
```



```
-- Subject DN from certificate --
</stsuser:Value>
</stsuser:Attribute>
</stsuser:Principal>
</stsuser:AttributeList>
<stsuser:Attribute name="--attr-name--" type="urn:ibm:security:gskit">
<stsuser:Value>--attr-value--</stsuser:Value>
</stsuser:Attribute>
...
</stsuser:AttributeList>
</stsuser:STSUniversalUser>
</XMLUMI>
```

以下に例を示します。

```
<?xml version="1.0" encoding='UTF-8'?>

<XMLUMI>
<stsuser:STSUniversalUser xmlns:stsuser="urn:ibm:names:ITFIM:1.0:stsuser">
<stsuser:Principal>
<stsuser:Attribute name="name">
<stsuser:Value>
CN=testuser,O=ibm,C=au
</stsuser:Value>
</stsuser:Attribute>
</stsuser:Principal>
</stsuser:AttributeList>
<stsuser:Attribute name="SubjectDN" type="urn:ibm:security:gskit">
<stsuser:Value>CN=testuser,O=ibm,C=au</stsuser:Value>
</stsuser:Attribute>

<stsuser:Attribute name="IssuerDN" type="urn:ibm:security:gskit">
<stsuser:Value>CN=ca,O=ibm,C=au</stsuser:Value>
</stsuser:Attribute>

<stsuser:Attribute name="ValidFromEx" type="urn:ibm:security:gskit">
<stsuser:Value>00:29:26 08-06-2009</stsuser:Value>
</stsuser:Attribute>

</stsuser:AttributeList>
</stsuser:STSUniversalUser>
</XMLUMI>
```

使用可能な属性の完全なリストについては、222 ページの『有効な証明書属性』を参照してください。

ルールに対して使用可能な XMLUMI 文書内の特定の UMI 項目を参照するとき、XPath パス指定子は XML エレメントのコンテナ名 (例えば stsuser:STSUniversalUser) から開始できます。呼び出し元は、明示的に独自のテンプレート突き合わせステートメントを指定したい場合は、そうすることができます。

追加情報については、218 ページの『ルールの形式および制約』を参照してください。

ユーザー・マッピング・ルール評価プログラム

ユーザー・マッピング・ルール評価プログラムは、ユーザー・マッピング・エンジンが必要とする制約内でユーザー・マッピング・ルールを評価します。事前構成済みのルールは、構成ファイルで新しい CDAS に提供されます。

ユーザー・マッピング・ルール評価プログラムは、ルール・ポリシーを証明書の XML 表記と一緒に取得し、評価のためにこれを XSL プロセッサに渡します。

変換の入力は、クライアント証明書の XML バージョン (上に定義したもの) です。XSL 変換ルールによって、提供された証明書情報から Security Access Manager ユーザー名をマッピングする方法が決まります。判断を下す際には以下の 2 つの入力が使用されます。

- クライアント証明書の XML 表記
- XML を解釈する方法を判別する XSL ルール

判断からの出力は、Security Access Manager ユーザー ID の判別に使用する単一ストリングです。

ユーザー・マッピング・エンジンでは、ルール評価の結果、以下にリストされているストリング ID のいずれかが戻されることを想定しています。これらの ID は、XSL ルールが不正に書き込まれ、評価によって不正な情報が戻される場合に、固有であることを保証します。ID を感嘆符 (!) を使用して明確に記述すると、評価プログラムは誤った事例を認識できます。

ストリングは以下のいずれかの定義に準拠する必要があります。

!free format text!

フリー・フォーマット・テキストで、ソース XML のエレメントを含むこともできます。このストリングは Security Access Manager ユーザー ID として使用されます。例えば、次のとおりです。

```
!cn=testuser,o=ibm,c=au!  
!<xsl:value-of select="stsuser:Attribute[@name='SerialNumber']"/  
stsuser:Value"/>!
```

!userreg base='%base%' attr='%name%'!%ldap-search-filter%!

提供された検索ストリングに基づいて、ユーザー・レジストリーで Security Access Manager ユーザー ID を検索することを示します。**attr** 値を使用して、Security Access Manager ユーザー ID を保持する LDAP 属性の名前を定義します。検索ストリングは RFC 2254 に準拠する必要があります。例:

```
!userreg base='o=ibm,c=au' attr='cn'! (&(objectClass=ePerson)  
(serialNum=<xsl:value-of select="stsuser:Attribute[@name=  
'SerialNumber']/stsuser:Value"/>))!
```

!no-matching-rule!

提供されたクライアント証明書に対して一致するルールが検出されなかったことを示します。ルール評価からこのストリングが戻されると、CDAS はエラーを戻します。例えば、次のとおりです。

```
!no-matching-rule!
```

注: アンパーサンド (&) 文字は処理中にエラーが生成されるため、XSLT 文書では使用できません。この文字は **&** に書き換える必要があります。

ルールの形式および制約

証明書ユーザー・マッピング・ルールは XSL スタイルシート内に XSL テンプレートとして定義する必要があります。このルールは有効な XSL テンプレート・ルー

ル形式で記述する必要があります。ルールは、217 ページの『ユーザー・マッピング・ルール評価プログラム』に示すいずれかのストリング ID を含むテキスト文書に戻す必要があります。

これらの ID は、出力文書内で唯一のテキストでなければなりませんが、空白文字で囲むことができます。定義済みの値以外の値または空の文書が戻される場合、ユーザー・マッピングが失敗し、ルールが準拠していないことを示すエラー・コードが CDAS に戻されます。

XSL 証明書ユーザー・マッピング・ルールによって実行された XSL 変換の結果は、サポートされるストリング ID のいずれか 1 つだけを含むテキスト出力文書である必要があります。

以下の証明書ユーザー・マッピング・ルールの例は、stsuser:STSUniversalUser に定義された XML データ項目を参照します。ルールが評価する条件は、次のように表現されます。

```
<?xml version="1.0" encoding='UTF-8'?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:stsuser="urn:ibm:names:ITFIM:1.0:stsuser" version="1.0">

  <!-- Required to constrain output of rule evaluation -->
  <xsl:output method="text" omit-xml-declaration="yes" encoding='UTF=8' indent="no"/>

  <!-- Need this to ensure default text node printing is off -->
  <xsl:template match="text()"></xsl:template>

  <!-- Let's make it easier by matching the constant part of our XML name -->
  <xsl:template match="/XMLUMI/stsuser:STSUniversalUser/stsuser:AttributeList">

    <!-- If this certificate was issued by our CA, just use the subject dn -->
    <xsl:when test='stsuser:Attribute[@name="IssuerDN"]/stsuser:Value = "cn=ca,o=ibm,c=au"'>
      <xsl:value-of select="stsuser:Attribute[@name='SubjectDN']/stsuser:Value"/>
    </xsl:when>

    <!-- If this certificate was issued by the tivoli CA, search for the certificate serial number -->
    <xsl:when test='stsuser:Attribute[@name="IssuerDN"]/stsuser:Value = "cn=ca,o=tivoli,c=au"'>
      !userreg base='o=ibm,c=us' attr='cn'!secCertSerialNumber=<xsl:value-of select="stsuser:Attribute[@name='SerialNumber']/stsuser:Value"/>
    </xsl:when>

    <!-- Otherwise we don't have a matching rule.'no-matching-rule' is a special string. -->
    <xsl:otherwise>
      !no-matching-rule!
    </xsl:otherwise>

  </xsl:template>

</xsl:stylesheet>
```

注: template match より前のすべての部分は、すべてのルールについて固定です。残りの部分の XSL ルールはカスタマイズできます。

文書内のデータ項目を参照するには、各ノードへの XPath に XMLUMI ノードを含める必要があります。ルールを作成するとき、ルール作成者は、XML データ・ノードおよびサブノードにアクセスするために、ツリー内の現行ポイントからの正しい XPath を理解しておく必要があります。ツリー内の現行ポイントは、テンプレート突き合わせステートメントを使用することによって選択されます。テンプレート突き合わせステートメントを使用すると、XSL プログラマーは、XPath 処理が XML 文書ツリーのかなり下方で発生されるように指定することによって、各データ・エレメントへの XPath を短くできます。

```
<xsl:template match="/XMLUMI/stsuuser:STSUniversalUser/
stsuuser:AttributeList"> ステートメントは、テンプレート・ステートメントの境
界範囲内のすべての相対 XPath が、ノード "/XMLUMI/stsuuser:STSUniversalUser/
stsuuser:AttributeList" に対して相対的だと推定されることを XSL プロセッサ
ーに通知します。例えば、"/XMLUMI/stsuuser:STSUniversalUser/
stsuuser:AttributeList/stsuuser:Attribute[@name='IssuerDN']/stsuuser:Value"
は単に "stsuuser:Attribute[@name='IssuerDN']/stsuuser:Value" として参照でき
ます。
```

ユーザー・マッピング・ルールの例

このセクションでは、出力 XSLT 評価の 2 つの例を示します。最初の例はフリー・フォーマットのテキスト・ストリングを使用し、次の例ではユーザー・レジストリーでユーザー DN を検索します。

最初の例では、ルールはクライアント証明書からのストリングを処理します。ストリングが一致すると、クライアント証明書から識別名 (DN) を選択します。

ルール:

```
<!-- Test a valid 'free format' string. -->
<xsl:when test='stsuuser:Attribute[@name="SubjectDN"]
/stsuuser:Value = "cn=testuser,o=ibm,c=au"'>
!<xsl:value-of select="stsuuser:Attribute[@name='SubjectDN']/
stsuuser:Value"/>!
</xsl:when>
```

クライアント証明書の詳細

```
<stsuuser:STSUniversalUser xmlns:stsuuser=
"urn:ibm:names:ITFIM:1.0:stsuuser">
<stsuuser:Principal>
<stsuuser:Attribute name="name">
<stsuuser:Value>
CN=testuser,O=ibm,C=au
</stsuuser:Value>
</stsuuser:Attribute>
</stsuuser:Principal>
<stsuuser:AttributeList>
<stsuuser:Attribute name="SubjectDN" type=
"urn:ibm:security:gskit">
<stsuuser:Value>CN=testuser,O=ibm,C=au</stsuuser:Value>
</stsuuser:Attribute>
...
</stsuuser:AttributeList>
</stsuuser:STSUniversalUser>
```

CDAS から戻されたストリング:

```
CN=testuser,O=ibm,C=au
```

2 番目の例では、ルールはユーザー・レジストリーを検索して属性を探し、レジストリーからユーザー名 (CN) を戻します。この事例でのレジストリーの検索は、クライアント証明書からの E メール・アドレスに関するものです。

ルール:

```
<!-- Test a matching 'userreg' string. -->
<xsl:when test='stsuuser:Attribute[@name="SubjectDN"]
/stsuuser:Value = "cn=testuser3,o=ibm,c=au">
!userreg base='o=ibm,c=au' attr='cn'!(description=<xsl:value-of
select="stsuuser:Attribute[@name='SubjectEmail']"/
stsuuser:Value"/>)!
</xsl:when>
```

クライアント証明書の詳細

```
<stsuuser:STSUniversalUser xmlns:stsuuser=
"urn:ibm:names:ITFIM:1.0:stsuuser">
<stsuuser:Principal>
<stsuuser:Attribute name="name">
<stsuuser:Value>
cn=testuser3,o=ibm,c=au
</stsuuser:Value>
</stsuuser:Attribute>
</stsuuser:Principal>
<stsuuser:AttributeList>
<stsuuser:Attribute name="SubjectDN" type=
"urn:ibm:security:gskit">
<stsuuser:Value>cn=testuser3,o=ibm,c=au</stsuuser:Value>
</stsuuser:Attribute>
<stsuuser:Attribute name="SubjectEmail" type=
"urn:ibm:security:gskit">
<stsuuser:Value>testuser3@ibm.com</stsuuser:Value>
</stsuuser:Attribute>
...
</stsuuser:AttributeList>
</stsuuser:STSUniversalUser>
```

ユーザー・レジストリーから戻されたストリング

```
cn=testuser3
```

CDAS の管理方法

このセクションでは、CDAS を WebSEAL に追加する方法および CDAS を構成する方法について説明します。

CDAS 機能の使用可能化

拡張された CDAS 機能を使用可能にすることができます。

このタスクについて

拡張された CDAS 機能を使用可能にするには、次のようにします。

手順

1. WebSEAL 構成ファイルの **[cert-map-authn]** スタンザを以下のように更新する必要があります。

```
[cert-map-authn]
rules-file = file
debug-level = level
```

ここで、

file 使用する証明書マッピング CDAS のルール・ファイルの名前。

level モジュールのトレース・レベルを制御します。

例:

```
[cert-map-authn]
rules-file = cert-rules.txt
debug-level = 5
```

注: *level* 変数はトレース・レベルを表し、1 が最小のトレース量、9 が最大のトレース量を指定します。また、Security Access Manager **pdadmin** トレース・コマンドを使用し、**pd.cas.certmap** というトレース・コンポーネント名を使用することでトレース・レベルを変更することもできます。このトレース・コンポーネントは、最初の HTTP 要求が処理された後にのみ利用できます。

2. 以下のようにローカル管理インターフェース (LMI) を使用して、必要に応じてルール・ファイル (**cert-rules.txt** など) を変更することができます。
 - a. トップ・メニューから「**セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)**」>「**グローバル設定**」>「**クライアント証明書マッピング**」を選択します。「クライアント証明書マッピング」管理ページが表示されます。
 - b. (オプション) ルール・ファイルが存在しない場合は、「**新規**」をクリックして新規ルール・ファイルを作成できます。新規ファイルの名前 (**cert-rules.txt** など) を入力して「**保存**」をクリックします。デフォルト・テンプレートに基づく新しいファイルが生成されます。
 - c. 使用可能な「ファイル名」のリストから、管理するファイル (**cert-rules.txt** など) をクリックします。
 - d. 「**編集**」をクリックします。
 - e. ファイルを更新します。
 - f. 「**保存**」をクリックします。

有効な証明書属性

マッピング・ルールに使用可能なクライアント証明書属性は、GSKit ツールキットで定義されています。

WebSEAL の場合、以下のいずれかの属性とすることができます。

- * Base64Certificate
- * SerialNumber
- * SubjectCN
- * SubjectLocality
- * SubjectState
- * SubjectCountry
- * SubjectOrganization
- * SubjectOrganizationalUnit
- * SubjectDN
- * SubjectPostalCode
- * SubjectEmail
- * SubjectUniqueID
- * IssuerCN
- * IssuerLocality
- * IssuerState
- * IssuerCountry
- * IssuerOrganization

- * IssuerOrganizationUnit
- * IssuerDN
- * IssuerPostalCode
- * IssuerEmail
- * IssuerUniqueID
- * Version
- * SignatureAlgorithm
- * ValidFrom
- * ValidFromEx
- * ValidTo
- * ValidToEx
- * PublicKeyAlgorithm
- * PublicKey
- * PublicKeySize
- * FingerprintAlgorithm
- * Fingerprint
- * BasicConstraintsCA
- * BasicConstraintsPathLength
- * DerCertificate
- * CertificatePolicyID
- * CRLDistributionPoints
- * DerSubjectDN
- * DerIssuerDN
- * KeyUsage
- * AlternativeDirectoryName
- * AlternativeDNSName
- * AlternativeIPAddress
- * AlternativeURI
- * AlternativeEmail

GSKit について詳しくは、「IBM Secure Sockets Layer の入門および iKeyman ユーザーズ・ガイド」を参照してください。

証明書マッピング・モジュールを使用するための WebSEAL の構成

始める前に

以下のファイル・パスを想定します。

- \$WEBSEAL_HOME は WebSEAL インストール・ディレクトリーを指します。通常のファイル・パスは /opt/pdweb です。
- \$WEBRTE_HOME は、Security Access Manager Web Security Runtime 環境のインストール・ディレクトリーを指します。通常のファイル・パスは /opt/pdwebрте です。

UNIX および Windows プラットフォームの場合の通常のファイル・パスは以下のとおりです。

– UNIX

/opt/pdwebрте

– Windows

C:\Program Files\Tivoli\PDWebRTE

手順

1. WebSEAL インスタンス構成ファイルを編集して、認証モジュールを使用可能にします。このファイルは、\$WEBSEAL_HOME/etc/webseald-instance.conf にあり

ます。ここで、*instance* は構成対象 WebSEAL インスタンスの名前です。
[authentication-mechanisms] スタンザで、以下の例に示すように *cert-ssl* エントリーを変更します。

```
cert-ssl = library-name & rule=rule-file [debug=level]
```

ここで、

- *library-name* はモジュールの共有ライブラリーへのパスであり、これは `$WEBRTE_HOME/lib/` にあります。形式は `[lib-prefix]amwcertmapauthn[lib-suffix]` です。ここで、`[lib-prefix]` および `[lib-suffix]` はプラットフォームによって異なります。
- *rule-file* は XSLT マッピング・ファイルへのパスです。カスタマイズ可能なテンプレートは `$WEBRTE_HOME/etc/cert-rules-template.txt` に入っています。ユーザーのニーズに合わせて変更することができます。
- *level* は、モジュールのトレース・レベルを定義するオプション・パラメーターです。

例えば、以下のようにエンTRIESを (1 行で) 定義することができます。

```
cert-ssl = /opt/pdwebрте/lib/libamwcertmapauthn.so & rule=/opt/pdwebрте/etc/cert-rules.txt debug=5
```

2. [certificate] スタンザ内の *accept-client-certs* エントリーを確認します。このエンTRIESの値は *required*、*optional*、または *prompt_as_needed* でなければなりません。これにより、クライアント証明書を要求するプロンプトがブラウザーによってユーザーに表示されるかどうか、また、いつ表示されるかが決定されます。

このエンTRIESの他の値について詳しくは、IBM Security Access Manager for Web インフォメーション・センター (http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.isam.doc_70/welcome/html) の 164 ページの『クライアント・サード証明書認証モード』を参照してください。

3. 構成ファイルを保存します。
4. WebSEAL を再始動して更新を実装します。

XSLT ルール・ファイルの構成

マッピング・モジュールを使用して、ユーザー ID への証明書属性のマッピングを可能にする柔軟なルールを定義することができます。ユーザー ID として、Security Access Manager ユーザー ID を使用できます。例えば、*testuser* などです。あるいは、ユーザー ID として、レジストリー内のユーザー DN を使用することもできます。例えば、*cn=testuser, o=ibm, c=au* などです。

ユーザー証明書を受け取ると、モジュールはその属性をすべてリストする XML 文書を作成します。XML 文書は、Universal Management Infrastructure (UMI) XML 文書モデルに準拠しています。例えば、モジュールは以下のような文書を作成できます。

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<XMLUMI>
```

```
<stsuser:STSUniversalUser xmlns:stsuser="urn:ibm:names:ITFIM:1.0:stsuser">  
  <stsuser:Principal>
```



```

<stsuser:Attribute name="name">
  <stsuser:Value>
    CN=testuser,O=ibm,C=au
  </stsuser:Value>
</stsuser:Attribute>
</stsuser:Principal>
<stsuser:AttributeList>
  <stsuser:Attribute name="SubjectDN" type="urn:ibm:security:gskit">
    <stsuser:Value>CN=testuser,O=ibm,C=au</stsuser:Value>
  </stsuser:Attribute>

  <stsuser:Attribute name="IssuerDN" type="urn:ibm:security:gskit">
    <stsuser:Value>CN=ca,O=ibm,C=au</stsuser:Value>
  </stsuser:Attribute>

  <stsuser:Attribute name="ValidFromEx" type="urn:ibm:security:gskit">
    <stsuser:Value>00:29:26 08-06-2009</stsuser:Value>
  </stsuser:Attribute>

</stsuser:AttributeList>
</stsuser:STSUniversalUser>
</XMLUMI>

```

XSLT ファイルは、XML 文書の変換方法を定義します。変換の結果は以下のいずれかの形式でなければなりません。

- *!identifier!*

Security Access Manager のユーザー ID またはユーザー DN。この形式は、ID を証明書から直接取得できる場合など、レジストリー検索を必要としない場合に使用されます。

- *!userreg base='baseDN' attr='attrName' ! ldapSearchFilter !*

baseDN は基本識別名、*attrName* はユーザー ID に対応する LDAP 属性名、*ldapSearchFilter* は LDAP 検索フィルターです。

この形式は、対応するユーザーのレジストリーを検索する際に証明書情報を使用する場合に使用されます。Active Directory でモジュールを使用することもできます。

- *!no-matching-rule!*

この形式は、ユーザー認証に必要な情報を検索する際にルールを使用できないことを示します。

以下の例は、\$WEBRTE_HOME/etc/ ディレクトリーにインストールされた cert-rules-template.txt ファイルです。

```

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:stsuuser="urn:ibm:names:ITFIM:1.0:stsuuser" version="1.0">

    <!-- Required to constrain output of rule evaluation -->
    <xsl:output method="text" omit-xml-declaration="yes" encoding='UTF=8'
indent="no"/>

    <!-- Need this to ensure default text node printing is off -->
    <xsl:template match="text()"></xsl:template>

    <!-- Let's make it easier by matching the constant part of our XML name -->
    <xsl:template match="/XMLUMI/stsuuser:STSUniversalUser/stsuuser:
AttributeList">
        !<xsl:value-of select="stsuuser:Attribute[@name='SubjectDN']/stsuuser:
Value"/>!
    </xsl:template>

</xsl:stylesheet>

```

この XSLT 文書では、認証モジュールによって作成された UMI XML 文書が変換され、! 文字の間に受信される証明書のサブジェクト DN が出力されます。例えば、!cn=testuser,o=ibm,c=au! などです。

この場合、ユーザー・レジストリー検索は行われません。この <xsl:output> エレメントは、XML 文書ではなく、テキストが変換の出力であることを示すために必要です。

この最初の <xsl:template> エレメントにより、文書内の残りのテキスト・ノードは出力にコピーされなくなります。

次の例は、証明書のデータでユーザー・レジストリー (例えば、LDAP) 検索を実行するマッピング・モジュールを指定する方法を示しています。証明書から SubjectEmail 属性の値が抽出され、このアドレスに等しい LDAP mail 属性を持つユーザーが検索されます。

検索に使用される基本 DN は o=ibm,c=au です。cn 属性の値は出力されます。この場合、ユーザー DN ではなく、Security Access Manager のユーザー ID が出力されると考えられます。

```

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:stsuuser="urn:ibm:names:ITFIM:1.0:stsuuser" version="1.0">

    <!-- Required to constrain output of rule evaluation -->
    <xsl:output method="text" omit-xml-declaration="yes" encoding='UTF=8'
indent="no"/>

    <!-- Need this to ensure default text node printing is off -->
    <xsl:template match="text()"></xsl:template>

    <!-- Let's make it easier by matching the constant part of our XML name -->
    <xsl:template match="/XMLUMI/stsuuser:STSUniversalUser/stsuuser:
AttributeList">
        !userreg base='o=ibm,c=au' attr='cn'!(mail=<xsl:value-of
select="stsuuser:Attribute[@name='SubjectEmail']/stsuuser:
Value"/>)!
    </xsl:template>

</xsl:stylesheet>

```

注: & 文字はエンティティの開始を指定するために XML で使用されるため、LDAP 検索フィルタの場合のように使用することはできません。結合する必要がある複数の条件が LDAP 照会に含まれている場合は、代わりに & エンティティを使用する必要があります。

モジュールが正常にマッピングされたことの検証

ユーザーのモジュールのマッピングが正常に行われたことを確認するには、保護リソースが認証されていないユーザーを拒否し、認証されたユーザーを許可するように Security Access Manager ポリシーが設定されていることを確認します。

始める前に

ブラウザーでこのリソースにアクセスする前に、クライアント証明書がブラウザーにインポートされていることを確認します。この過程はブラウザーによって異なります。

Mozilla Firefox バージョン 3.6 を使用する場合は、以下のステップを実行します。

1. 「ツール」メニューをクリックします。
2. 「オプション」を選択します。
3. ツールバーの「詳細」をクリックします。
4. 「暗号化」タブをクリックします。
5. 「証明書を表示」をクリックします。
6. 「インポート」をクリックします。
7. 認証用に WebSEAL に送信するクライアント証明書を表すファイルを選択します。
8. ソフトウェア・セキュリティ・デバイスの新しいマスター・パスワードを作成するか、既存のパスワードを入力します。
9. クライアント証明書のパスワードを入力します。

Mozilla Firefox バージョン 10 を使用する場合は、以下のステップを実行します。

1. メニューから「オプション」を選択します。
2. ツールバーの「詳細」をクリックします。
3. 「暗号化」タブをクリックします。
4. 「証明書を表示」をクリックします。
5. 「インポート」をクリックします。
6. 認証用に WebSEAL に送信するクライアント証明書を表すファイルを選択します。

Microsoft Internet Explorer バージョン 8 を使用する場合は、以下のステップを実行します。

1. メニュー・バーの「ツール」をクリックします。
2. 「インターネット オプション」を選択します。
3. 「コンテンツ」タブを選択します。
4. 「証明書」セクションで「証明書」ボタンをクリックします。
5. 「インポート」をクリックします。

6. 「証明書のインポート ウィザード」に表示される説明に従って、クライアント証明書を含むファイルをインポートします。

手順

1. 保護リソースにアクセスしようとする、クライアント証明書を選択するためのプロンプトがブラウザによって出されます。先ほどブラウザにインポートしたクライアント証明書を選択します。 223 ページの『証明書マッピング・モジュールを使用するための WebSEAL の構成』での説明に従って WebSEAL 構成ファイルで適切にデバッグ・レベルを設定してあれば、マッピング・モジュールに関するトレース・メッセージが WebSEAL のログに書き込まれ、成功したか失敗したかが示されます。
2. XSLT 変換の結果は、マッピング・モジュールがユーザー・レジストリーを検索する必要があるかどうかを示します。結果が以下の形式になっている場合、マッピング・モジュールはユーザー・レジストリーを検索します。

```
!userreg base='baseDN' attr='attrName' ! ldapSearchFilter!
```

それ以外の場合、マッピング・モジュールはユーザー・レジストリーを検索しません。検索の結果は、Security Access Manager ユーザー ID またはユーザーの DN です。

正常にマッピングされ、ユーザー・レジストリーが検索された場合は、以下のメッセージが WebSEAL ログに出力されます。UNIX マシンの場合、WebSEAL ログは通常 /var/pdweb/log/msg_webseal-instance.log にあります。Windows マシンの場合は C:\Program Files\Tivoli\PDWeb\log\msg_webseal-instance.log にあります。

```
2012-06-07-16:37:11.113+10:00I----- thread(2) trace.pd.cas.certmap:5
/sandbox/amwebрте611/src/pdwebрте/authn/modules/certmapauthn/
AMWCertLDAPUserRegistry.cpp:146: ISAM user identity: testuser
```

レジストリーが検索されなかった場合、以下のようなメッセージのみが出力されます。

```
2012-06-07-18:34:29.200+10:00I----- thread(2) trace.pd.cas.certmap:3
/sandbox/amwebрте611/src/pdwebрте/authn/modules/certmapauthn/
AMWCertRulesEngine.cpp:219: result: CN=testuser,0=IBM,C=AU
```

第 10 章 認証後の処理

この章では、補足的な認証後処理について説明します。

この章で扱うトピックは以下のとおりです。

- 『認証後の自動リダイレクト』
- 233 ページの『サーバー・サイド要求キャッシング』

認証後の自動リダイレクト

このセクションは、以下のトピックで構成されています。

- 『自動リダイレクトの概要』
- 230 ページの『自動リダイレクトの使用可能化』
- 230 ページの『自動リダイレクトの使用不可』
- 231 ページの『制限』

自動リダイレクトの概要

ユーザーが WebSEAL ドメイン内のリソースを要求すると、通常、WebSEAL は、認証とポリシー検査が成功した時点でそのリソースをユーザーに送ります。この標準的な応答方法の代わりに、指定された特定のホーム・ページまたは「ようこそ」ページにユーザーを自動的にリダイレクトするように、WebSEAL を構成することもできます。

カスタマイズ済みのリダイレクトをさらに構成するには、ユーザーの許可レベル、ユーザー名、ホスト名などを指定する一連のマクロを使用します。

ログイン後におけるこの強制リダイレクトは、例えば、ユーザーがポータル・ページを使用して WebSEAL ドメインに入る場合などに適しています。また、自動リダイレクトは、ユーザーが、ユーザー・ブックマークを選択してドメイン内の特定ページに直接アクセスしようとする試行をオーバーライドします。

自動リダイレクトのプロセス・フローは以下のとおりです。

1. ユーザーは要求を送り、認証が成功します。
2. WebSEAL はカスタム応答を作成し、その応答を、リダイレクトとしてブラウザーに戻します。

このリダイレクト応答には、WebSEAL 構成ファイルの **login-redirect-page** スタンザ・エントリーで指定されている URL 値が含まれています。

3. ブラウザーは、リダイレクト応答 (構成された URL が入っている) の指示に従います。
4. WebSEAL は、構成された URL にあるページに戻します。

ログイン後の自動リダイレクトは、認証方式ごとに個別に使用可能または使用不可にされます。自動リダイレクトは、以下の認証方式でサポートされています。

- フォーム認証
- 基本認証
- 外部認証インターフェース

自動リダイレクトの使用可能化

このタスクについて

自動リダイレクトを構成するには、次のステップを実行します。

手順

1. 編集するために WebSEAL 構成ファイルをオープンします。
2. **[enable-redirects]** スタンザにあるそれぞれの認証方式のエントリーをアンコメントして、該当するそれぞれの認証方式の自動リダイレクトを使用可能にします。

```
[enable-redirects]
redirect = forms-auth
redirect = basic-auth
redirect = cert-auth
redirect = ext-auth-interface
```

上記の例によって、フォーム認証、基本認証、証明書認証、および EAI 認証の自動リダイレクトが使用可能になります。

3. ログインの後でユーザーがリダイレクトされる先の URL を指定します。URL は、組み込みマクロを含んでいるかどうかに関係なく、絶対 URL またはサーバー相対 URL のどちらでも指定できます。以下に例を示します。

```
[acct-mgt]
login-redirect-page = http://www.ibm.com
```

OR:

```
[acct-mgt]
login-redirect-page = /jct/intro-page.html
```

OR:

```
[acct-mgt]
login-redirect-page = /jct/intro-page.html?level=%AUTHNLEVEL%&url=%URL%
```

4. WebSEAL サーバーを停止し、再始動します。

自動リダイレクトの使用不可

このタスクについて

自動リダイレクトを使用不可にするには、次のステップを実行します。

手順

1. 編集するために WebSEAL 構成ファイルをオープンします。
2. **[enable-redirects]** スタンザにあるそれぞれの認証方式のエントリーをコメント化するか除去して、該当するそれぞれの認証方式の自動リダイレクトを使用不可にします。

```
[enable-redirects]
#redirect = forms-auth
#redirect = basic-auth
#redirect = cert-auth
#redirect = ext-auth-interface
```

各行の始めにハッシュ文字 (#) が追加されていることに注意してください。上記の例によって、フォーム認証、基本認証、証明書認証、および EAI 認証の自動リダイレクトが使用不可になります。

3. WebSEAL サーバーを停止し、再始動します。

制限

WebSEAL は、以下の条件のもとでは、ログイン時の自動リダイレクトをサポートしません。

- 再認証中。
- 基本認証を使用している間にブラウザーが再オープンされたとき。

リダイレクトは、初めてユーザーがブラウザーを使用してページを訪問し、有効なユーザー名とパスワードを使用して認証を受けるときには、期待されるとおりに作動します。しかし、ブラウザーのそのインスタンスがクローズし、別のインスタンスがオープンされた場合、ユーザーが認証された後でも、リダイレクトされたページは表示されません。

自動リダイレクト用マクロ・サポート

自動リダイレクトは、静的リダイレクト URL をカスタマイズする、WebSEAL から提供されたマクロのサブセットをサポートします。マクロは、WebSEAL からの情報の動的置換を許可します。

マクロはリダイレクト URL の照会ストリング内で引数として指定されます。マクロの値の特定文字は、URI エンコードです (232 ページの『マクロの内容のエンコード』を参照)。

自動リダイレクトに使用できる有効な WebSEAL マクロは、以下のとおりです。

マクロ	説明
AUTHNLEVEL	認証強度ポリシー (ステップアップ) で必要な認証レベル。
HOSTNAME	完全修飾ホスト名。
PROTOCOL	使用されるクライアント接続プロトコル。HTTP または HTTPS のどちらでもかまいません。
URL	クライアントによって要求される URL。
USERNAME	ログインしたユーザーの名前。ログインしていないユーザーには「unauthenticated」という値が使用されます。(200 ページの『再認証用のログイン・フォームのカスタマイズ』も参照してください。)

マクロ	説明
HTTPHDR{name}	特定の HTTP ヘッダーの内容を含めるために使用されます。指定された HTTP ヘッダーが要求内にはない場合は、マクロに Unknown というテキストが含まれます。 例えば、「Host」 HTTP ヘッダーを含めるためのマクロ名は HTTPHDR{Host} になります。
CREDATTR{name}	指定した属性の内容をユーザー資格情報に含めるために使用されます。指定された資格情報属性が要求内にはない場合は、マクロに「Unknown」というテキストが含まれます。 例えば、セッションの秘密トークンが含まれる tagvalue_session_index 属性を含めるには、マクロ名 CREDATTR{tagvalue_session_index} を使用します。

マクロの内容のエンコード

マクロの内容には、要求された URI や要求のリファラー・ヘッダーなどのユーザー提供データが含まれている場合があります。セキュリティ上の理由から、クライアント提供データ内の予約文字または特殊文字は必ずエンコードされていることが重要です。

WebSEAL URI は、クライアントに戻されるマクロの内容に予約文字または特殊文字が含まれないようにマクロの内容をエンコードします。URI エンコードは国際標準で、世界中で広く使用されている広範な文字を URI が使用する限られた文字セットにマップできます。

マクロの内容のエンコードに関する注意:

- WebSEAL は常に、マクロの内容に URI エンコードを適用します。元データが既にエンコード済みでも関係ありません。
- エンコードされたマクロの内容は、標準の URI デコード・ルールを使用してデコードする必要があります。
- URI エンコードによってマクロの内容のストリング長が長くなるため、(マクロの内容が照会ストリングとして組み込まれている) ロケーション・ヘッダーも長くなります。ロケーション・ヘッダーの長さの問題については、『マクロの内容の長さについての考慮事項』を参照してください。

マクロの内容の長さについての考慮事項

マクロが提供する情報によって、ロケーション URI ヘッダーのストリングの長さが長くなります。また、マクロの内容の URI エンコードによって、このストリングの長さはさらに長くなります。

クライアント・アプリケーションによっては (携帯電話の WAP ブラウザーなど) デバイスのメモリー容量が小さいため、URI の長さに制限がある場合があります。このようなクライアント・デバイスで URI が長さの制限を超えた場合、エラーが発生し、リンクが欠落する可能性が高くなります。

WebSEAL は、ロケーション URI ヘッダーの長さを制限しません。そこで、ローカル応答リダイレクト用のマクロを構成する場合、サイトにアクセスするクライアント・デバイスの考えられる制限事項を慎重に考慮する必要があります。URI の固定長および TAM_OP の値を決定してロケーション・ヘッダーの長さを見積もり、照会ストリング内で使用するマクロの予想サイズを予想することができます。

以下の表で、ローカル応答リダイレクトで使用されるマクロが提供する内容の可能な長さについての情報を提供します。

マクロ	内容のサイズ
AUTHNLEVEL	10 文字以内。
HOSTNAME	対応する要求の HOST ヘッダーの長さ、または HOST ヘッダーが存在しない場合の WebSEAL システムの完全修飾ホスト名。
PROTOCOL	10 文字以内。
URL	要求 URI の長さ。
USERNAME	この WebSEAL のユーザー名の長さを定めるポリシーで定義された最大長。
HTTPHDR{name}	指定された HTTP ヘッダーの長さ。
CREDATTR{name}	ユーザー資格情報に指定された属性のコンテンツの長さ。

サーバー・サイド要求キャッシング

このセクションは、以下のトピックで構成されています。

- ・ 『サーバー・サイド要求キャッシングの概念』
- ・ 234 ページの『サーバー・サイド要求キャッシングのプロセス・フロー』
- ・ 235 ページの『サーバー・サイド・キャッシングの構成』

サーバー・サイド要求キャッシングの概念

過去の WebSEAL のバージョンでは、認証が必要になるたびに、ユーザーの要求の URL 用のキャッシュ・エントリが作成されていました。そして、認証が成功すると、WebSEAL は、この URL を含む HTTP リダイレクトをブラウザーに送ります。ブラウザーは、このリダイレクトに従って、オリジナル・リソース位置にアクセスしていました。

このインプリメンテーションの制限は、例えば、POST 要求がセッション・タイムアウトにより中断され、ユーザーの再度ログインが必要になったときに明白になります。つまり、WebSEAL はオリジナルの要求の URL しかキャッシュに保管していないため、HTTP リダイレクトにより、POST データ (方式およびメッセージ本体を含む) は失われてしまいます。ユーザーは、POST 要求を再ビルドしなければなりません。

WebSEAL はより複雑な要求データ・セットをキャッシュし、HTTP リダイレクト中に要求処理が中断され、ユーザーの再ログインが必要となったときにはいつでも、このキャッシュされたデータを使用します。このソリューションは、特に POST および PUT 要求に対して大きな効果を発揮します。なぜなら、これらの要求タイプの要求には、メッセージ本体が含まれていることがあるからです。

ログイン、再認証、または認証強度 (ステップアップ) が必要なために要求処理が中断された場合はいつでも、フォーム、外部認証インターフェース、e-community シングル・サインオン、および資格情報認証方式でサーバー・サイド要求キャッシングがサポートされます。

サーバー・サイド要求キャッシングのプロセス・フロー

追加認証が必要になって要求が中断した場合、ユーザーは再度ログインするようにプロンプト表示されます。

認証の成功後、WebSEAL はブラウザーに元のリソースに関するリダイレクトを送信します。この要求を受信すると、WebSEAL はキャッシュされたデータを使用して要求を再作成し、このデータを使用して要求を処理します。

キャッシュに保管される要求データには、URL、方式、メッセージ本体、照会ストリング、およびその他のすべての HTTP ヘッダー (Cookie を含む) が含まれます。これらのデータは、WebSEAL セッション・キャッシュに一時的に保管されます。

次のダイアグラムは、一般的なサーバー・サイド要求キャッシング・プロセスのフローを示しています。

1. ユーザーは、正常にログインし、CGI 生成のデータ・フォームに関連したリソースを求める HTTP 要求をサブミットします。WebSEAL はユーザーのセッション・キャッシュ・エントリーを作成します。
2. バックエンド・アプリケーション・サーバーは、ユーザーにフォームを戻します。
3. ユーザーがこのフォームへの記入を行っている間に、このユーザー用に設定されているセッション・タイムアウト期限が切れます。WebSEAL はユーザーのキャッシュ・エントリー (資格情報も含む) およびセッション ID を削除します。
4. ユーザーは、セッション・タイムアウトに気付かずに、入力したフォームをサブミットします (POST)。WebSEAL は、ユーザーのセッション・キャッシュ・エントリーが見つからないため、新規のキャッシュ・エントリー作成します。
5. WebSEAL はこのユーザーの資格情報を見つけることができないため、ユーザーは認証を行う必要があります。WebSEAL は POST 要求に含まれている完全な情報を一時的にキャッシュし、ユーザーにログイン・フォームを送信します。
6. ユーザー入力したログイン・フォームを WebSEAL にサブミットします。そして、認証が成功します。この時点で、キャッシュには、ユーザーの資格情報と、最初にキャッシュに入れられていた要求のデータが入っています。
7. WebSEAL は、最初に要求されたリソースの URL を含むリダイレクト応答をブラウザーに戻します。
8. ブラウザーはこのリダイレクトに従います。WebSEAL はリダイレクトをインターセプトし、キャッシュされた POST データを使用して、元の要求 (CGI 生成のデータ・フォーム) を再ビルドします。復元されたフォームは、URL の宛先に送信されます。

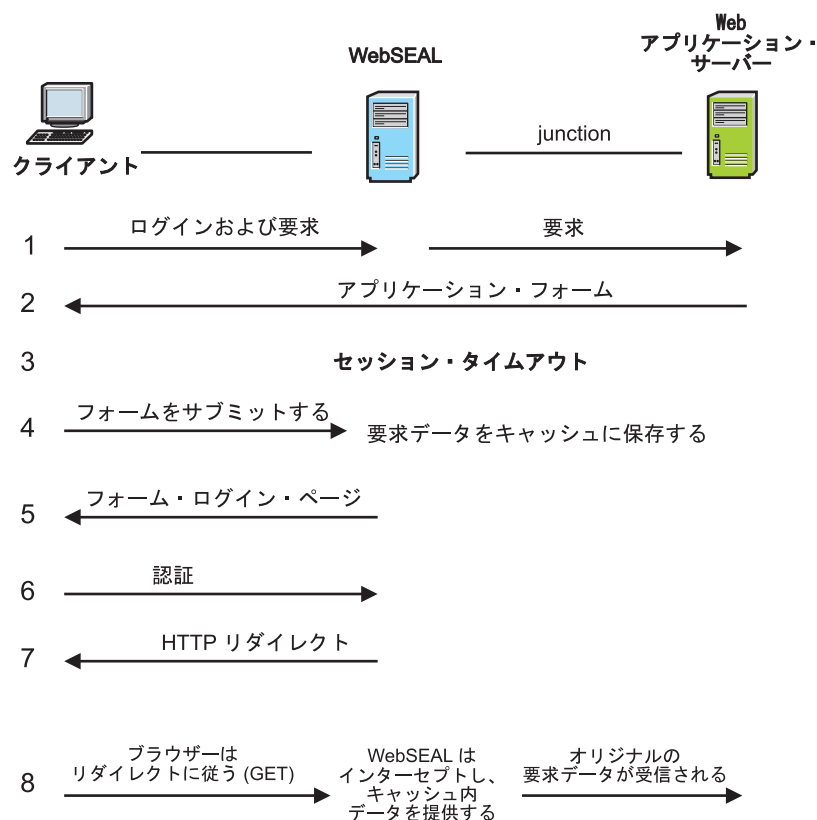


図 17. WebSEAL 要求キャッシング・プロセス・フローの例

サーバー・サイド・キャッシングの構成

ユーザーは、WebSEAL 構成ファイルの `[server]` スタンザの設定値を変更して、WebSEAL が読み取ってキャッシュに入れる要求のサイズについての制限を指定できます。

サーバー・サイド要求キャッシングの使用上の注意

- サーバー・サイド・キャッシングを指定することにより、処理能力を超えるデータをキャッシュに入れさせようとするサービス妨害攻撃のタイプから WebSEAL を保護することができます。
- ログイン・プロセス中にユーザー・セッション・タイムアウトが生じた場合は、サーバー・サイド要求キャッシングは正しく機能しないことがあります。その場合、キャッシュ・エントリーは失われます。
- サーバー・サイド要求キャッシングを使用すると、ブラウザーからリソースを操作する能力が制約されることがあります。ブラウザーは、WebSEAL が HTTP リダイレクトを再ビルドしたことを認識しません。したがって、ブラウザーの再ロード/リフレッシュ機能およびキャッシング能力が妨げられることがあります。

以下のセクションでは、ユーザーが変更できる設定値について説明します。

- 236 ページの『request-body-max-read の変更』
- 236 ページの『request-max-cache の変更』

request-body-max-read の変更

request-body-max-read スタンザ・エントリーは、POST 要求の本体から読み取るコンテンツの最大バイト数を指定します。

この構成は、dynurl、認証、および要求のキャッシングに使用されます。

request-body-max-read スタンザ・エントリーは、要求本体のみに影響します。要求行およびヘッダーなどの、要求のその他のコンポーネントに制限を課すことはありません。(「*IBM Security Web Gateway Appliance: Web リバース・プロキシー・スタンザ・リファレンス*」の **max-client-read** 構成エントリーを参照してください。)

request-body-max-read スタンザ・エントリーの値は、要求が満たされるには、その前に認証を受けなければならないユーザーのために WebSEAL がキャッシュに入れるデータの量に影響します。例えば、ログイン・フォームを使用してサブミットされたユーザー名およびパスワードは、**request-body-max-read** の制限に適合する必要があります。このスタンザ・エントリーは、POST 要求および PUT 要求など、本体コンテンツを持つすべての要求に影響します。

このスタンザ・エントリーはフォーム認証に影響します。それは、その種の認証を実行するときに処理される POST データのサイズを制限するからです。フォーム認証用に十分な要求本体サイズを維持するために、WebSEAL は、

request-body-max-read に、絶対最小値である 512 バイトを設定します。最小値より小さい値を指定した場合、その設定は無視され、値 512 が使用されます。最大値に関する制限はありません。

要求本体には POST 要求 URI の照会部分が含まれているため、このスタンザ・エントリーは、動的 URL の処理にも影響を与えます。

注: この設定は、最大 POST サイズを制限しません。最大 POST サイズは制限無しです。

デフォルト値は 4096 バイトです。

```
[server]
request-body-max-read = 4096
```

request-body-max-read のサーバー・サイド・キャッシュの設定値が要求の実行中に超えると、WebSEAL は要求のキャッシング処理を終了します。WebSEAL は「キャッシングの要求が失敗しました」というエラー・メッセージをブラウザーに戻し、このエラーをログ・ファイルに書き込みます。このエラー・メッセージはカスタマイズできます。87 ページの『HTML 応答ページのカスタマイズのためのガイドライン』を参照してください。

request-body-max-read の値は、**request-max-cache** のために指定された値にも影響します。『request-max-cache の変更』を参照してください。

request-max-cache の変更

要求を実現するために、ユーザーに認証を受けるようプロンプトが出されると、その要求にあるデータは、認証完了後の処理のためにキャッシュに入れられます。1 要求についてキャッシュに入れられる最大データ量は、**request-max-cache** スタンザ・エントリーで指定されます。

request-body-max-read のすべての値をキャッシュに入れるには、この値で、他のすべての要求コンポーネントの最大サイズを計算する必要があります。例えば、2048 バイトの要求本体コンテンツをキャッシュに入れる必要があります、さらに、すべての要求ヘッダーと Cookie の最大サイズが 4096 バイトであると予想できる場合、次のようにします。

1. **request-body-max-read** = 2048 と設定する。
2. **request-max-cache** = 2048 + 4096 = 6144 と設定する。

request-max-cache のデフォルト値は 8192 です。

```
[server]
request-max-cache = 8192
```

request-max-cache のサーバー・サイド・キャッシングの設定値が要求の実行中に超えると、WebSEAL は要求のキャッシング処理を終了します。WebSEAL は「キャッシングの要求が失敗しました」というエラー・メッセージをブラウザーに戻し、このエラーをログ・ファイルに書き込みます。このエラー・メッセージはカスタマイズできます。87 ページの『HTML 応答ページのカスタマイズのためのガイドライン』を参照してください。

この値の最大サイズは、データ・タイプによって強制される最大サイズ以外にはありません。しかし、サイズを大きくすることによって、パフォーマンスとシステム・セキュリティに悪い影響を及ぼす可能性が増します。大きなバッファを割り振れば、メモリーの使用が増え、パフォーマンスを下げる方向に働きます。もっと重要なことは、非常に大きなバッファを割り振ると、悪意あるユーザーによるサービス妨害攻撃のリスクが増えます。このリスクが増えるのは、単純に言えば、WebSEAL が、より多くのデータをメモリーにロードし保持すれば、より大きなバッファをユーザーに与え、そこでアタックが試行されるからです。

第 11 章 パスワード処理

この章では、WebSEAL 認証で使用可能なパスワード処理オプションについて説明します。

この章で扱うトピックは以下のとおりです。

- 『ログイン失敗ポリシー (「スリー・ストライク」 ログイン・ポリシー)』
- 244 ページの『パスワード・ストレングス・ポリシー』

ログイン失敗ポリシー (「スリー・ストライク」 ログイン・ポリシー)

このセクションは、以下のトピックで構成されています。

- 『ログイン失敗ポリシーの概念』
- 240 ページの『ログイン失敗ポリシーの設定』
- 241 ページの『アカウント使用不可時間間隔の設定』
- 242 ページの『アカウント使用不可通知応答の構成』
- 242 ページの『複製された WebSEAL サーバーにおけるログイン失敗ポリシー』

ログイン失敗ポリシーの概念

ログイン失敗 (「スリー・ストライク」) ポリシーは、LDAP ベースのユーザー・レジストリーを使用した Security Access Manager のインストールで使用できます。このポリシーを使用すると、ログイン試行の失敗が許される最大回数 (n) と、ペナルティー・ロックアウト時間 (x) を指定することができます。つまり、「n」回ログインの試行に失敗したユーザーを、「x」秒間ロックアウトする (つまり、アカウントを使用不可にする) ようにできます。

ログイン失敗ポリシーによって、コンピューターに対するパスワード攻撃を防ぐことができます。このポリシーでは、ユーザーがログインの試行を再度行えるようになるまでに一定時間待機しなければならないという条件が作成されます。例えば、このポリシーでは、ログインの試行に 3 回失敗したら 180 秒間のロックアウト・ペナルティーを科すことができます。この種のログイン・ポリシーでは、コンピューター生成のランダムなログインが 1 秒間に何回も試行されるのを防ぐことができます。

ログイン失敗ポリシーでは、2 つのポリシー設定が連動して機能する必要があります。

- ログイン試行の失敗が許される最大回数:

max-login-failures

- ログイン試行失敗の設定に達したまたは超えた場合のペナルティー:

disable-time-interval

ペナルティーの設定には、一時的なアカウント・ロックアウトの時間間隔、またはアカウントの完全な使用不可化が含まれます。

WebSEAL は、「サーバー応答エラー (server response error)」ページ (acct_locked.html) を戻し、ユーザーにペナルティーを通知します。WebSEAL 構成ファイルの [server] スタンザ内の **late-lockout-notification** スタンザ・エントリーで、ユーザーが **max-login-failures** 制限に達したとき、または制限に達した後の次のログイン試行のときにこの通知を行うかどうかを指定します。

199 ページの『ログイン失敗ポリシー制限でのユーザー・セッションの削除』も参照してください。

ログイン失敗ポリシーの設定

このタスクについて

ログイン失敗ポリシーは、アカウント・ロックアウト・ペナルティーが科されるまでに許容されるログイン試行失敗の最大回数を制御します。

手順

- **pdadmin policy** コマンドを使用して、ログイン失敗ポリシーを設定します。ログイン失敗ポリシーを設定するには、以下の構文を使用します。

```
policy set max-login-failures {number|unset} [-user username]
```

- 現在のログイン失敗ポリシーの設定を表示するには、以下の構文を使用します。

```
policy get max-login-failures [-user username]
```

number 引数で、ペナルティーが適用されるまでに許容されるログイン試行失敗の回数を指定します。デフォルトでは、このポリシーはログイン試行回数 10 回に設定されています。例:

```
pdadmin> policy get max-login-failures  
Maximum login failures: 10
```

unset 引数により、ポリシーは使用不可になります。これを設定すると、ポリシーには値が含まれず、ポリシーは検査も実行もされません。

- **max-login-failures** ポリシーを特定のユーザーに対して適用することも、ユーザー・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。

例

グローバル設定の例:

```
pdadmin> policy set max-login-failures 3
```

ユーザー固有設定の例:

```
pdadmin> policy set max-login-failures 5 -user laura
```

アカウント・ロックアウト・ペナルティーの値は、**disable-time-interval** ポリシーで指定します。241 ページの『アカウント使用不可時間間隔の設定』を参照してください。

アカウント使用不可時間間隔の設定

このタスクについて

ログイン失敗ポリシーは、アカウント・ロックアウト・ペナルティーが科されるまでに許容されるログイン試行失敗の最大回数を制御します。

手順

- **pdadmin policy** コマンドを使用して、ログイン失敗ポリシーのペナルティー時間間隔を設定します。ペナルティー時間間隔を設定するには、以下の構文を使用します。

```
policy set disable-time-interval {number|unset|disable} [-user username]
```

- 現在のペナルティー時間間隔の設定を表示するには、以下の構文を使用します。

```
policy get disable-time-interval [-user username]
```

number 引数で、ログイン試行失敗の最大数に達したか超えた場合にアカウントをロックアウトする秒数を指定します。デフォルトでは、ロックアウト時間間隔は 180 秒です。例:

```
pdadmin> policy get disable-time-interval  
Disable time interval: 180
```

unset 引数により、ポリシーは使用不可になります。これを設定すると、ポリシーには値が含まれず、ポリシーは検査も実行もされません。

disable 引数は、ログイン試行制限に達したか超えた後、このユーザーをアカウントから永続的にロックアウトし、このユーザーの **LDAP account valid** 属性を「no」に設定します。アドミニストレータは、Web ポータル・マネージャー (WPM)、または **pdadmin** ユーティリティを使用して、アカウントを再度使用可能にできます。

注: **disable-time-interval** を「disable」に設定すると、管理オーバーヘッドが増えます。これは、アドミニストレータが、アカウントを手動で再び使用可能にしなければならないからです。アカウントが再び使用可能になっても、更新された **account valid** LDAP 属性情報が即時に有効にならない場合があります。この状態が起こるのは、複製された LDAP サーバーが組み込まれた LDAP 環境を持つ WebSEAL を使用している場合です。この場合、更新を実行する時間間隔を指定した LDAP 構成設定値にしたがって、更新された情報が LDAP レプリカに伝搬されます。

disable-time-interval ポリシーを特定のユーザーに対して適用することも、ユーザー・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。

例

グローバル設定の例:

```
pdadmin> policy set disable-time-interval 60
```

ユーザー固有設定の例:

```
pdadmin> policy set disable-time-interval disable -user laura
```

WebSEAL 構成ファイルの **[server]** スタンザ内の **late-lockout-notification** スタンザ・エントリーで、ユーザーが **max-login-failures** 制限に達したとき、または制限に達した後の次のログイン試行のときにこのアカウント・ロックアウト通知を行うかどうかを指定します。『アカウント使用不可通知応答の構成』を参照してください。

アカウント使用不可通知応答の構成

このタスクについて

WebSEAL は、**max-login-failures** 制限に達するか超えた場合に、「サーバー応答エラー (server response error)」ページ (acct_locked.html) を戻し、ユーザーにペナルティーを通知します。

WebSEAL 構成ファイルの **[server]** スタンザ内の **late-lockout-notification** スタンザ・エントリーで、ユーザーが **max-login-failures** 制限に達したか制限に達した後の次のログイン試行のときにこのエラー・ページを戻すかどうかを指定します。

アカウント・ロックアウトまたはアカウント使用不可のアクションによってユーザーのセッション・キャッシュ・エントリーが削除されるわけではありませんが、アカウントがアンロックされるまで、そのユーザーによる以降のログインが抑制されます。

手順

- WebSEAL の新規インストールにおけるデフォルトの **late-lockout-notification** 設定は「no」です。**max-login-failures** ポリシーで設定されている最大値に達すると、WebSEAL はアカウント使用不可エラー・ページをユーザーに即時に送信します。例:

```
[server]
late-lockout-notification = no
```

- WebSEAL のマイグレーション・インストールのデフォルト設定は「yes」です。**max-login-failures** ポリシーで設定されている最大値に達すると、WebSEAL が別のログイン・プロンプトをユーザーに戻します。次のログイン試行時までは、WebSEAL はアカウント使用不可エラー・ページをユーザーに送信しません。この設定は、6.0 より前のバージョンの **max-login-failures** ポリシーの動作を表しています。例:

```
[server]
late-lockout-notification = yes
```

- **disable-time-interval** ポリシーが秒数に設定されている場合、アカウントが一時的にロックアウトされていることを示すエラー・メッセージが表示されます。
- **disable-time-interval** ポリシーが「disable」に設定されている場合、アカウントが使用不可でアドミニストレーターがそのアカウントをリセット (アンロック) する必要があることを示すエラー・メッセージが表示されます。

複製された WebSEAL サーバーにおけるログイン失敗ポリシー

ログイン失敗ポリシーを使用して、ログインが指定した回数失敗するとアカウントがロックされるようにすることができます。このポリシーが意図したとおりに働くのは、WebSEAL サーバーを 1 つだけ含む構成の場合です。ロード・バランシ

グ・メカニズムが適用された複数のフロントエンド WebSEAL サーバーを含む構成の場合、このポリシーの結果は、各 WebSEAL サーバーが、ログイン失敗の回数について、それぞれのローカル・カウントを維持するというデフォルト動作の影響を受けます。

例えば、**max-login-failures** が 3 回に設定されているときに、クライアントによる最初の 3 回の試行が失敗した場合、このサーバーではそのクライアントのアカウントはロックされます。しかし、このクライアントがさらにログイン試行を続行した場合、ロード・バランシング・メカニズムは、第 1 のサーバーへの接続が失敗したことを検出し、他の使用可能な複製 WebSEAL サーバーに要求をリダイレクトします。したがって、クライアントには、さらに 3 回ログインを試行する機会が与えられることになります。

各 WebSEAL サーバーでそれぞれ試行回数「n」が設定され、複製フロントエンド WebSEAL サーバー数が「m」であるとすれば、1 つのサーバーで「n」回の試行後に初回のアカウント・ロックが発生することが保証されます。さらに、構成済みのすべてのサーバーで合計試行回数が「n」x「m」になるまでログインできることも保証されます。しかし、「n」回の試行の後で再び認証が失敗した場合、その原因が特定サーバー上のロックにあるのか、それとも、残りの複製サーバーで不正なログイン試行を続けたことにあるのかは分かりません。

「n」x「m」の計算は、完全なロックアウトが生じるまでの、連続的なログイン試行の合計回数についての固定した最大上限数を表します。この数は、統計上「パスワード破り」に必要とされている試行回数よりはるかに少ないと見ることができます。

ビジネス・セキュリティー・ソリューションでログイン失敗ポリシーが必要な場合、このポリシーにおけるロード・バランシング化された複数フロントエンド WebSEAL 構成の影響を理解する必要があります。

実行できるログイン試行回数の削減

連続するログイン試行の合計数の上限を小さくするには、**login-failures-persistent** スタンザ・エントリーを使用します。

このタスクについて

login-failures-persistent エントリーは、WebSEAL 構成ファイルの **[ldap]** スタンザにあります。このエントリーは、ログイン失敗数を WebSEAL システムのローカル・キャッシュで追跡するか LDAP レジストリーで追跡するかを制御します。失敗をレジストリーで追跡する場合は、レジストリーを共有するすべてのレプリカで失敗回数が共有されます。この構成により、最大試行回数は「n」x「m」でなく、「n」に削減されます。ログインを試行するたびに LDAP サーバーへの書き込み操作が発生するため、**login-failures-persistent** を使用可能にすると、結果としてパフォーマンスが軽微の影響を受けます。

パスワード・ストレングス・ポリシー

パスワード・ストレングス・ポリシーは、パスワードの作成の際に適用されるパスワード・ポリシー・ルールによる規定のことです。

このセクションは、以下のトピックで構成されています。

- 『パスワード・ストレングス・ポリシー』
- 『パスワード・ストレングス・ポリシー・コマンドの構文』
- 246 ページの『デフォルトのパスワード・ストレングス・ポリシーの値』
- 246 ページの『有効なパスワードと無効なパスワードの例』
- 247 ページの『ユーザーおよびグローバル設定の指定』

パスワード・ストレングス・ポリシーの概念

パスワード・ストレングス・ポリシーを制御できます。

Security Access Manager には、パスワード・ストレングス・ポリシーを制御するための 5 つの **pdadmin policy** コマンドが用意されています。これらの **pdadmin policy** コマンドについては、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

パスワード・ストレングス・ポリシー

pdadmin policy コマンドによりインプリメントされる 5 つのパスワード・ストレングス・ポリシーは、以下のとおりです。

- 最小のパスワード長 (**min-password-length**)
- 最小英字数 (**min-password-alphas**)
- 最小非英字数 (**min-password-non-alphas**)
- 最大反復文字数 (**max-password-repeated-chars**)
- スペースの使用を許可 (**password-spaces**)

これらのポリシーは、**pdadmin** ユーティリティーまたは Web Portal Manager でユーザーを作成する場合、あるいは **pdadmin** ユーティリティー、Web Portal Manager、または **pkmspasswd** ユーティリティーによってパスワードが変更される場合に、適用されます。

パスワード・ストレングス・ポリシー・コマンドの構文

以下の **pdadmin policy** コマンドは、パスワード・ストレングス・ポリシーを設定するために使用され、ユーザー・レジストリーの LDAP タイプでのみ使用できます。

unset オプションは、このポリシー属性を使用不可にします。このオプションが使用されるとポリシーは実施されません。

コマンド	説明
<code>policy set min-password-length {number unset} [-user username]</code>	
<code>policy get min-password-length [-user username]</code>	

コマンド	説明
	パスワードの最小文字数をコントロールするポリシーを管理します。 アドミニストレーターは、このポリシーを特定のユーザーに対して適用することも、デフォルト・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。 デフォルトの設定は 8 です。
policy set min-password-alphas { <i>number</i> unset} [-user <i>username</i>]	
policy get min-password-alphas [-user <i>username</i>]	
	パスワード内で使用できる英字の最小数をコントロールするポリシーを管理します。 アドミニストレーターは、このポリシーを特定のユーザーに対して適用することも、デフォルト・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。 デフォルトの設定は 4 です。
policy set min-password-non-alphas { <i>number</i> unset} [-user <i>username</i>]	
policy get min-password-non-alphas [-user <i>username</i>]	
	パスワード内で使用できる非英字 (数字) の最小数をコントロールするポリシーを管理します。 アドミニストレーターは、このポリシーを特定のユーザーに対して適用することも、デフォルト・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。 デフォルトの設定は 1 です。
policy set max-password-repeated-chars { <i>number</i> unset} [-user <i>username</i>]	
policy get max-password-repeated-chars [-user <i>username</i>]	
	パスワード内で使用できる最大反復文字数をコントロールするポリシーを管理します。 アドミニストレーターは、このポリシーを特定のユーザーに対して適用することも、デフォルト・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。 デフォルトの設定は 2 です。
policy set password-spaces {yes no unset} [-user <i>username</i>]	
policy get password-spaces [-user <i>username</i>]	

コマンド	説明
	パスワードにスペースを含めるかどうかを制御するポリシーを管理します。 アドミニストレーターは、このポリシーを特定のユーザーに対して適用することも、デフォルト・レジストリーにリストされているすべてのユーザーに対してグローバルに適用することもできます。 デフォルトの設定は <code>unset</code> です。

デフォルトのパスワード・ストレングス・ポリシーの値

次の表には、パスワード・ストレングス・ポリシーとデフォルト値が記載されています。

ポリシー	デフォルト値
min-password-length	8
min-password-alphas	4
min-password-non-alphas	1
max-password-repeated-chars	2
password-spaces	設定しない

Security Access Manager の以前のリリースと同じようにパスワード・ポリシーを動作させるには、上記リストの 5 つのパスワード・ポリシーにそれぞれ **unset** オプションを適用してください。

有効なパスワードと無効なパスワードの例

以下の表は、いくつかのパスワードの例と、5 つのパスワード・ストレングス・ポリシーのデフォルト値が設定されている場合の結果を示したものです。

例	結果
password	無効: 最低でも 1 文字以上の非英字が含まれていなければなりません。
pass	無効: 最小でも 8 文字でなければなりません。
pass1234	無効: 反復文字が 3 つ以上使用されています。
12345678	無効: 最低でも 4 文字の英字が含まれていなければなりません。
password3	有効。

ユーザーおよびグローバル設定の指定

このタスクについて

pdadmin policy コマンドは、特定のユーザーに対して設定する (**- user** オプションを使用) ことも、グローバルに設定する (**- user** オプションを使用しない) こともできます。ユーザー固有の設定は、すべて、ポリシーのグローバル設定をオーバーライドします。

ポリシーを使用不可にすることもできます (**unset** 引数を使用)。ポリシーには値が含まれず、ポリシーは検査も実行もされません。

例

最小のパスワード長を 8 文字とするグローバル・ポリシーが作成されます。このポリシーの例外として、ユーザー **matt** には、最小のパスワード長を 4 文字とするポリシーが適用されます。

```
pdadmin> policy set min-password-length 8
pdadmin> policy set min-password-length 4 -user matt
pdadmin> policy get min-password-length
Minimum password length: 8
pdadmin> policy get min-password-length -user matt
Minimum password length: 4
```

ユーザー **matt** に固有の最小パスワード長ポリシーは **unset** です。ユーザー **matt** は、パスワードの長さの下限が 8 文字というグローバルなポリシーによって管理されるようになります。

```
pdadmin> policy set min-password-length unset -user matt
pdadmin> policy get min-password-length -user matt
Minimum password length: 8
```

グローバル最小パスワード長ポリシーは、**unset** です。ユーザー **matt** を含むすべてのユーザーに対して、パスワードの長さの下限を定めるポリシーは適用されなくなります。

```
pdadmin> policy set min-password-length unset
pdadmin> policy get min-password-length
Minimum password length: unset
```

第 12 章 資格情報の処理

この章では、WebSEAL ユーザー資格情報の処理に影響を与える機能を説明します。

この章で扱うトピックは以下のとおりです。

- 『資格情報用の拡張属性』
- 254 ページの『資格情報リフレッシュ』

資格情報用の拡張属性

このセクションは、以下のトピックで構成されています。

- 『資格情報にレジストリー属性を追加するためのメカニズム』
- 250 ページの『レジストリー属性資格サービスの構成』
- 252 ページの『拡張資格情報属性の junction の取り扱い』

資格情報にレジストリー属性を追加するためのメカニズム

ユーザー資格情報に属性を追加するように外部サービスを構成することができます。

WebSEAL 認証プロセスは、Security Access Manager ユーザー・レジストリーにアクセスして、ユーザーの資格情報をビルドします。資格情報には、アクセス決定を行うために必要な、ユーザー名およびユーザーが属するグループのリストなどのユーザー情報が含まれています。

WebSEAL では、アドミニストレーターやアプリケーション開発者が認証プロセスを拡張できるようにするさまざまなメカニズム (サービス) をサポートしています。WebSEAL は、認証プロセスを実行するとき、インプリメントされ、構成済みになっている外部サービスがあるかどうかチェックします。そのような外部サービスがあれば、WebSEAL はこれらのサービスを呼び出します。これらのサービスは独自の処理を行って、ユーザー ID についての拡張属性のリストをビルドできます。これらの拡張属性はユーザー資格情報に追加されます。

以下のサービスがサポートされています。

レジストリー属性資格サービス

この資格サービスは、デフォルトで Security Access Manager に組み込まれています。このサービスは、Security Access Manager 資格サービスの、資格情報属性付与サービス というクラスのインプリメンテーションです。レジストリー属性資格サービス は、ユーザー・レジストリー (例えば LDAP ユーザー・レジストリー) から指定されたユーザー情報を取得し、そのデータを、ユーザー資格情報の中の属性リストに挿入します。この組み込みレジストリー属性資格サービスは汎用資格サービスであり、多くのリソース・マネージャーで使用することができます。このサービスは、前のメソッドにとって代わるものです (前のメソッドでは、アドミニストレーターが、

pd.conf 構成ファイルの **[ldap-ext-creds-tag]** スタンザに「tag/value」エントリーを追加することが必要でした)。構成情報については、『レジストリー属性資格サービスの構成』を参照してください。

注: Security Access Manager では、追加情報を追加するために使用できる、追加の組み込み資格サービスが提供されていることに注意してください。ただし、これらの追加サービスでは、追加情報を、ユーザー・レジストリー・エントリー以外のソースから得ます。例えば、拡張属性資格サービスは、保護リソース・オブジェクト・スペースにある ACL および POP から情報を得ます。詳しくは、「*IBM Security Access Manager for Web: Authorization C API Developer Reference*」の資格サービスの説明を参照してください。

レジストリー属性資格サービスの構成

以下のセクションにある手順を実行してください。

1. 『資格情報に追加する属性の決定』
2. 『資格情報に追加する属性の指定』

資格情報に追加する属性の決定

ユーザー資格情報に追加する属性を決定する必要があります。

このタスクについて

ユーザー資格情報に追加する各ユーザー属性は、Security Access Manager 構成ファイルで定義する必要があります。通常、この構成は、WebSEAL 構成ファイルに行われます。

手順

- Security Access Manager ユーザー・レジストリー (例えば、LDAP ユーザー・レジストリー) に進みます。
- 資格情報属性付与サービスを使用してレジストリーから抽出しユーザー資格情報に入れさせたい、それぞれのユーザー・レジストリー項目の名前のリストを作成します。ユーザー DN およびグループ DN も必要です。

資格情報に追加する属性の指定

資格情報に追加する属性は、いくつかのスタンザで構成します。

このタスクについて

この情報は、WebSEAL 構成ファイルに追加します。

以下のエントリー例を検討してください。

```
[TAM_CRED_ATTRS_SVC]
eperson = azn_cred_registry_id
group = cn=enterprise, o=tivoli

[TAM_CRED_ATTRS_SVC:eperson]
tagvalue_credattrs_lastname = sn
tagvalue_credattrs_employeetype = employeetype
tagvalue_credattrs_address = homepostaladdress
```

```
tagvalue_credattrs_email = mail

[TAM_CRED_ATTRS_SVC:group]
tagvalue_credattrs_businesscategory = businesscategory
```

スタンザ名 **[TAM_CRED_ATTRS_SVC]** はサービス ID です。このスタンザの内側には、検索する属性のソースが入っています。ソース *names*、例えば *eperson* および *group* は、レジストリーの中のソースのロケーションを識別するために使用されます。これらは定義することが必要です。これらのソースの値は、このレジストリーに存在するレジストリー ID です。これらの値は、現存する資格情報属性の名前でもかまいません。その場合は、サービスは、それぞれの値を自動的に検索して使用します。

手順

別のスタンザのサービス・スタンザのもののソースのそれぞれのレジストリー属性を構成します。別のスタンザの構文は、サービス ID ライブラリー名のあとにコロロン (:) を続け、その後にソース名を続けます。複数のサービスを 1 つのファイルに構成できるので、この接続は必要です。構成ファイルのエントリーには、ユーザー定義の資格情報属性へのユーザー・レジストリー属性のマッピングが入ります。例えば、LDAP ユーザー・レジストリーでは、ユーザーの DN は次のようになります。

```
cn=joeuser,o=tivoli
```

このユーザーの場合、LDAP ユーザー・レジストリー項目は次のようになります。

```
sn=Smith
employeetype=bankteller
homepostaladdress="3004 Mission St Santa Cruz CA 95060"
mail=joeuser@bigco.com
```

グループ *cn=enterprise,o=tivoli* の場合、LDAP グループ・レジストリー項目は次のようになります。

```
businesscategory=finance
```

これらのサンプル構成エントリーを使用した場合、返される属性リストには以下のエントリーが設定されます。

属性名	属性値
credattrs_lastname	Smith
credattrs_employeetype	bankteller
credattrs_address	3004 Mission St Santa Cruz CA 95060
credattrs_email	joeuser@example.com
credattrs_businesscategory	finance

サービス、ソース、および属性は、多値でもかまわないことに注意してください。同じ属性名をスタンザ・エントリーのキーワードとして指定した場合、リトリートされる属性は、それらが異なるソースからの属性であっても、多値属性として追加されます。

例えば、複数の資格サービスを、1 つにチェーニングできます。これによって、1 つのサービスからリトリートされた値を、別のサービスの入力値として使用できるようになります。同様に、ユーザー・レジストリー内の複数の DN から属性をリトリートできます。したがって、1 つのグループのユーザーのすべての

`businesscategory` エントリーのリストが必要な場合は、上記の例を使用して、複数のユーザー (DN) の中にある値を 1 つの `credattrs_businesscategory` 属性に追加することができます。

例えば、資格情報に追加する `myemployeeinfo` という名前の属性をビルドし、さらに、この属性に、認証される各人の姓と従業員タイプを入れるようにしたい場合、以下の定義を行います。

```
[myID]
source = azn_cred_authzn_id

[myID:source]
myemployeeinfo = lastname
myemployeeinfo = employeetype
```

拡張資格情報属性の `junction` の取り扱い

前のセクションで作成したユーザー定義の資格情報を、`junction` を介してバックエンド・サーバーに送られる要求の HTTP ヘッダーに組み込むことができます。

そのためには、資格情報から拡張属性データを抽出し、要求の HTTP ヘッダーにそのデータを挿入するように、`junction` を構成する必要があります。この機能を達成するには、WebSEAL 保護オブジェクト・スペース内の `junction` オブジェクトに、**HTTP-Tag-Value** という `junction` 拡張属性を設定する必要があります。

WebSEAL 保護オブジェクト・スペース内の `junction` オブジェクトに拡張属性を設定するには、**pdadmin object modify set attribute** コマンドを使用します。

```
pdadmin> object modify object_name set attribute attr_name attr_value
```

注: 上記のコマンドは、1 つの連続したコマンド行として入力する必要があります。

拡張属性 (`attr_name`) を使用することにより、`junction` が特定タイプの機能を実行できるようにすることができます。**HTTP-Tag-Value** 拡張属性は、`junction` に対して、ユーザーの資格情報から特定値を抽出し、その値を HTTP ヘッダーに入れてバックエンド・サーバーに送るように指示します。**HTTP-Tag-Value** 拡張属性の値の形式は以下のとおりです。

```
credential_extended_attribute_name = http_header_name
```

`credential_extended_attribute_name` エントリーは、WebSEAL 構成ファイルに指定されている属性と同じですが、「tagvalue_」接頭部はありません。このエントリーは大文字小文字の区別はありません。`http_header_name` エントリーは、`junction` を介してデータを配信するために使用する HTTP ヘッダーの名前を指定します。

例えば、以下のようにします (1 行で入力します)。

```
pdadmin> object modify /WebSEAL/WS1/junctionA set attribute
HTTP-Tag-Value credattrs_lastname=surname
```

WebSEAL は、バックエンド・アプリケーション・サーバーへのユーザー要求を処理するときに、`junction` オブジェクトに **HTTP-Tag-Value** 属性が構成されていないかどうかを確認します。

この例では、構成済み junction は、要求を出しているユーザーの資格情報を調べ、資格情報拡張属性 `tagvalue_credattrs_lastname` の値を抽出し、その値を次のように HTTP ヘッダーに組み込みます。

`surname:Smith`

上記を要約すると以下ようになります。

Junction オブジェクトに設定される HTTP-Tag-Value 属性の値	<code>credattrs_lastname=surname</code>
ユーザー資格情報に現れる属性の名前と値 (<code>tagvalue_credattrs_lastname=sn</code> 以降):	<code>tagvalue_credattrs_lastname:Smith</code>
HTTP ヘッダー内の名前と値	<code>surname:Smith</code>

バックエンド・アプリケーションが CGI アプリケーションである場合、CGI 指定は、次の形式の環境変数により、CGI プログラムが HTTP ヘッダーを使用できるようにすることを指示します。

`HTTP_http_header_name`

例:

`HTTP_surname=Smith`

Junction 先サーバーに、複数のユーザー属性データを HTTP ヘッダーに入れて渡すことができます。そのためには、複数の `pdadmin object modify set attribute` コマンドを使用して、複数の **HTTP-Tag-Value** junction 属性 (1 コマンドにつき 1 属性ずつ) を指定できます。

HTTP-Tag-Value 拡張属性は junction に直接付加することが必要

junction ポイントの下の特定のオブジェクトに対していずれの資格情報属性を渡すかを判別するために評価を実行するとき、評価は子オブジェクトではなく junction オブジェクトについて実行されます。例えば、以下の名前の junction オブジェクトについて **HTTP-Tag-Value** 拡張属性を作成したとします。

`/WebSEAL/myinstance/jct1`

また、以下の名前を持つ junction の子オブジェクトについても **HTTP-Tag-Value** 拡張属性を作成したとします。

`/WebSEAL/myinstance/jct1/child`

ここで、クライアントが `/WebSEAL/myinstance/jct1/child` にアクセスすると、`/WebSEAL/myinstance/jct1` に付加されている属性のみが使用され、子の属性は無視されます。したがって、**HTTP-Tag-Value** 拡張属性は junction に直接付加する必要があります。

ここで理解しておくべき重要なことは、上記のことは継承が原因ではなく、WebSEAL では junction オブジェクトを子と関連付けて判別し、junction オブジェクトそれ自体の **HTTP-Tag-Value** 属性を検出することが原因だということです。実際、拡張属性の継承はバージョン 6.0 で導入されましたが、**HTTP-Tag-Value** 属性の処理ではこの継承は使用されません。したがって、以下の場所に junction を作成したとします。

/WebSEAL/myinstance/jct1

上記の場所には **HTTP-Tag-Value** 属性を付加せず、その親である以下の場所に属性を付加するとします。

/WebSEAL/myinstance/

このとき、付加された属性は、jct1 またはそのすべての子について使用されません。**HTTP-Tag-Value** 拡張属性を /WebSEAL/myinstance/jct1 に直接付加する必要があります。

資格情報リフレッシュ

このセクションは、以下のトピックで構成されています。

- 『資格情報リフレッシュの概念』
- 258 ページの『資格情報リフレッシュの構成』
- 260 ページの『資格情報リフレッシュの使用法』

資格情報リフレッシュの概念

このセクションは、以下のトピックで構成されています。

- 『資格情報リフレッシュの概要』
- 255 ページの『資格情報リフレッシュのルール』
- 256 ページの『キャッシュに入れられた資格情報のリフレッシュ』
- 257 ページの『構成ファイルの構文と使用法』
- 257 ページの『保存およびリフレッシュのためのデフォルトの設定値』
- 258 ページの『制限』

資格情報リフレッシュの概要

WebSEAL では、資格情報リフレッシュ機能を構成できます。

ユーザーが WebSEAL の認証を受けるとき、認証プロセスは、Security Access Manager ユーザー・レジストリーにアクセスして、ユーザーの資格情報をビルドします。資格情報には、要求されたリソースへのアクセスをユーザーに認可するかどうかを決めるために Security Access Manager が必要とするユーザーについての情報が入っています。資格情報の 1 つの例は、ユーザーが所属しているグループのリストです。

ユーザー・セッション中に、ユーザー情報の変更が起こります。例えば、ユーザーが新規のグループに追加されることがあります。このような変更が起こると、新しいユーザー情報を反映するために、ユーザー資格情報の内容を更新する、つまりリフレッシュする 必要が起こります。WebSEAL には、ユーザーがログアウトし、認証を再度受けなくても、資格情報のリフレッシュができるようにするメカニズムが用意されています。

アドミニストレーターは、資格情報リフレッシュ機能がどのように動作するかをコントロールできます。WebSEAL には、資格情報属性をリフレッシュする (更新す

る) か、保持する (保存する) かを指定できる構成設定値があります。これによって、ユーザー資格情報を、ユーザー・セッション中にどのように取り扱うかを正確にコントロールできます。

ご使用の WebSEAL サーバーの認証プロセスに、ユーザーに関する追加情報または拡張情報を提供するメカニズムへのコールアウトが組み込まれているときは、資格情報リフレッシュの構成設定値の使用が重要になります。これらのメカニズムには、以下のものがあります。

- 資格情報属性付与サービス

このサービスは、デフォルトで、Security Access Manager に組み込まれています。

上にリストした資格情報属性サービスについて詳しくは、249 ページの『資格情報にレジストリー属性を追加するためのメカニズム』を参照してください。

資格情報リフレッシュが実行されると、デフォルトの資格情報属性付与サービスが実行されます。

資格情報リフレッシュ構成設定値を使用すると、資格サービスの初期使用中に得られた属性を保存できます。例えば、ユーザー・セッション開始のタイム・スタンプが属性に入っていた場合、資格情報がリフレッシュされても、タイム・スタンプを保存する必要がある場合があります。

また、資格情報リフレッシュ構成設定値を使用すると、資格情報拡張属性認証モジュールから得られた属性を保存できます。カスタム認証モジュールは、資格情報の再ビルド中に再び実行されることはないので、構成ファイル設定値を使用して、新しい資格情報に追加される属性を指定します。

資格情報リフレッシュのルール

資格情報リフレッシュには、ユーザー識別のための新しい資格情報の生成が必要で、これに続けて、最初のユーザー認証の際に得られた古い資格情報の内容に照らして新しい資格情報の内容の評価が行われます。2 つの資格情報の内容は、以下のルールにしたがって、マージされた資格情報に結合されます。

1. ある属性が新しい資格情報にはあるが古い資格情報にないときは、その属性はマージされた資格情報に追加されます。
2. 以下の属性はマージされた資格情報に追加されますが、それぞれの値は古い資格情報にのみ基づいています。これらの属性は、許可 API によって使用されます。これらの値は、新しい資格情報の中の値によって変更されることはありません。

```
AZN_CRED_AUTHNMECH_INFO  
AZN_CRED_BROWSER_INFO  
AZN_CRED_IP_ADDRESS  
AZN_CRED_PRINCIPAL_NAME  
AZN_CRED_AUTH_METHOD  
AZN_CRED_USER_INFO  
AZN_CRED_QOP_INFO
```

3. 新しい資格情報の中に対応する属性がある古い資格情報の属性のそれぞれに、以下のルールが適用されます。

- この属性に一致するエントリーが構成ファイルにあると、マージされた資格情報の中の属性は、構成ファイルのエントリーの値にしたがって保存あるいはリフレッシュされます。
 - この属性に一致するエントリーが構成ファイルにないときは、マージされた資格情報の中の属性に、新しい資格情報にある値が割り当てられます。
4. 新しい資格情報の中に対応する属性がない 古い資格情報の属性のそれぞれに、以下のルールが適用されます。
- `refresh` を指定している属性用の構成ファイル・エントリーがあるときは、その属性はマージされた資格情報に追加されません。
 - `preserve` を指定している属性用の構成ファイル・エントリーがあるときは、その属性はマージされた資格情報に追加されます。
 - 構成ファイルに、その属性用のエントリーがないときは、その属性はマージされた資格情報に追加されません。

キャッシュに入れられた資格情報のリフレッシュ

ユーザー・レジストリーの一部は、キャッシュに入れられた情報を維持しています。キャッシュに入れられたデータは、特定の時間だけ保持されて、その後廃棄されます。キャッシュに入れられたデータが期限切れになると、次にユーザー・レジストリーがアクセスされるまで、データはキャッシュに再ロードされません。したがって、ユーザー・レジストリー・データに変更が行われても、データは、メモリー内のキャッシュに即時に入れられません。同様に、複製された LDAP ユーザー・レジストリーを使用しているとき、複製されたレジストリーへの更新は即時に行われません。

WebSEAL ユーザー・キャッシュにあるデータのデフォルトの存続時間は 30 秒です。この存続時間は、データが最初にキャッシュに入るとき、例えば、ユーザーが初めて認証されたとき、あるいは、キャッシュに入れられたデータの期限が切れ、WebSEAL がレジストリーに連絡してデータを更新するときに始まります。WebSEAL は、資格情報リフレッシュのイベント中にレジストリーに連絡してデータを更新します。キャッシュに入れられた情報は、レジストリーから最初に得られてから 30 秒間有効です。30 秒がたってからは、すべての資格情報リフレッシュ操作は、ユーザー・レジストリーで直接行われます。ユーザー・レジストリーにアクセスすることによっても、ユーザー・データはキャッシュに再ロードされます。

次の例に、ユーザー・キャッシュを更新するアルゴリズムを示します。

1. ユーザーは、時間 `auth_time` に認証される。
2. ユーザーは、時間 `auth_time + 120 秒` にグループに追加される。
3. ユーザーの資格情報は、時間 `auth_time + 130 秒` にリフレッシュされる。

ユーザー・キャッシュ・データは、時間 `auth_time + 30 秒` に期限が切れているので、新しいグループ・メンバーシップがユーザーの資格情報に追加されます。

4. ユーザーが、別のグループに、時間 `auth_time + 135 秒` に追加される。
5. ユーザーの資格情報は、時間 `auth_time + 140 秒` にリフレッシュされる。

ユーザー資格情報が `auth_time + 140 秒` でリフレッシュされるとき、ユーザー資格情報に、新しいグループ・メンバーシップは追加されません。これは、キャッシュに入れられたユーザー・データが有効 (期限切れでない) と考えられているときに、

キャッシュに入れられたユーザー・データに基づかないでユーザー資格情報がビルドされているからです。ユーザー・キャッシュ・データは、時間 `auth_time + 130 秒` で更新されたので、`auth_time + 160 秒` まで更新されるようにスケジュールされていません。したがって、アドミニストレーターは、リフレッシュ・コマンドを実行するのを時間 `auth_time + 160 秒` まで待たなければなりません。その時点で、新しいグループ・メンバーシップがユーザー資格情報に追加されます。

構成ファイルの構文と使用法

資格情報リフレッシュの振る舞いは、WebSEAL 構成ファイルの **[credential-refresh-attributes]** スタンザのエントリーによってコントロールされます。次の形式が使用されます。

```
attribute_name_pattern = {preserve|refresh}
```

属性名パターンは、属性のセットを選択するために使用されます。ワイルドカード・マッチングがサポートされています。

1 つの属性が、多くの異なるワイルドカード・パターンでマッチングされます。したがって、構成ファイル内のエレメントの順序が重要になります。ある属性に一致する最初のパターンが、その属性に適用される唯一のパターンになります。

資格情報の中の属性名が大文字小文字の区別がないので、`attribute_name_pattern` の中の属性名は大文字小文字の区別があってはなりません。

例 - 拡張属性外部認証 C API モジュールによって追加されたすべてのタグ値属性を保存する。

```
[credential-refresh-attributes]
tagvalue_* = preserve
```

例 - `tagvalue_last_refresh_time` 属性を、新しい資格情報にある値で更新する。ただし、`tagvalue_` で始まるその他のすべての属性は保存する。

```
[credential-refresh-attributes]
tagvalue_last_refresh_time = refresh
tagvalue_* = preserve
```

ファイル内の属性の順序が重要であることに注意してください。次の例では、`tagvalue_last_refresh_time` は、まず `tagvalue_*` エントリーにマッチングされ、このエントリーが「preserve」に設定されているために、リフレッシュされません、

```
[credential-refresh-attributes]
tagvalue_* = preserve
tagvalue_last_refresh_time = refresh
```

AZN_ という文字で始まる属性は保存しないでください。このような属性は、通常、許可決定の際に、許可 API によって内部的に使用されます。これらの属性について詳しくは、「*IBM Security Access Manager for Web: Authorization C API Developer Reference*」で説明されています。この資料の中の、資格情報からの属性リストの入手方法の説明を参照してください。

保存およびリフレッシュのためのデフォルトの設定値

WebSEAL 構成ファイルのデフォルトの設定値は次のようになっています。


```
[credential-refresh-attributes]
authentication_level = preserve
tagvalue_* = preserve
```

これらの設定値により、振る舞いは次のようになります。

- 資格情報がリフレッシュされるときにユーザー認証レベルが保存されます。ユーザー・セッション中に、認証強度ポリシー（ステップ認証）が適用されると、ユーザー認証レベルが変更されます。資格情報のリフレッシュ中は、ほとんどの場合、変更された認証レベルを保存する必要があります。

認証レベルを保存する必要がない場合は、構成ファイルのエントリーを次のように変更します。

```
authentication_level = refresh
```

- tagvalue_* エントリーは、名前が tagvalue_ という文字で始まるすべての資格情報属性を保存します。

tagvalue_ という接頭部が付いた属性は、通常、ユーザー情報を資格情報に追加したい外部認証 C API サービスによって提供されます。この接頭部は、WebSEAL が、junction を越えて送る資格情報データを HTTP ヘッダーに挿入するときに、必ず資格情報が含まれるようにするために必要です。

制限

- 資格情報リフレッシュ中に、資格情報属性付与サービス呼び出すことを避けることはできません。一回しか設定できない属性がある場合、(最初の認証時に) 拡張属性外部認証 C API モジュールを使用してその属性を設定します。

資格情報リフレッシュの構成

資格情報リフレッシュを構成するには、以下のステップを実行します。

- 『1. 保存またはリフレッシュする属性の指定』
- 259 ページの『2. ユーザー・セッション ID の使用可能化』
- 259 ページの『3. junction ヘッダーへのサーバー名の配置の使用可能化』

1. 保存またはリフレッシュする属性の指定

手順

- WebSEAL サーバーを停止します。
- WebSEAL 構成ファイルを編集します。

- 保存する属性のエントリーを追加します。例:

```
[credential-refresh-attributes]
my_cred_attribute1 = preserve
my_cred_attribute2 = preserve
```

- リフレッシュするエントリーを次のように追加します。

```
[credential-refresh-attributes]
my_cred_attribute3 = refresh
my_cred_attribute4 = refresh
```

- 必要に応じて、エントリーの順序を使用し、特定のエントリーおよびエントリーのグループを処理します。例えば、属性 special_cred_attr1 を保存し、命

名の構成に `special_cred_attr*` があるその他のすべての属性をリフレッシュするには、以下のエントリーを追加します。

```
[credential-refresh-attributes]
special_cred_attr1 = preserve
special_cred_attr* = refresh
```

2. ユーザー・セッション ID の使用可能化

このタスクについて

WebSEAL インスタンスでできるように、ユーザー・セッション ID が使用可能になっていることを確かめてください。ユーザー・セッション ID が使用可能になっていないと、資格情報リフレッシュ管理コマンドは作動できません。

```
[session]
user-session-ids = yes
```

3. junction ヘッダーへのサーバー名の配置の使用可能化

サーバー名を Junction ヘッダーに追加するように WebSEAL を構成できます。

[header-names] `<header-data>` スタンザ・エントリーを使用して、WebSEAL が、Junction アプリケーションに対する要求に、URI エンコードされた許可 API 管理サーバー名が設定されたヘッダーを追加するように構成します。このエントリーを構成しない場合、WebSEAL は要求に一切ヘッダーを追加しません。

`<header-data>` エントリーの形式は以下のとおりです。

```
[header-names]
<header-data> = <header-name>
```

説明:

`<header-data>`

WebSEAL が要求の `<header-name>` ヘッダーに追加するデータのタイプ。

注: WebSEAL サーバー用の Security Access Manager 許可サーバー名を追加するには値 `server-name` を使用します。

`<header-name>`

データを保持するヘッダーの名前。

以下の値が、デフォルトの WebSEAL 構成ファイルに設定されています。

```
[header-names]
server-name = iv_server_name
```

この設定は、**iv_server_name** という名前のヘッダーを追加して、サーバーの名前を Junction アプリケーションに渡します。この例では、WebSEAL インスタンスが `default-webseald-diamond.subnet1.ibm.com` の場合、WebSEAL は以下のヘッダーを Junction に渡します。

```
iv_server_name:default-webseald-diamond.subnet1.ibm.com
```

通常、デフォルト値 `iv_server_name` が使用されます。ただし、このデフォルト値は、任意の有効なストリングで置き換えることができます。有効なストリングは、A-Z、a-z、0-9、ハイフン (-)、またはアンダースコア () の文字に限られます。

1. `<header-data>` スタンザ・エントリーが、WebSEAL インスタンス用の構成ファイル内の `server-name` の `<header-data>` 値を使用して設定されていることを確認します。例:

```
[header-names]
server-name = iv_server_name
```
2. WebSEAL サーバーを再始動します。

資格情報リフレッシュの使用法

このセクションは、以下のトピックで構成されています。

- 『指定ユーザーの資格情報のリフレッシュ』
- 261 ページの『資格情報リフレッシュのトラブルシューティング』

指定ユーザーの資格情報のリフレッシュ

注: `refresh all_sessions` コマンドは SMS 環境ではサポートされません。代わりに、`sms-session refresh` コマンドを使用します。

WebSEAL サーバーにコマンドを送って、WebSEAL サーバー上の指定ユーザーのすべてのセッションの資格情報リフレッシュ操作を実行するよう指示します。構文は以下のとおりです (1 行で入力します)。

```
pdadmin> server task instance_name-webseald-host_name
refresh all_sessions user_name
```

上記のコマンドは、1 つの連続したコマンド行として入力します。

サーバー名を正しい形式で入手するには、`pdadmin server list` コマンドを使用します。次に `pdadmin` コマンドを入力して、すべてのセッションをリフレッシュします。例えば、`pdadmin` に、管理ユーザー `sec_master` としてログインしているときは、次のように入力します。

```
pdadmin sec_master> server list
default-webseald-diamond.subnet1.ibm.com
default-webseald-cmd
pdadmin sec_master> server task default-webseald-diamond.subnet1.ibm.com
refresh all_sessions brian
DPWWA2043IThe user's credential was updated.
```

`pdadmin` サーバー・タスク・コマンドは、それぞれ、1 つの連続したコマンド行として入力しなければならないことに注意してください。

ユーザーが WebSEAL サーバーにログインしていない場合は、警告メッセージが戻されます。

使用上の注意:

- WebSEAL の資格情報リフレッシュを構成してから、この `pdadmin` コマンドを使用してください。258 ページの『資格情報リフレッシュの構成』を参照してください。
- 資格情報をリフレッシュするそれぞれのユーザーごとに、別個に `pdadmin` コマンドを実行する必要があります。一時に複数のユーザーの資格情報をリフレッシュすることはできません。

- このコマンドを起動するユーザーは、`/WebSEAL/hostname_instance_name` サーバー・オブジェクト上に、サーバー `admin` (`s ACL` ビット) 許可を持っていないければなりません。この許可によって、無許可ユーザーは、資格情報リフレッシュ操作を実行できなくなります。

`hostname_instance_name` サーバー・オブジェクトの名前は、サーバー名とは違うことに注意してください。サーバー・オブジェクトの正確な名前を調べるには、**pdadmin object list** を使用してください。例えば、**pdadmin** に、管理ユーザー `sec_master` としてログインしているときは、次のように入力します。

```
pdadmin sec_master> object list /WebSEAL
/WebSEAL/cmd-default
/WebSEAL/diamond.subnet1.ibm.com-default
```

資格情報リフレッシュのトラブルシューティング

問題:

新規グループ記入項目がユーザー・レジストリー内のユーザーの情報に追加されているとき、資格情報リフレッシュ・コマンドが新規エントリーを取得しません。

ソリューション:

ユーザー・レジストリーの一部は、キャッシュに入れられた情報を維持しています。キャッシュは定期的に更新されます。資格情報リフレッシュが正常に行われるようにするには、その前にキャッシュの更新が行われていなければなりません。同様に、複製された LDAP ユーザー・レジストリーを使用しているとき、複製されたレジストリーへの更新は即時に行われません。30 秒待ってから、資格情報リフレッシュを再び試行してください。詳細については、256 ページの『キャッシュに入れられた資格情報のリフレッシュ』を参照してください。

第 13 章 外部認証インターフェース

この章では、外部認証インターフェース (EAI ともいう) について説明します。

この章で扱うトピックは以下のとおりです。

- 『外部認証インターフェースの概要』
- 『外部認証インターフェースのプロセス・フロー』
- 267 ページの『外部認証インターフェースの構成』
- 275 ページの『外部認証インターフェース HTTP ヘッダーのリファレンス』
- 276 ページの『既存の WebSEAL 機能を持つ外部認証インターフェースの使用』

外部認証インターフェースの概要

Security Access Manager は、外部認証インターフェースを提供しており、これによって WebSEAL 認証プロセスを拡張できます。外部認証インターフェースによって、独立したリモート・サービスで WebSEAL 認証プロセスを扱うことができます。外部認証インターフェース・サービスから戻される ID 情報を使用して、ユーザー資格情報を生成します。

この拡張認証機能は、Web セキュリティー外部認証 C API が提供する既存のカスタム認証モジュール機能と似ています。ただし、外部認証インターフェースを使用すると、認証モジュール・インターフェース経由ではなく HTTP 応答ヘッダーでユーザー ID 情報を戻す点が異なります。

外部認証インターフェースを使用する場合、認証操作は、リモートの junction 先サーバー上にあるカスタム・アプリケーションによって WebSEAL の外部で実行されます。カスタム認証アプリケーションの設計、方法論、コードは、アプリケーション開発者に全面的に任されます。この開発者向け解説書では、カスタム認証操作の構築する手順については説明していません。ただし、このアプリケーションの要件は、カスタム認証プロセスの結果として作成された ID 情報を、特別に名付けられた HTTP 応答ヘッダーに戻すことです。

外部認証インターフェースのプロセス・フロー

以下の図および詳細ステップに、外部認証インターフェースの認証のためのプロセス・フローを示します。このプロセス・フロー・シナリオの例のコンポーネントには以下が含まれます。

- WebSEAL。
- 外部認証インターフェースを使用した外部認証アプリケーションがある junction 先サーバー。

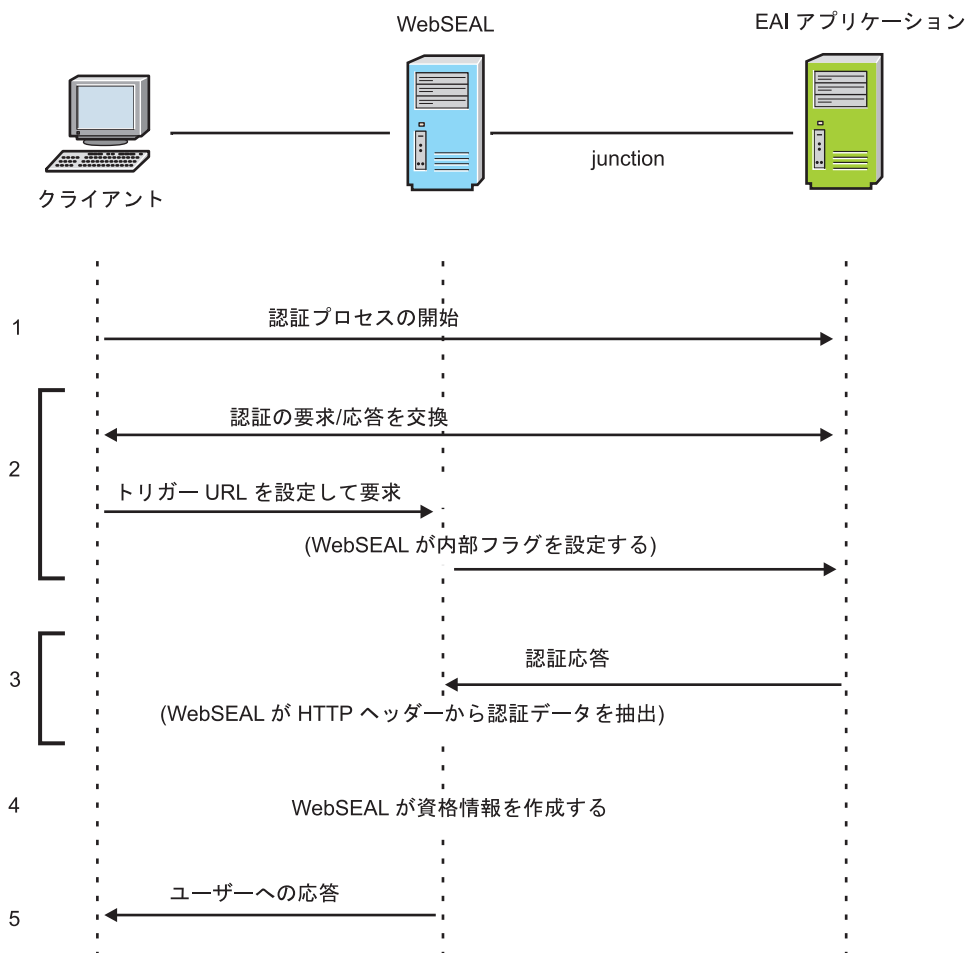


図 18. 外部認証インターフェースのプロセス・フロー

1. 認証プロセスが開始されます。

認証プロセスが開始される可能性はたくさんあります。典型的な例は以下のとおりです。

- 非認証ユーザーが保護リソースを要求する。
- WebSEAL が要求をインターセプトして、カスタマイズされた `login.html` 応答ページにリダイレクトを戻す。

`login.html` ページは、外部認証アプリケーションへの「送信 (submit)」リンクが含まれるようにカスタマイズされています。

- ユーザーは、フォームにログイン情報 (ユーザー名およびパスワード) を入力し、「送信 (submit)」リンクをクリックして、外部認証アプリケーションにデータを送信します。

認証プロセスを開始するその他の例としては、以下のものがあります。

- 外部認証アプリケーションへの適切なリンクの手動によるタイプ入力。
- CDSSO 環境におけるサインオン要求。
- キャッシュ・ページ。

- IBM Tivoli Federated Identity Management シナリオでは、ユーザーは、自分用に提供されている ID を取得するために、サービス・プロバイダーから外部認証アプリケーションにリダイレクトされる。

注: 隠し構成オプションによって、URL への正常なログインをリダイレクトすることを EAI ヘッダーに優先設定できます。この機能を使用可能にするには、以下のオプションおよび値を **[eai]** スタンザに追加します。

```
eai-redir-url-priority = yes
```

2. 証明書要求および応答交換。

認証プロセスは、外部認証アプリケーションとクライアントの間で多数の交換が必要な場合があります。交換は、(インターセプトされるのではなく) WebSEAL によってストリームされます。

外部認証アプリケーションに対する最後の証明書要求は、個別の URL に送信される必要があります。例えば、この URL には、state=perform-login などのログイン・タスクを示す照会ストリングが含まれています。この最後の URL は、トリガー URL として WebSEAL 構成ファイルに一部または全体が指定されます。WebSEAL は、このトリガー URL に対する各要求を調査します。

トリガー URL が検出されると、WebSEAL は HTTP ヘッダー (WebSEAL 構成ファイルに指定) にある認証データに対応する応答を調べます。

WebSEAL は EAI ログアウトをサポートします。EAI から戻される HTTP ヘッダーにより、セッションをログアウトするよう WebSEAL に指示できます。このヘッダーは pdadmin server task コマンド行入力をエミュレートしているため、同じ名前の隠し WebSEAL pdadmin コマンドに類似しています。ヘッダーの構文は am-eai-server-task: terminate session <user-sess-id> です。ここで、terminate session は変換不能なキーワード・ペア、<user-sess-id> は、pdadmin を使用して terminate session コマンドを実行するのに使用されるものと同じフォーマットおよび内容のユーザー・セッション ID です。

交換の例 1:

- ユーザーはカスタム・ログイン・ページの「送信 (submit)」リンクをクリックします。このリンクはトリガー URL です。
- 要求内のトリガー URL を認識すると、WebSEAL は対応する応答の中に認証データを探します。
- 外部認証アプリケーションはユーザーを認証し、その応答で特殊 HTTP ヘッダーに認証データを取り込みます。

交換の例 2:

- 外部認証アプリケーションでは、必要なログイン情報を取得するために、ユーザーと複数回数交換を行う必要があります。
- 外部認証アプリケーションに対する各要求は、トリガー URL を使用します。したがって、WebSEAL は応答ごとに認証データを検索します。
- WebSEAL は、外部認証インターフェースの HTTP ヘッダーで戻された認証データに対応する各応答を調べます。

- 認証が行われていない場合、各応答の HTTP ヘッダーは空です。WebSEAL は、アクションを起こさずに要求および応答をストリームし続けます。
- 交換を複数回数行ったら、外部認証アプリケーションは必要なすべてのログイン情報を受信します。外部認証アプリケーションがユーザーを認証し、その最後の応答で特殊 HTTP ヘッダーに認証データを取り込みます。

交換の例 3

- 外部認証アプリケーションでは、必要なログイン情報を取得するために、ユーザーと複数回数交換を行う必要があります。
- 外部認証アプリケーションに対する各要求は、トリガー URL と一致しない URL を使用します。したがって、WebSEAL は対応する応答ごとには認証データを検索しません。
- WebSEAL は、アクションを起こさずに要求および応答をストリームします。
- 外部認証アプリケーションに対する最後の要求でトリガー URL を使用します。
- この最後の要求内のトリガー URL を認識すると、WebSEAL は対応する応答の中に認証データを探します。
- 外部認証はユーザーを認証し、その最後の応答で特殊 HTTP ヘッダーに認証データを取り込みます。

3. 認証応答。

WebSEAL は対応する応答を調査し、HTTP ヘッダー内で認証データを検出します。

4. WebSEAL は認証データを使用して、ユーザーの資格情報を作成します。
5. WebSEAL は以下の優先順位に従って、ユーザーに応答を送信します。
 - a. 自動リダイレクトが使用可能な場合、ユーザーは WebSEAL 構成ファイルに指定された場所にリダイレクトされます。

277 ページの『WebSEAL 指定 (自動) リダイレクト』を参照してください。

- b. 外部認証アプリケーションからの応答にストリーミング・フラグが含まれている場合、WebSEAL は元の応答をクライアントに向けてストリームします。

274 ページの『外部認証インターフェース - 認証フラグ』を参照してください。

- c. 最初の要求がキャッシュされていた場合、ユーザーに対する要求は再処理されます。

276 ページの『外部認証インターフェースによる要求キャッシング』を参照してください。

- d. 外部認証アプリケーションからの応答にリダイレクト URL ヘッダーが含まれている場合、ユーザーはその URL によって指定される場所にリダイレクトされます。

277 ページの『外部認証インターフェース指定リダイレクト』を参照してください。

- e. それ以外の場合、WebSEAL は標準の login_success.html ページで応答します。

79 ページの『静的 HTML サーバー応答ページ』を参照してください。

外部認証インターフェースの構成

このセクションでは、WebSEAL で外部認証インターフェースを使用するように構成する方法について説明します。

- 『外部認証インターフェースの使用可能化』
- 268 ページの『認証処理の開始』
- 268 ページの『外部認証インターフェース・トリガー URL の構成』
- 269 ページの『認証データの HTTP ヘッダー名』
- 270 ページの『特殊 HTTP ヘッダーからの認証データの抽出』
- 271 ページの『資格情報の生成方法』
- 271 ページの『外部認証インターフェースの資格情報交換』
- 273 ページの『外部認証アプリケーションの作成方法』
- 275 ページの『外部認証インターフェース HTTP ヘッダーのリファレンス』

外部認証インターフェースの使用可能化

このタスクについて

外部認証インターフェース機能を使用可能または使用不可にするには、WebSEAL 構成ファイルの **[eai]** スタンザにある **eai-auth** スタンザ・エントリーを使用します。外部認証インターフェースは、HTTP、HTTPS、またはその両方を使用してインプリメントできます。

デフォルトでは、外部認証インターフェース認証は使用不可になっています。

外部認証インターフェースを構成するには、次のようにします。

手順

1. WebSEAL サーバーを停止します。
2. WebSEAL 構成ファイルを編集します。**[eai]** スタンザで、ご使用のネットワーク環境でサポートするプロトコルを指定します。サポートするプロトコルを、次の表に示します。

表 26. 外部認証インターフェースの構成

サポートするプロトコル	構成ファイル・エントリー
HTTP	eai-auth = http
HTTPS	eai-auth = https
HTTP および HTTPS	eai-auth = both
外部認証インターフェースの使用不可化 (デフォルト)	eai-auth = none

例えば、両方のプロトコルをサポートするには次のようにします。

```
[eai]
eai-auth = both
```

3. WebSEAL サーバーを再始動します。

タスクの結果

eai-auth = none (使用不可) の場合、他のすべての構成済み外部認証インターフェース関連スタンザ・エントリーは影響を与えません。

認証処理の開始

このタスクについて

通常、外部認証インターフェース認証は、リダイレクト・コマンドまたは外部アプリケーション・ページ内のカスタム・リンクから開始することができます。外部認証インターフェース・シナリオでは、WebSEAL は認証プロセスを開始する組み込みメソッドを提供していません。WebSEAL は、特別なプロンプトもログイン・ページも提供しません。

手順

- WebSEAL の既存の login.html フォームを変更して、外部認証アプリケーションへのカスタム・リンクを含めることができます。再認証および認証強度 (ステップアップ) をサポートするには、login.html フォームの変更が必要です。

279 ページの『外部認証インターフェースによるログイン・ページおよびマクロのサポート』を参照してください。

- また、ローカル応答リダイレクトをインプリメントして、クライアント要求に対するサーバー応答を処理することもできます。103 ページの『ローカル応答リダイレクト』を参照してください。

外部認証インターフェース・トリガー URL の構成

外部認証インターフェースの認証プロセスでは、複数の要求と応答の交換をサポートしています。WebSEAL サーバーの効率およびセキュリティのために、これらの交換は一般に WebSEAL によってストリームされます。WebSEAL は、要求内に特別なトリガー URL が指定されている場合のみ、この交換をインターセプトします。

トリガー URL は、WebSEAL 構成ファイルにあるサーバー相対 URL または絶対 URL です。トリガー URL は通常、外部認証アプリケーションの認証を要求します。例えば、トリガー URL は、カスタマイズされたログイン・ページの特別なリンクに含まれる外部認証アプリケーションを指す URL などです。

WebSEAL は、要求内でトリガー URL を検出すると、対応する応答をインターセプトし、特別な HTTP ヘッダーにある認証データかどうかを調べます。

トリガー URL スtring

- 標準的なワイルドカード・パターンを使用できます。パターン・マッチングは、ASCII ベース・ストリングの場合のみ適します。
- パターン・マッチングを使用する場合は、ASCII フォーマットにする必要があります。要求内のマッチング URL も ASCII フォーマットでなければなりません。
- WebSEAL が要求と応答の交換をインターセプトする回数を制限するため、構成された URL においてできるだけ固有の値にする必要があります。

トリガー URL ストリングは、WebSEAL 構成ファイルの **[eai-trigger-urls]** スタンザの **trigger** スタンザ・エントリーで指定します。

仮想ホスト junction

- プロトコル、仮想ホスト名、およびポートが仮想ホストの定義と一致する場合は、トリガーを突き合わせます。
- 仮想ホスト定義に一致しないような通常の WebSEAL junction トリガーは使用しないでください。通常の WebSEAL junction は、仮想ホスト junction トリガーを使用しません。

表 27. 外部認証アプリケーションに対する認証要求の例:

junction タイプ	URL	対応するトリガー URL
標準	http://webseal.example.com/ eai-jct/login.asp?url=/ return_authn_data.asp	[eai-trigger-urls] trigger = /eai-jct/login.asp*authn*
仮想ホスト	http://vhj.webseal.example.com/ login.asp?url=/return_authn_ data.asp	[eai-trigger-urls] trigger = http://vhj.webseal. example.com/login.asp*authn*

認証データの HTTP ヘッダー名

外部認証アプリケーションから戻される認証データを含む HTTP ヘッダーの名前を指定する必要があります。

認証データを保持する HTTP ヘッダーには、以下の 4 つのカテゴリーがあります。

• 特権属性証明書 (PAC) 形式

PAC は、ID 情報を示すのに使用される ASN.1 データ構造です。PAC 形式で WebSEAL に戻された認証データは、直接資格情報に変換できます。

• WebSEAL ユーザー ID 構造

WebSEAL ユーザー ID 構造は、WebSEAL のデフォルト組み込み認証モジュールによって生成される構造と同じ構造です。ユーザー ID 形式タイプが使用されている場合、情報は **eaiauthn** 認証モジュールによって処理され、資格情報は Security Access Manager 許可 API によって作成されます。

• SMS セッション ID

SMS セッション ID は、Session Management Server によって管理される分散セッションに使用されます。353 ページの『複数の DNS ドメインでのセッション共有』を参照してください。

- 共通

共通ヘッダー・カテゴリには追加情報が含まれており、PAC またはユーザー ID のどちらの形式でも使用できます。

これらの特殊ヘッダーに関する詳細は、275 ページの『外部認証インターフェース HTTP ヘッダーのリファレンス』を参照してください。

WebSEAL 構成ファイルの **[eai]** スタンザを使用して、外部認証インターフェース・サーバーから戻される認証データを含む HTTP ヘッダーの名前を指定します。ヘッダー名はカスタマイズできます。構成したヘッダー名を使用するには、カスタム外部認証インターフェース認証モジュールを作成する必要があります。

以下の例は、WebSEAL 構成ファイルで使用されるデフォルトのヘッダー名を示しています。

PAC ヘッダー:

```
[eai]
eai-pac-header = am-eai-pac
eai-pac-svc-header = am-eai-pac-svc
```

ユーザー ID ヘッダー:

```
[eai]
eai-user-id-header = am-eai-user-id
eai-auth-level-header = am-eai-auth-level
eai-xattrs-header = am-eai-xattrs
```

SMS セッション ID

```
[eai]
eai-session-id-header = am-eai-session-id
```

共通ヘッダー:

```
[eai]
eai-flags-header = am-eai-flags
eai-redirect-url-header = am-eai-redirect-url
```

eai-flags-header 共通ヘッダーの使用の詳細については、274 ページの『外部認証インターフェース - 認証フラグ』を参照してください。

eai-redirect-url-header 共通ヘッダーの使用の詳細については、277 ページの『外部認証インターフェース指定リダイレクト』を参照してください。

特殊 HTTP ヘッダーからの認証データの抽出

このタスクについて

要求内にトリガー URL を検出すると、WebSEAL は対応する応答の特殊ヘッダーを調査します。

特殊 HTTP ヘッダーには、カスタム外部認証アプリケーションによって提供された認証データが含まれています。PAC ヘッダーまたはユーザー ID ヘッダーが存在する場合、WebSEAL は、ヘッダーから認証データを抽出してユーザーの資格情報を

作成します。セッション ID ヘッダーが存在する場合、WebSEAL は指定されたセッションを Session Management Server から取得します。

WebSEAL は以下の特定の手順に従って、特殊 HTTP 認証ヘッダーを処理します。

手順

1. セッション ID ヘッダーが存在する場合、他の認証ヘッダーよりも優先されます。
2. 両方のヘッダーが存在する場合、PAC データが優先され、ユーザー ID データは無視されます。
3. どちらのヘッダーも存在しない場合、応答はクライアントにストリームされます。この動作により、外部認証アプリケーションでも認証エラー処理を実行できます。
4. PAC またはユーザー ID ヘッダーのどちらかが存在し、ヘッダーの値が NULL か破損している場合、エラーが戻されます。外部認証インターフェース・サーバーが正しく構成されていない場合、このようなエラーが発生する可能性があります。

資格情報の生成方法

WebSEAL は、PAC ヘッダー・データから直接資格情報を作成できます。許可 API は、ユーザー ID ヘッダー・データ用の資格情報を作成します。

他の認証データは、ユーザー ID 認証データから資格情報を作成するときに、WebSEAL システム自体によって提供されます。WebSEAL は、資格情報の作成が必要なクライアント・システムについての追加情報を保有しています。この情報は、外部認証インターフェースの認証データを使用して資格情報を生成する場合に提供されます。

これらの値の一部は、ヘッダー・データに対して拡張属性を使用する **eaiauthn** モジュールによってオーバーライドされることがあります。

表 28. WebSEAL が提供する補足の資格情報データ

フィールド	ソース	外部認証インターフェースが値をオーバーライドできるかどうか
クライアント IP アドレス	最初のクライアント要求から派生	はい
ブラウザー情報	最初のクライアント要求から派生	はい
レジストリー・タイプ	現在の WebSEAL 構成によって決定	いいえ
ドメイン	現在の WebSEAL 構成によって決定	いいえ

外部認証インターフェースの資格情報交換

WebSEAL では、前に認証済みのユーザーは、外部認証インターフェース・トリガー URL を使用して認証を再度要求し、新規のセッションを確立できます。

WebSEAL は、古いセッション・キャッシュ・エントリーを削除し、そのユーザー

の新規の資格情報を含む新規セッション・キャッシュ・エントリーを作成して (資格情報交換)、ユーザーに新規のセッション鍵を提供します。

外部認証インターフェースの資格情報交換に関する操作条件

- 前に認証済みのユーザーがトリガー URL を使用して要求を作成する場合、その要求を外部認証アプリケーションにパススルーすることができます。

注: 前のバージョンの外部認証インターフェースでは、前に認証済みのユーザーは強制的にログアウトされ、トリガー URL を使用して要求を作成するときに再度ログインしていました。

- ユーザー要求に対する外部認証インターフェースの応答に認証データが含まれ、ユーザーのセッション・キャッシュ・エントリーで認証強度ポリシー (ステップアップ) または再認証にフラグが立っている場合、WebSEAL はステップアップ・プロセスまたは再認証プロセスを実行します。そのユーザーの既存のセッション・キャッシュ (および資格情報) は交換されません。
- ユーザー要求に対する外部認証インターフェースの応答に認証データが含まれ、ユーザーのキャッシュ・エントリーでステップアップまたは再認証にフラグが立っていない場合、以下の処理が行われます。
 - 既存のセッション・キャッシュ・エントリーは削除され、ユーザーの新規の資格情報を含む新規エントリーと交換されます。
 - ユーザーがセッション Cookie を使用してセッション状態を維持する場合、新規セッション鍵が作成され、ユーザーに戻されます。
 - ユーザーが SSL セッション ID または HTTP ヘッダーを使用してセッション状態を維持する場合、既存のセッション鍵が再使用されます。
 - フェイルオーバー Cookie を使用する場合、新規のフェイルオーバー Cookie が作成され、ユーザーに戻されます。
 - ユーザー・セッション ID を使用する場合、WebSEAL セッション ID に対するユーザー・セッション ID マッピングが更新されます。
 - LTPA Cookie を使用する場合、新規の LTPA Cookie が作成され、ユーザーに戻されます。

外部認証インターフェースの資格情報交換機能は、例えば Liberty のフェデレート機能が提供するアカウントのリンク機能をサポートするために重要です。Tivoli Federated Identity Manager 環境では、Liberty フェデレート機能 (Liberty Alliance Project) を実行するために、前に認証済みのユーザーを再認証する機能が必要です。フェデレート操作によって、サービス・プロバイダーのローカル・アカウントとアイデンティティー・プロバイダーのアカウントとのリンクが可能になります。

これを実行するには、ユーザーはまずユーザーのサービス・プロバイダーにサインインし、次にユーザーのアカウントとアイデンティティー・プロバイダーをリンクすることに同意する必要があります。一度フェデレート操作が発生すると、ブラウザーのフォーカスはサービス・プロバイダーに戻り、ユーザーの資格情報はアイデンティティー・プロバイダーが生成した新規の資格情報で更新されます。

ユーザー ID の検証

EAI アプリケーションは、前の認証済みセッションの新しい認証情報を返すことによって、ユーザーを再認証することができます。デフォルトで WebSEAL は、この新しい認証情報を検証しません。ただし、その後の EAI 認証中にユーザー ID が変更されていないことを検証するように WebSEAL を構成することができます。

WebSEAL は、プリンシパル名 (`azn_cred_principal_name` 属性) を使用してユーザー ID を検証します。新規作成された資格情報に含まれているプリンシパル名は、既存の資格情報に含まれているプリンシパル名と比較されます。2 つの名前が異なる場合は、検証プロセスは失敗となり、WebSEAL はユーザーに認証エラーを返します。

その後の EAI 認証操作でユーザー ID を検証するには、`eai-verify-user-identity` スタンザ・エントリーを `yes` に設定します。このエントリーは、WebSEAL 構成ファイルの `[eai]` スタンザにあります。

```
[eai]
eai-verify-user-identity = yes
```

外部認証アプリケーションの作成方法

外部認証アプリケーションの設計、方法論、コードは、全面的にアプリケーション開発者に任されます。この開発者向け解説書では、この認証操作の構築手順については説明しません。

ただし、カスタム・アプリケーションを開発する際には、外部認証インターフェースの操作に関する以下の条件を考慮する必要があります。

- 外部認証インターフェース・サーバーは WebSEAL に junction されます。
- カスタム認証プロセスの結果として作成された ID 情報は、特別に名前が付いた HTTP 応答ヘッダー (WebSEAL 構成ファイル内に構成) で WebSEAL に戻す必要があります。
- 複数ステップの認証が許可されます。
- 非認証ユーザーが外部認証アプリケーションを使用できる必要があります。
- WebSEAL は、ユーザー・レジストリーの資格情報を検査します。したがって、外部認証アプリケーションが WebSEAL と同じレジストリーを共用するか、外部認証アプリケーションが WebSEAL ユーザー・レジストリー内のエントリーと一致するユーザー情報を戻すかのどちらかを行う必要があります。

外部認証インターフェースのデモンストレーション・プログラム

外部認証インターフェースのデモンストレーション・プログラムは、WebSphere Application Server 用に作成された J2EE アプリケーションです。

このプログラムは、外部認証インターフェースとローカル応答リダイレクトの両方の利点を活用し、企業がカスタム・アプリケーションを作成して WebSEAL junction を使用してサーバー応答ページ処理や認証サービスを提供する方法をデモンストレーションします。プログラムは、ログインごとに同じユーザーに対して資格情報を生成する単一ユーザー名マッピング機能を実行します。

デモンストレーションは、単純モードと拡張モードの 2 つのモードで稼働します。単純デモンストレーションでは、HTTP ヘッダーを使用してユーザーの認証データを提供します。拡張デモンストレーションでは、PAC でユーザーの認証データを提供します。デモンストレーション・モードの選択は、ログインの際にユーザーが適切なラジオ・ボタンを選択して行います。

デモンストレーション・プログラムは、Security Access Manager Web Security ADK (PDWebADK) の一部として配布され、Security Access Manager Web Security Runtime と Security Access Manager Application Development Kit (ADK) を前提条件として必要とします。デモンストレーション・プログラムは C:\Program Files\Tivoli\PDWebRTE\esai_demo ディレクトリー (Windows) にインストールされます。詳しくは、同じディレクトリーの README.html ファイルを参照してください。

PDWebADK について詳しくは、*IBM Security Access Manager for Web: Web Security Developer Reference* を参照してください。

外部認証インターフェース - 認証フラグ

EAI アプリケーションは、認証が正常に実行されると、応答を作成してその応答をトリガー URL に返します。WebSEAL は、トリガー URL 応答内でこの認証情報を検出します。この応答に認証フラグを付加し、WebSEAL による認証処理を制御することができます。

これらの認証フラグは、HTTP ヘッダーに含まれています。WebSEAL 構成ファイルの [esai] スタンザにある **esai-flags-header** スタンザ・エントリーを使用して、フラグ・ヘッダーの名前を指定します。

WebSEAL は以下のフラグをサポートします。

stream デフォルトで WebSEAL は、EAI 生成の応答を認証操作の WebSEAL 生成応答で置き換えます。このデフォルトの動作をオーバーライドし、EAI 生成応答をストリーミングしてクライアントに戻すように WebSEAL を構成できます。つまり、EAI が正常に認証された後、WebSEAL は応答から EAI 固有のヘッダーを取り除き、ストリーミングしてそのクライアントに戻すことができます。

この EAI 応答ストリーミングを実行するには、フラグ・ヘッダーに **stream** フラグを含める必要があります。

EAI フラグ・ヘッダーの例:

```
am-esai-flags: stream
```

esai-flags-header 構成エントリーは、フラグを含んでいる HTTP ヘッダーの名前を指定します。例:

```
[esai]
esai-flags-header = am-esai-flags
```

外部認証インターフェース HTTP ヘッダーのリファレンス

表 29. PAC ヘッダー

PAC ヘッダー				
説明	スタンザ・エントリー	デフォルト・ヘッダー名	必須	注
PAC	[eai] eai-pac-header	am-eai-pac	はい	認証データは PAC 形式です。資格情報に直接変換されます。 このヘッダーは、ユーザー ID ヘッダーよりも優先されます。 このヘッダーは、応答ヘッダー内のその他のヘッダーの前に置いてください。
PAC サービス ID	[eai] eai-pac-svc-header	am-eai-pac-svc	いいえ	PAC を資格情報に変換するために使用するサービス ID。 サービス ID が指定されていない場合、デフォルトの PAC サービスが使用されます。

表 30. ユーザー ID ヘッダー

ユーザー ID ヘッダー				
説明	スタンザ・エントリー	デフォルト・ヘッダー名	必須	注
ユーザー ID	[eai] eai-user-id-header	am-eai-user-id	はい	資格情報を生成する対象のユーザーの ID。 このヘッダーは、HTTP 応答内の他のすべてのヘッダーよりも前になければなりません。
認証レベル	[eai] eai-auth-level-header	am-eai-auth-level	いいえ	生成された資格情報の認証強度レベル。 値が指定されていない場合、デフォルト値の 1 が使用されます。
拡張属性リスト	[eai] eai-xattrs-header	am-eai-xattrs	いいえ	拡張属性として資格情報に追加する HTTP ヘッダー名のコンマ区切りリスト。 外部認証 C API で作成されたカスタム認証モジュールによって同じ名前の属性が指定された場合、そのカスタム・モジュールの属性は HTTP ヘッダーの属性よりも優先されます。

表 31. セッション ID ヘッダー

セッション ID ヘッダー				
説明	スタンザ・エントリー	デフォルト・ヘッダー名	必須	注
セッション ID	[eai] eai-session-id-header	am-eai-session-id	はい	Session Management Server によって管理される分散セッションの ID。

表 32. 共通ヘッダー

共通ヘッダー				
説明	スタンザ・エントリー	デフォルト・ヘッダー名	必須	注
リダイレクト URL	[eai] eai-redirect-url-header	am-eai-redirect-url	いいえ	WebSEAL に要求がキャッシュされていない場合、または自動リダイレクトが使用不可の場合のみ使用されます。 認証が成功したときにクライアントをリダイレクトする URI を指定します。 URI を指定しないと、「ログイン・サクセス」ページが戻されます。
フラグ・ヘッダー	[eai] eai-flags-header	am-eai-flags	いいえ	サポートされている唯一のフラグは stream です。 例: am-eai-flags: stream

既存の WebSEAL 機能を持つ外部認証インターフェースの使用

このセクションは、以下のトピックで構成されています。

- 『外部認証インターフェースによる要求キャッシング』
- 277 ページの『外部認証インターフェースによる認証後リダイレクト』
- 277 ページの『外部認証インターフェースによるセッション処理』
- 277 ページの『外部認証インターフェースによる認証強度レベル』
- 278 ページの『外部認証インターフェースによる再認証』
- 279 ページの『外部認証インターフェースによるログイン・ページおよびマクロのサポート』

外部認証インターフェースによる要求キャッシング

外部認証インターフェース認証でサーバー・サイド要求キャッシングが発生するのは、WebSEAL が以下の処理の結果としてログイン・プロンプトを戻す場合です。

- 非認証ユーザーが保護リソースを要求した場合。
- 認証済みユーザーが再認証によって保護されているリソースを要求した場合。

- 認証済みユーザーが認証強度ポリシー (ステップアップ) によって保護されているリソースを要求した場合。

これらのイベントのいずれかが発生すると、WebSEAL は最初の要求をキャッシュに入れます。外部認証アプリケーションとのあらゆる認証の対話の間に、WebSEAL はキャッシュされた要求を保存します。WebSEAL は、認証が成功したときのみ、キャッシュされた要求を再処理します。

外部認証インターフェースを使用した認証に対する標準の要求キャッシングをサポートする場合、変更は必要ありません。

233 ページの『サーバー・サイド要求キャッシング』を参照してください。

外部認証インターフェースによる認証後リダイレクト

WebSEAL 指定 (自動) リダイレクト

ユーザー・ログイン中の自動リダイレクトが WebSEAL で使用可能である場合、外部認証インターフェース認証が成功するとクライアントは指定されたリソースにリダイレクトされます。

```
[enable-redirects]
redirect = ext-auth-interface
```

229 ページの『認証後の自動リダイレクト』を参照してください。

外部認証インターフェース指定リダイレクト

外部認証アプリケーションを作成して、リダイレクト URL を指定した認証応答で特殊 HTTP ヘッダーを送信できます。認証が成功すると、クライアントはその URL にリダイレクトされます。

このオプション・ヘッダーは、特殊外部認証インターフェース・ヘッダーと同じ方法で構成されます (269 ページの『認証データの HTTP ヘッダー名』を参照)。

例:

```
[eai]
eai-redirect-url-header = am-eai-redirect-url
```

外部認証インターフェースによるセッション処理

外部認証インターフェース認証には、セッションの維持、セッション Cookie の処理、セッション・キャッシュ・パラメーターの構成を行う既存のオプションが適用されます。

外部認証インターフェースによる認証強度レベル

外部認証インターフェース認証では、認証強度ポリシー (ステップアップ認証) がサポートされています。

```
[authentication-levels]
level = ext-auth-interface
```

200 ページの『認証強度ポリシー (ステップアップ)』を参照してください。

認証強度レベルを、外部認証インターフェース・モジュールによって実行される認証と関連付けることができます。この認証レベルを指定するために、外部認証インターフェース・モジュールはオプションの HTTP ヘッダーを戻すことができます。

このヘッダーは、その他の特殊外部認証インターフェース・ヘッダーと同じ方法で構成されます (269 ページの『認証データの HTTP ヘッダー名』を参照)。

例:

```
[eai]
eai-auth-level-header = am-eai-auth-level
```

認証強度レベルの値は、ID 構造およびその結果として作成される資格情報の属性となります。認証強度レベル属性によって、ステップアップ認証機能をインプリメントできます。それには、単一の外部認証インターフェース・サーバーで複数の外部認証インターフェース認証モジュールを操作する必要があります。モジュールごとに異なる認証方式を処理できます。

認証強度レベルが存在しないか、空の値が含まれている場合、認証レベルの割り当てにはデフォルトのメカニズムが使用されます。

外部認証インターフェース認証でステップアップ認証を使用可能にしている場合、標準の WebSEAL ログイン・ページを適切に変更する必要があります。279 ページの『外部認証インターフェースによるログイン・ページおよびマクロのサポート』を参照してください。

外部認証インターフェースによる再認証

外部認証インターフェース認証では、再認証がサポートされます。

再認証では、クライアントが再認証するのに使用する方式とクライアントが最初に認証したときに使用した方式が同じでなければなりません。WebSEAL がカスタム外部認証アプリケーションから認証応答を受信すると、(他の再認証処理と同様に) 以下を確認するための検査が行われます。

- 使用される認証方式が最初の資格情報を作成するのに使用されたものと同じであること
- ユーザー名が一致していること
- すべての外部認証インターフェース指定認証レベルが既存レベルと一致していることが検査済みであること

外部認証インターフェース認証で再認証を使用可能にしている場合、標準の WebSEAL ログイン・ページを適切に変更する必要があります。279 ページの『外部認証インターフェースによるログイン・ページおよびマクロのサポート』を参照してください。

外部認証インターフェースによるログイン・ページおよびマクロのサポート

外部認証インターフェース・サーバーへのリダイレクトで認証を実行したり、ユーザーが外部認証インターフェース・サーバーとの認証交換を開始する場合にクリックするリンク（またはボタン）を挿入するために、WebSEAL ログイン・ページを変更することができます。この変更したログイン・ページは、再認証または外部認証インターフェースへのステップアップを使用可能にする場合に必要です。

外部認証インターフェース指定マクロ (%EIAUTHN%) を使用して、`certlogin.html` および `stepuplogin.html` ログイン・フォームのセクションを個別に追加またはマスクすることができます。認証方式（マクロ名で示される）が有効な場合、マクロが管理するフォーム内のセクションが表示されます。認証方式が有効でない場合、マクロはコメント開始区切り文字（`<!--`）に置き換えられます。フォーム内のすべての後続の情報は、コメント終了区切り文字（`-->`）に達するまで、コメントとして扱われます。

ステップアップで照会ストリング内の引数として必要な認証レベルの引き渡しを容易にするために、WebSEAL は別のマクロ (%AUTHNLEVEL%) を `stepuplogin.html` ログイン・フォームに引き渡します。

これらのマクロはどちらも、デフォルトのログイン・フォームには存在しません。マクロは手動で追加する必要があります。

また、ローカル応答リダイレクトをインプリメントして、クライアント要求に対するサーバー応答を処理することもできます。

クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定

このタスクについて

WebSEAL 構成ファイルの `[session]` スタンザ内の **timeout** スタンザ・エントリーでは、WebSEAL セッション・キャッシュに保管されているすべてのクライアント・セッション情報の最大存続時間タイムアウト値をグローバルに設定します。このグローバル存続時間値をクライアントごとの存続時間値でオーバーライドすることができます。クライアントごとの存続時間値は、外部認証インターフェース・サービスの認証応答フォームのヘッダーとして提供されます。この値は WebSEAL によって抽出され、ユーザーの資格情報に拡張属性として保管されます。

WebSEAL は、クライアント固有のタイムアウト情報を外部認証インターフェースの認証応答のヘッダーの値として受信します。WebSEAL は、このヘッダーの値を使用して、そのクライアントの新規セッション・キャッシュ・エントリーの存続時間タイムアウトを設定します。この値は、**timeout** スタンザ・エントリーの値をオーバーライドします。

この値は、1970 年 1 月 1 日 00:00:00 UTC 以降の秒数で表された絶対時間として示す必要があります。例えば、UNIX `time ()` 関数の出力は、この絶対時間の値を正しいフォーマットで示します。

以下のステップは、クライアント固有のキャッシュ・エントリー存続時間タイムアウト値の設定に必要な構成を要約しています。

手順

1. カスタム外部認証インターフェース・プログラムを構成して、認証応答で、そのクライアントに適したセッション・キャッシュ存続時間タイムアウト値を含む HTTP ヘッダーを提供します。このヘッダーに必須の名前は次のとおりです。

```
am_eai_xattr_session_lifetime
```

注: この特殊ヘッダーの名前は構成できません。
以下に例を示します。

```
am_eai_xattr_session_lifetime:1129225478
```

2. カスタム外部認証インターフェース・プログラムを構成して、拡張属性値を含む HTTP ヘッダー名のコンマ区切りリストを指定する HTTP ヘッダーを追加で提供します。

このヘッダー名を検索するように WebSEAL を構成する必要があります (ステップ 4 を参照)。このヘッダーのデフォルト名は、**am-eai-xattrs** です。

(**am-eai-xattrs** ヘッダー名は構成可能です。)

3. **am_eai_xattr_session_lifetime** ヘッダー名が **am-eai-xattrs** ヘッダーに対する値として含まれるようにカスタム外部認証インターフェース・プログラムを構成します。例:

```
am-eai-xattrs: am_eai_xattr_session_lifetime
```

4. WebSEAL 構成ファイルの **[eai]** スタンザを使用して、外部認証インターフェース・サーバーから戻される認証データを含む HTTP ヘッダーの名前を指定します。

[eai] スタンザで、WebSEAL が **am-eai-xattrs** ヘッダー名を検索するように指定します。

```
[eai]
```

```
eai-xattrs-header = am-eai-xattrs
```

注: 外部認証インターフェースで使用するヘッダー名はカスタマイズできません。カスタム外部認証インターフェース認証モジュールが、構成されているとおりにヘッダー名を使用するように作成してください。

タスクの結果

外部認証インターフェースからのフラグ付き応答に **am_eai_xattr_inactive_timeout** ヘッダーが存在する場合、WebSEAL はその値を拡張属性としてユーザーの資格情報に追加します。この例の資格情報のエントリーは以下のようになります。

```
am_eai_xattr_session_lifetime:1129225478
```

資格情報が正常に作成されると、WebSEAL はそのクライアントのセッション・キャッシュにエントリーを作成し、拡張属性の値を使用して、そのクライアントのセッション・キャッシュ・エントリーの非アクティブ・タイムアウトを設定します。

am_eai_xattr_session_lifetime ヘッダーが指定されていない場合、WebSEAL は **timeout** スタンザ・エントリーが提供するデフォルトのタイムアウト値を使用します。

例:

例えば、Tivoli Federated Identity Manager 環境では、ID プロバイダーが、サービス・プロバイダーでのユーザーのセッション中にサービス・プロバイダーに指示するために使用する Liberty 認証応答のオプションの要素があります。

ユーザーの認証に使用する外部認証インターフェースを変更することによって、単一の属性 (アイデンティティー・プロバイダー・トークンから引き出された値) を WebSEAL に戻し、この属性を使用してそのユーザーのセッション・キャッシュ・エントリーの存続時間タイムアウトを設定できます。このキャッシュ・エントリー存続時間値が期限切れになると、サービス・プロバイダーは、ID プロバイダーとの新規シングル・サインオンによる対話を常に要求します。

以下のセクションも参照してください。

- 333 ページの『古いセッション Cookie に対するカスタマイズ応答』
- 296 ページの『キャッシュ・エントリーの存続時間タイムアウト値』
- 『クライアント固有のセッション・キャッシュ・エントリー非アクティブ・タイムアウト値の設定』

クライアント固有のセッション・キャッシュ・エントリー非アクティブ・タイムアウト値の設定

このタスクについて

WebSEAL 構成ファイルの [session] スタンザにある **inactive-timeout** スタンザ・エントリーは、WebSEAL セッション・キャッシュに保存されている非アクティブ・エントリーの最大存続時間をグローバルに設定します。このグローバル非アクティブ・タイムアウト値をクライアントごとの値でオーバーライドすることができます。クライアントごとの値は、外部認証インターフェース・サービスの認証応答のヘッダーとして提供されます。この値は WebSEAL によって抽出され、ユーザーの資格情報に拡張属性として保管されます。

WebSEAL はクライアント固有の非アクティブ・タイムアウト情報を、外部認証インターフェースの認証応答のヘッダーの値として受け取ります。WebSEAL はこのヘッダーの値を使用して、そのクライアントの新規セッション・キャッシュ・エントリーの非アクティブ・タイムアウトを設定します。この値は、**inactive-timeout** スタンザ・エントリーの値をオーバーライドします。

この値は、セッションが非アクティブな状態が続いた場合に、WebSEAL セッション・キャッシュから削除されるまでの最大の時間を秒数で表します。

以下のステップは、クライアント固有のキャッシュ・エントリー非アクティブ・タイムアウト値の設定に必要な構成を要約したものです。

手順

1. カスタム外部認証インターフェース・プログラムを構成して、クライアントに適したセッション・キャッシュ非アクティブ・タイムアウト値を含む HTTP ヘッダーを認証応答で提供するようにします。このヘッダーに必須の名前は次のとおりです。

```
am_eai_xattr_session_inactive_timeout
```

注: この特殊ヘッダーの名前は構成できません。以下に例を示します。

```
am_eai_xattr_session_inactive_timeout:120
```

2. カスタム外部認証インターフェース・プログラムを構成して、拡張属性値を含む HTTP ヘッダー名のコンマ区切りリストを指定する HTTP ヘッダーを追加で提供します。

このヘッダー名を検索するように WebSEAL を構成する必要があります (ステップ 4 を参照)。このヘッダーのデフォルト名は、**am-eai-xattrs** です。

(**am-eai-xattrs** ヘッダー名は構成可能です。)

3. **am_eai_xattr_session_inactive_timeout** ヘッダー名が **am-eai-xattrs** ヘッダーに対する値として含まれるようにカスタム外部認証インターフェース・プログラムを構成します。以下に例を示します。

```
am-eai-xattrs: am_eai_xattr_session_inactive_timeout
```

4. WebSEAL 構成ファイルの **[eai]** スタンザを使用して、外部認証インターフェース・サーバーから戻される認証データを含む HTTP ヘッダーの名前を指定します。

[eai] スタンザで、WebSEAL が **am-eai-xattrs** ヘッダー名を検索するように指定します。

```
[eai]
eai-xattrs-header = am-eai-xattrs
```

注: 外部認証インターフェースで使用するヘッダー名はカスタマイズできません。カスタム外部認証インターフェース認証モジュールが、構成されているとおりヘッダー名を使用するように作成してください。

タスクの結果

外部認証インターフェースからのフラグ付き応答に

am_eai_xattr_session_inactive_timeout ヘッダーが存在する場合、WebSEAL はその値を拡張属性としてユーザーの資格情報に追加します。この例の資格情報のエントリーは以下ようになります。

```
am_eai_xattr_session_inactive_timeout:120
```

資格情報が正常に作成できたら、WebSEAL はそのクライアントのセッション・キャッシュにエントリーを作成し、拡張属性の値を使用して、そのクライアントのセッション・キャッシュ・エントリーの非アクティブ・タイムアウトを設定します。

am_eai_xattr_session_inactive_timeout ヘッダーが提供されない場合、WebSEAL は **inactive-timeout** スタンザ・エントリーが提供するデフォルトのタイムアウト値を使用します。

以下のセクションも参照してください。

- 333 ページの『古いセッション Cookie に対するカスタマイズ応答』
- 296 ページの『キャッシュ・エントリーの存続時間タイムアウト値』
- 279 ページの『クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定』

第 4 部 セッション状態

第 14 章 セッション状態の概要

この章では、WebSEAL がセッション状態を維持する方法の基本概念について説明します。

この章で扱うトピックは以下のとおりです。

- 『セッション状態の概念』
- 『サポートされるセッション ID データ・タイプ』
- 286 ページの『クライアント要求から取得される情報』
- 287 ページの『WebSEAL セッション・キャッシュ構造』
- 288 ページの『クラスター環境におけるデプロイメントの考慮事項』
- 289 ページの『クラスター環境におけるフェイルオーバー処理のオプション』

セッション状態の概念

クライアント/サーバー・セッションは、一定期間に発生する単一のクライアントとサーバー間の関連した一連の対話を指します。確立されたセッションで、サーバーは各要求に関連するクライアントを識別し、多数の要求にわたって特定のクライアントを認識することができます。

確立済みのセッション状態がない場合は、以後の各要求ごとに、クライアントとサーバーの間の通信を再折衝する必要があります。セッション状態の情報は、以下のようにしてパフォーマンスを改善します。

- WebSEAL サーバーへのすべての要求に認証データが含まれる基本認証などのクライアント認証方式の場合、セッション状態情報によって、要求ごとにユーザー名およびパスワードを検査する必要がなくなります。
- ユーザーにログイン・プロンプトを出す必要がある他のクライアント認証方式の場合、セッション状態情報によって、WebSEAL サーバーへの要求ごとにユーザーにログインを求める必要がなくなります。クライアントは、各要求ごとに新たなログインを行うことなく、1 回ログインするだけで多数の要求を出すことができます。

サポートされるセッション ID データ・タイプ

WebSEAL は、HTTP と HTTPS クライアントの両方とのセッション状態を維持することができます。SSL トランスポート・プロトコルは特に、セッション ID を提供するように設計されているので、セッション状態情報を維持できます。対照的に、HTTP は「ステートレス」プロトコルであり、ある要求を別の要求と区別する手段は提供しません。(HTTP 通信は、SSL 上でカプセル化され、HTTPS となることができます。)

しかし、WebSEAL は、非認証クライアントからの HTTP 通信を取り扱わなければならないことがよくあります。また、SSL セッション ID が適切なソリューションとは言えない場合もあります。

HTTP または HTTPS を介してクライアントとのセッション状態を維持するために、WebSEAL は複数のデータ・タイプの中の 1 つを使用して、WebSEAL セッション ID と呼ばれるクライアント識別セッション鍵を提供します。

WebSEAL は、セッション・キャッシュに特定のクライアント識別およびセッション情報を維持します。各セッション・キャッシュ・エントリーは、セッション鍵 (WebSEAL セッション ID) によって索引付けされています。

以下のサポートされるデータ・タイプでは、WebSEAL がクライアントとのセッション状態の維持に使用するセッション鍵を提供できます。

- SSL セッション ID (SSL プロトコルにより定義される)
- サーバー固有セッション Cookie

WebSEAL は、クライアント要求を検査するときに、セッション鍵を上記のリストの順序で検索します。

クライアント要求から取得される情報

セッションの識別とは、HTTP 要求 (URL、HTTP ヘッダーや Cookie、IP アドレス、および SSL セッション ID など) に関連した情報を調査し、特定のクライアントと要求を関連付けるために使用するセッション ID を取得するプロセスです。

WebSEAL は、クライアント要求に、以下の情報が付随しているか調べます。

- **セッション鍵**

セッション鍵は、クライアントと WebSEAL サーバーの間の特定の接続を識別する情報です。セッション鍵はクライアントと共に保管され、そのクライアントによる以後の要求に付随します。このデータは、WebSEAL に対してクライアント・セッションを再識別するため、および個々の要求ごとに新規セッションを確立することによるオーバーヘッドを回避するために使用されます。セッション鍵は、WebSEAL サーバー・セッション・キャッシュに保管された関連セッション・データに対するロケータ索引です。セッション鍵は、WebSEAL セッション ID ともいいます。

- **認証データ**

認証データは、WebSEAL サーバーに対してクライアントを識別するために、クライアントから提供される情報です。認証データのタイプの例として、クライアント・サイド証明書、パスワード、トークン・コードなどがあります。

WebSEAL はクライアント要求を受け取ると、常に、最初にセッション鍵および関連するセッション・データを調べ、次に認証データを調べます。

WebSEAL セッション・キャッシュ構造

WebSEAL セッション・キャッシュは、認証済みユーザーが確立したすべてのセッションに関する情報を WebSEAL が格納する内部テーブルと考えることができます。クライアントで格納されるセッション鍵は、WebSEAL セッション・キャッシュに保管された関連するセッション・データに対するロケータ索引です。

WebSEAL セッション・キャッシュ

セッション鍵	キャッシュ・データ	タイム・スタンプ
1234	- ユーザー資格情報 - 内部フラグ - 内部データ	- 作成時刻 - 最終アクティブ時刻

図 19. WebSEAL セッション・キャッシュ

各ユーザー・セッションは、キャッシュ・テーブル内の 1 つのエントリーで表されます。

それぞれのキャッシュ・エントリーには、次のタイプの情報が入ります。

- **セッション鍵**

セッション鍵 (WebSEAL セッション ID) とは、そのユーザーが出す各要求で送信される固有の ID または鍵です。セッション鍵は、そのユーザー用の特定のキャッシュ・エントリーを識別します。

- **キャッシュ・データ**

キャッシュ・エントリーに格納される最も重要なデータは、**ユーザー資格情報**です。資格情報は、ユーザーが保護リソースを要求するときに必要になります。許可サービスは、資格情報の情報を使用して、リソースへのアクセスを許可または拒否します。

WebSEAL は、キャッシュ・エントリーに、特定の機能をサポートするためのマークを付ける (つまり「フラグ」を立てる) ことができます。例えば、セッション非活動の再認証を使用可能にした場合、セッション非活動値の期限が切れると、キャッシュ・エントリーに「フラグ」が立てられます。

- **タイム・スタンプ**

キャッシュ・エントリーのタイム・スタンプが作成された時点は、そのセッションの存続期間値の参照点になります。キャッシュ・エントリーの「最後のアクティブな」タイム・スタンプは、セッションの非アクティブ・タイマーの参照点になります。

ユーザー資格情報は、認証済みユーザーを表すエンコードされた不透明データ構造です。資格情報の内容には以下のものが含まれます。

- ユーザー名
- グループ・メンバーシップ
- 拡張属性

拡張属性を使用することで、カスタマイズ済みのデータをユーザー資格情報に格納できます。

資格情報およびその処理について詳しくは、「*IBM Security Access Manager for Web: Authorization C API Developer Reference*」を参照してください。

クラスター環境におけるデプロイメントの考慮事項

フォールト・トレランスまたはパフォーマンス上の理由でクラスター環境に複数のレプリカ WebSEAL サーバーをデプロイする場合、以下のトピックを考慮してください。

すべての WebSEAL レプリカ・サーバーでの整合性のとれた構成

クライアントがアクセスする WebSEAL サーバーに関係なく、一貫性のあるユーザー・エクスペリエンスを維持するには、すべての WebSEAL レプリカ・サーバーをまったく同じに構成する必要があります。

例えば、ある WebSEAL サーバーに `junction` が存在し、別のサーバーには存在しない場合、適切な `junction` 定義を保有していない WebSEAL サーバーにアクセスしたクライアントはエラーを受け取る可能性があります。クラスター上のすべての WebSEAL サーバーで、すべての構成（例えば、動的 URL、`junction` マッピング・テーブル、認証、許可など）がまったく同じでなければなりません。

WebSEAL 構成ファイルの `[server]` スタンザの `server-name` 構成オプションを使用して、すべての WebSEAL サーバーが同じ保護オブジェクト・スペース上で許可検査を実行するように強制することができます。この構成により、ACL および POP を一度だけ適用することができます。他の大半の WebSEAL 構成オプションは、クラスター内のサーバーごとに個別に設定する必要があります。

35 ページの『構成ファイル内の WebSEAL サーバー名』を参照してください。

ロード・バランサーでのクライアントとサーバー間のセッション・アフィニティー

可能な場合は、セッション・アフィニティーを維持するようにロード・バランサーを構成します。

セッション・アフィニティーによって、パフォーマンスおよびユーザー・エクスペリエンスが改善され、WebSEAL 構成を単純化できます。

新規マスターへのフェイルオーバー

クラスター・マスターが長期間使用できなくなった場合は、別のマスターを使用するようにスレーブを再構成します。新規マスターを構成するには、スレーブ

WebSEAL サーバーごとに、[cluster] スタンザの **master-name** 構成エントリに変更を加えます。新しく指定されたマスターは最新の構成になるようにしてください。

ある WebSEAL サーバーから別のサーバーへのフェイルオーバー

クライアントが、ある WebSEAL サーバーから別のサーバーにフェイルオーバーする場合、新規 WebSEAL サーバーがクライアントを識別するための何らかのメカニズムが存在しなければなりません。

セキュリティ設計者は、フェイルオーバー・イベントを処理するために使用可能な複数のオプションから選択する必要があります。オプションごとに、複雑性、セキュリティ、パフォーマンスにおける異なるトレードオフがあります。

また、フェイルオーバー・イベントを処理するためのオプションによって、追加機能が提供される場合もあります。例えば、シングル・サインオンや、ソフトウェア・サポート要員および WebSEAL アドミニストレーターが使用する柔軟性の高いセッション管理ツールなどがあります。

クラスター環境におけるフェイルオーバー処理のオプション

WebSEAL では、クラスター環境の複数サーバーでセッション状態をセキュアに共有するという課題に対して、いくつかのソリューションを提供します。以下のセクションで、クラスター環境でフェイルオーバー・イベントを処理するために使用可能なオプションについて説明します。

オプション 1: WebSEAL がフェイルオーバー・イベントを処理しない

WebSEAL クラスターの前にあるロード・バランサーが長期間セッション・アフィニティを維持できる場合、フェイルオーバー・イベントが発生することはまれです。

それでもフェイルオーバー・イベントが発生した場合は、クライアントは再度ログインする必要があります。

このオプションは比較的構成が容易ですが、WebSEAL サーバーが何らかの理由で使用不可になった場合、またはロード・バランサーがセッション・アフィニティを維持出来なくなった場合にユーザー・エクスペリエンスが悪化するリスクを含んでいます。

オプション 2: 認証データが各要求に含まれる

基本認証またはクライアント・サイド証明書などの一部の認証方式では、要求ごとに認証データを提供します。

クラスター内の WebSEAL サーバーがそのような認証方式を使用するように構成すると、フェイルオーバー・イベントが発生した場合、ユーザーに再度ログインを求めることなくユーザーは自動認証されます。

フェイルオーバー・イベントを処理するこの方式は、比較的構成が容易で、良好なユーザー・エクスペリエンスを構成および提供します。ただし、この方式では、再認証および **pkmslogout** などの特定の WebSEAL 機能を使用することはできません。

オプション 3: フェイルオーバー Cookie

フェイルオーバー Cookie は、ユーザーを透過的に再認証するためのメカニズムで、実際にはセッション維持を目的としたメカニズムではありません。フェイルオーバー Cookie には、WebSEAL サーバーがユーザー ID の検査に使用できる暗号化されたユーザー認証データが含まれています。フェイルオーバー Cookie には以下の情報が含まれています。

- ユーザー資格情報
- セッション非アクティブ・タイムアウト値
- セッション存続時間タイムアウト値

ただし、他のすべてのセッション状態データは、フェイルオーバー Cookie が収集または維持するわけではありません。

フェイルオーバー Cookie の構成では、クラスター内のすべての WebSEAL サーバーに共用秘密鍵が配布される必要があります。また説明した最初の 2 つのオプションよりも多くの構成が必要になります。

フェイルオーバー Cookie では、通常のセッション Cookie よりも大きなセキュリティ・リスクがあります。アタッカーがセッション Cookie を乗っ取っても、セッション Cookie は、WebSEAL サーバーが関連したセッションを削除するまでの間のみ有効です。一方、フェイルオーバー Cookie は、フェイルオーバー Cookie における存続時間または非アクティビティのタイムアウトに達するまで有効です。

フェイルオーバー Cookie では、セッション存続時間タイムアウト、非アクティビティ・タイムアウト、および **pkmslogout** を適用できます。また、フェイルオーバー Cookie は、同じ DNS ドメイン内の複数の WebSEAL クラスターにシングル・サインオンを提供することもできます。

フェイルオーバー Cookie メカニズムに関する詳細は、303 ページの『第 16 章 フェイルオーバー・ソリューション』を参照してください。

オプション 4: Session Management Server

Session Management Server (SMS) は、クラスター内のすべての WebSEAL サーバーがセッションを保管するために使用します。クライアントがフェイルオーバーした場合、新規 WebSEAL サーバーは SMS からユーザーのセッション・データを取得できるため、ユーザーに対して再度ログインを求める必要がありません。

フェイルオーバー Cookie と同様、SMS は、クラスター内のすべての WebSEAL サーバーで一貫した非アクティビティおよび存続時間のタイムアウトの追跡を可能にします。また、フェイルオーバー Cookie のように、同じ DNS ドメイン内の複数の WebSEAL クラスターでのシングル・サインオンを可能にします。

通常のセッション Cookie しか使用されないため、SMS はフェイルオーバー Cookie によって発生するセキュリティー・リスクを削減します。

また、SMS は、複数のサーバー・クラスターのセッション状態を維持する他の方式では使用できない追加機能を提供します。例えば、SMS では、カスタマー・サポート要員および WebSEAL アドミニストレーターが、ある時点でクラスターにログインしているすべてのユーザーを表示することができます。

また、SMS では、ユーザーごとに許可された同時セッション数を制限する **max-concurrent-web-sessions** ポリシーもサポートします。

SMS は、以下の目的で利用できるログイン・ヒストリー・データベースを提供します。

- WebSEAL サーバーにログインした最終時刻をユーザーに表示する。
- 最終ログイン以降、所有するユーザー ID で WebSEAL サーバーへの認証に失敗した回数をユーザーに表示する。

SMS は多数の利点を提供しますが、WebSEAL のデプロイメントに対して追加構成が必要で、いくつかのリスクもあります。特に、SMS が使用できない場合、WebSEAL サーバーはユーザーからの要求をまったく処理できません。デプロイメント上に Single Point of Failure が発生するのを避けるためには、SMS を複製する必要があります。

セッション管理サーバーに関する詳細は、363 ページの『第 20 章 SMS を使用した WebSEAL の構成』を参照してください。

オプション 5: LTPA Cookie

フェイルオーバー Cookie は実際にはセッション維持を目的としたメカニズムではなく、本来はユーザーを透過的に認証するためのメカニズムです。LTPA Cookie には、WebSEAL サーバーがユーザー ID の検査に使用できる暗号化されたユーザー認証データが含まれています。LTPA Cookie には以下の情報が含まれています。

- ユーザー名
- セッション存続時間タイムアウト値

ただし、他のすべてのセッション状態データは、LTPA Cookie が収集または維持するわけではありません。LTPA Cookie の構成では、クラスター内のすべてのサーバーに共用秘密鍵が配布される必要があります。また説明した最初の 2 つのオプションよりも多くの構成が必要になります。

LTPA Cookie では、通常のセッション Cookie よりも大きなセキュリティー・リスクがあります。アタッカーがセッション Cookie を乗っ取っても、セッション Cookie は、WebSEAL サーバーが関連したセッションを削除するまでの間のみ有効です。一方、LTPA Cookie は、LTPA Cookie における存続時間タイムアウトに達するまで有効です。

LTPA Cookie では、セッション存続時間タイムアウトおよび **pkmslogout** を適用できます。また、LTPA Cookie は、同じ DNS ドメイン内の複数の WebSEAL クラス

ターにシングル・サインオンを提供したり、同じ DNS ドメイン内の他の LTPA 対応サーバー (WebSphere Application Server、DataPower など) にシングル・サインオンを提供したりできます。

Cookie ベースのフェイルオーバー方式を使用している場合は、LTPA Cookie オプションを介して、オプション 3 で示したフェイルオーバー Cookie を使用する必要があります。LTPA Cookie の主な目的は、サード・パーティー・サーバー (WebSphere Application Server、DataPower など) に対するシングル・サインオンを使用可能にすることです。

LTPA Cookie メカニズムについて詳しくは、177 ページの『LTPA 認証』を参照してください。

第 15 章 セッション・キャッシュ構成

この章では、SSL セッション・キャッシュおよび WebSEAL セッション・キャッシュの構成について説明します。

この章で扱うトピックは以下のとおりです。

- 『セッション・キャッシュ構成の概要』
- 294 ページの『SSL セッション ID キャッシュ構成』
- 295 ページの『WebSEAL セッション・キャッシュの構成』

セッション・キャッシュ構成の概要

サーバーは、セッション・キャッシュを使用して、複数のクライアントからのセッション情報を保管します。WebSEAL は、クライアントと WebSEAL 間の HTTPS と HTTP の両方のセッション状態情報を格納する 2 つのタイプのセッション・キャッシュを使用します。

- **WebSEAL セッション・キャッシュ**

WebSEAL セッション・キャッシュは、認証済みユーザーおよび非認証ユーザーが確立したすべてのセッションに関する情報を格納します。クライアントで格納されるセッション鍵は、WebSEAL セッション・キャッシュに保管された関連するセッション・データに対するロケーター索引です。

WebSEAL セッション・キャッシュには、いくつかあるデータの中から特に、クライアントごとに取得された資格情報が格納されます。資格情報をキャッシュに保管するのは、許可検査時に繰り返しユーザー・レジストリー・データベースを照会しなくて済むようにするためです。

- **SSL セッション ID キャッシュ**

SSL セッション・キャッシュには、SSL セッション状態を維持するために使用される SSL セッション ID が格納されます。

SSL セッション ID は、WebSEAL セッション・キャッシュのセッション索引として使用できます。

WebSEAL セッション・キャッシュおよび SSL セッション ID キャッシュを構成するための構成ファイル・エントリーは、以下の図に要約されています。

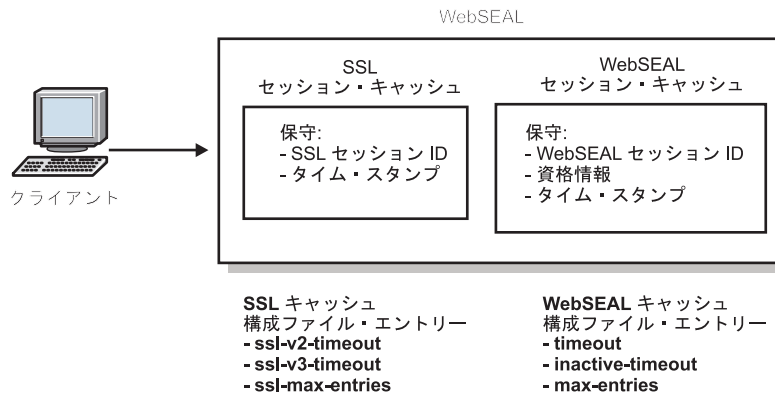


図 20. セッション・キャッシュ構成ファイル・エントリー

セッション状態の概念の概要については、285 ページの『第 14 章 セッション状態の概要』を参照してください。

SSL セッション ID キャッシュ構成

SSL セッション ID キャッシュについては、以下の構成タスクを行うことができます。

- 『キャッシュ・エントリーのタイムアウト値』
- 『最大同時 SSL セッション値』

キャッシュ・エントリーのタイムアウト値

SSL セッション ID キャッシュ内のエントリーの最大存続時間タイムアウトを設定するためのスタンザ・エントリーは、WebSEAL 構成ファイルの **[ssl]** スタンザに入っています。スタンザ・エントリーは 2 つあり、1 つは SSL v2 接続用 (**ssl-v2-timeout**) で、もう 1 つは SSL v3 接続用 (**ssl-v3-timeout**) です。SSL v3 セッション・タイムアウトは、TLS v1 接続にも使用されます。

デフォルトの SSL v2 セッション・タイムアウト値は、100 秒です (指定できる値の範囲は 1 から 100)。

```
[ssl]
ssl-v2-timeout = 100
```

デフォルトの SSL v3 セッション・タイムアウト値は、7200 秒です (指定できる値の範囲は 1 から 86400)。

```
[ssl]
ssl-v3-timeout = 7200
```

最大同時 SSL セッション値

WebSEAL 構成ファイルの **[ssl]** スタンザにある **ssl-max-entries** スタンザ・エントリーは、SSL セッション ID キャッシュ内の最大同時 SSL セッション数を設定します。

この値は、WebSEAL サーバーがその時々追跡する SSL セッション数を制限します。キャッシュ・サイズがこの値に達すると、最低使用頻度アルゴリズムに従って、キャッシュからエントリーが削除されます。SSL セッションが廃棄されたクライアントが再度 WebSEAL サーバーに接続する場合、WebSEAL は自動的に新規 SSL セッションをクライアントと折衝します。

SSL セッション ID が WebSEAL セッション・キャッシュのセッション索引として使用されている場合、再折衝のため、クライアントの WebSEAL セッション ID は変更されます。クライアントは WebSEAL に対して再認証する必要があります。

デフォルトの同時 SSL セッション数は、4096 です。

```
[ssl]
ssl-max-entries = 4096
```

パフォーマンスの考慮事項については、「*IBM Security Access Manager for Web: Performance Tuning Guide*」を参照してください。

WebSEAL セッション・キャッシュの構成

WebSEAL では 2 つの独立したセッション・キャッシュが維持されます。1 つは認証済みユーザー用で、もう 1 つは認証中のユーザー用です。ユーザーが認証されると、そのユーザーのセッション・キャッシュ・エントリーは非認証セッション・キャッシュから認証済みセッション・キャッシュに移動されます。

以下のセクションで、WebSEAL セッション・キャッシュの構成方法と使用法について説明します。

- 『最大セッション・キャッシュ・エントリー値』
- 296 ページの『キャッシュ・エントリーの存続時間タイムアウト値』
- 279 ページの『クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定』
- 299 ページの『キャッシュ・エントリーの非アクティブ・タイムアウト値』
- 300 ページの『並行セッションの制限』
- 301 ページの『セッション・キャッシュの制限』

最大セッション・キャッシュ・エントリー値

WebSEAL 構成ファイルの `[session]` スタンザにある `max-entries` スタンザ・エントリーは、WebSEAL 非認証セッション・キャッシュおよび認証済みセッション・キャッシュ内のセッション・キャッシュ・エントリーの最大数を指定します。

この値は、同時ログイン・セッションの数と同じです。キャッシュ・サイズがこの値に達すると、最低使用頻度アルゴリズムにしたがって、キャッシュからエントリーが削除され、新しい着信ログインを入れるためのスペースが作られます。

指定する値には以下の条件が反映されます。

- 指定値が 0 以下の場合、キャッシュ・サイズは無制限になります。
- 指定値が 0 から 8192 の間の場合、許容されるエントリーの実際数は次の 32 の倍数に切り上げられます。

- 指定した値が 8192 より大きい場合、その値はそのまま受け入れられます。

WebSEAL は、最大値を強要しません。整数値の最大サイズに関するガイドラインについては、703 ページの『付録 A. 構成ファイルを変更するためのガイドライン』を参照してください。

デフォルトの最大同時ログイン・セッション数は、4096 です。

```
[session]
max-entries = 4096
```

特定のセッション・キャッシュ（非認証または認証のいずれか）の値は、構成エントリーにセッション・キャッシュ名（unauth または auth のいずれか）の接頭部をつけることにより指定できます。以下に例を示します。

```
unauth-max-entries = 1024
```

パフォーマンスの考慮事項については、「*IBM Security Access Manager for Web: Performance Tuning Guide*」を参照してください。

キャッシュ・エントリーの存続時間タイムアウト値

WebSEAL 構成ファイルの [session] スタンザ内にある **timeout** スタンザ・エントリーは、WebSEAL 認証済みセッション・キャッシュまたは非認証セッション・キャッシュに保存されるすべてのユーザー・セッション情報の最大存続時間タイムアウト値を設定します。

WebSEAL は資格情報を内部的にキャッシュします。そこで、セッション・キャッシュの **timeout** スタンザ・エントリーで、許可資格情報が WebSEAL 上のメモリー内にとどまる時間の長さを指示します。

このスタンザ・エントリーは、非アクティブ・タイムアウトではありません。この値は、「セッション非アクティブ・タイムアウト」ではなく「資格情報存続時間」に対応します。この目的は、指定されたタイムアウト限界に到達したときにユーザーに再認証を強制することによって、セキュリティを強化することです。

デフォルトのセッション・キャッシュ・エントリー存続時間タイムアウト（秒数）は、3600 です。

```
[session]
timeout = 3600
```

特定のセッション・キャッシュ（非認証または認証のいずれか）の値は、構成エントリーにセッション・キャッシュ名（unauth または auth のいずれか）の接頭部をつけることにより指定できます。以下に例を示します。

```
unauth-max-entries = 1024
```

WebSEAL は、このスタンザ・エントリーの最大値を強要しません。

値が 0 の場合、このタイムアウト機能は使用不可になります（存続時間値は制限されません）。キャッシュ・エントリーの制御は、**inactive-timeout** および **max-entries** スタンザ・エントリーによって行われます。

キャッシュがフルになると、最低使用頻度アルゴリズムに基づいてエントリーがクリアされます。295 ページの『最大セッション・キャッシュ・エントリー値』を参照してください。

注: このスタンザ・エントリーは、基本認証 (BA)、SPNEGO、および特定の形式の証明書認証のように、WebSEAL サーバーへの各要求に認証データを含む認証方式では無効です。これらの認証方式は、ユーザー・セッションが非アクティブまたは存続時間タイムアウトによって削除された場合、WebSEAL サーバーに対してそのユーザーを自動的に再認証します。その結果、非アクティブおよび存続時間タイムアウトの値のリセットが繰り返されます。

も参照してください。

- 333 ページの『古いセッション Cookie に対するカスタマイズ応答』
- 279 ページの『クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定』

クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定

このタスクについて

WebSEAL 構成ファイルの `[session]` スタンザ内の `timeout` スタンザ・エントリーでは、WebSEAL セッション・キャッシュに保管されているすべてのクライアント・セッション情報の最大存続時間タイムアウト値をグローバルに設定します。このグローバル存続時間値をクライアントごとの存続時間値でオーバーライドすることができます。クライアントごとの存続時間値は、外部認証インターフェース・サービスの認証応答フォームのヘッダーとして提供されます。この値は WebSEAL によって抽出され、ユーザーの資格情報に拡張属性として保管されます。

WebSEAL は、クライアント固有のタイムアウト情報を外部認証インターフェースの認証応答のヘッダーの値として受信します。WebSEAL は、このヘッダーの値を使用して、そのクライアントの新規セッション・キャッシュ・エントリーの存続時間タイムアウトを設定します。この値は、`timeout` スタンザ・エントリーの値をオーバーライドします。

この値は、1970 年 1 月 1 日 00:00:00 UTC 以降の秒数で表された絶対時間として示す必要があります。例えば、UNIX `time ()` 関数の出力は、この絶対時間の値を正しいフォーマットで示します。

以下のステップは、クライアント固有のキャッシュ・エントリー存続時間タイムアウト値の設定に必要な構成を要約しています。

手順

1. カスタム外部認証インターフェース・プログラムを構成して、認証応答で、そのクライアントに適したセッション・キャッシュ存続時間タイムアウト値を含む HTTP ヘッダーを提供します。このヘッダーに必須の名前は次のとおりです。

```
am_eai_xattr_session_lifetime
```

注: この特殊ヘッダーの名前は構成できません。
以下に例を示します。

```
am_eai_xattr_session_lifetime:1129225478
```

2. カスタム外部認証インターフェース・プログラムを構成して、拡張属性値を含む HTTP ヘッダー名のコンマ区切りリストを指定する HTTP ヘッダーを追加で提供します。

このヘッダー名を検索するように WebSEAL を構成する必要があります (ステップ 4 を参照)。このヘッダーのデフォルト名は、**am-eai-xattrs** です。

(**am-eai-xattrs** ヘッダー名は構成可能です。)

3. **am_eai_xattr_session_lifetime** ヘッダー名が **am-eai-xattrs** ヘッダーに対する値として含まれるようにカスタム外部認証インターフェース・プログラムを構成します。例:

```
am-eai-xattrs: am_eai_xattr_session_lifetime
```

4. WebSEAL 構成ファイルの **[eai]** スタンザを使用して、外部認証インターフェース・サーバーから戻される認証データを含む HTTP ヘッダーの名前を指定します。

[eai] スタンザで、WebSEAL が **am-eai-xattrs** ヘッダー名を検索するように指定します。

```
[eai]
```

```
eai-xattrs-header = am-eai-xattrs
```

注: 外部認証インターフェースで使用するヘッダー名はカスタマイズできません。カスタム外部認証インターフェース認証モジュールが、構成されているとおりヘッダー名を使用するように作成してください。

タスクの結果

外部認証インターフェースからのフラグ付き応答に **am_eai_xattr_inactive_timeout** ヘッダーが存在する場合、WebSEAL はその値を拡張属性としてユーザーの資格情報に追加します。この例の資格情報のエントリーは以下のようになります。

```
am_eai_xattr_session_lifetime:1129225478
```

資格情報が正常に作成されると、WebSEAL はそのクライアントのセッション・キャッシュにエントリーを作成し、拡張属性の値を使用して、そのクライアントのセッション・キャッシュ・エントリーの非アクティブ・タイムアウトを設定します。

am_eai_xattr_session_lifetime ヘッダーが指定されていない場合、WebSEAL は **timeout** スタンザ・エントリーが提供するデフォルトのタイムアウト値を使用します。

例:

例えば、Tivoli Federated Identity Manager 環境では、ID プロバイダーが、サービス・プロバイダーでのユーザーのセッション中にサービス・プロバイダーに指示するために使用する Liberty 認証応答のオプションの要素があります。

ユーザーの認証に使用する外部認証インターフェースを変更することによって、単一の属性 (アイデンティティー・プロバイダー・トークンから引き出された値) を WebSEAL に戻し、この属性を使用してそのユーザーのセッション・キャッシュ・エントリーの存続時間タイムアウトを設定できます。このキャッシュ・エントリー

存続時間値が期限切れになると、サービス・プロバイダーは、ID プロバイダーとの新規シングル・サインオンによる対話を常に要求します。

以下のセクションも参照してください。

- 333 ページの『古いセッション Cookie に対するカスタマイズ応答』
- 296 ページの『キャッシュ・エントリーの存続時間タイムアウト値』
- 281 ページの『クライアント固有のセッション・キャッシュ・エントリー非アクティブ・タイムアウト値の設定』

キャッシュ・エントリーの非アクティブ・タイムアウト値

ユーザー・セッションの非アクティブ状態のタイムアウト値を設定するには、WebSEAL 構成ファイルの `[session]` スタンザ内の `inactive-timeout` スタンザ・エントリーを使用します。

例えば、ユーザーが非アクティブ・タイムアウトより長い間非アクティブな状態にあると、WebSEAL はユーザー・セッションを完全に削除するか、または再認証が必要なセッションとしてフラグを立てます。非アクティブ・セッションに対する再認証の必要性については、278 ページの『外部認証インターフェースによる再認証』を参照してください。

デフォルトのログイン・セッション非アクティブ・タイムアウト (秒数) は、600 です。

```
[session]
inactive-timeout = 600
```

特定のセッション・キャッシュ (非認証または認証のいずれか) の値は、構成エントリーにセッション・キャッシュ名 (`unauth` または `auth` のいずれか) の接頭部をつけることにより指定できます。以下に例を示します。

```
unauth-inactive-timeout = 300
```

WebSEAL は、このスタンザ・エントリーの最大値を強要しません。

値が 0 の場合、この非アクティブ・タイムアウト機能は使用不可になります (非アクティブ・タイムアウト値は制限されません)。キャッシュ・エントリーの制御は、`timeout` および `max-entries` スタンザ・エントリーによってのみ行われます。

キャッシュがフルになると、最低使用頻度アルゴリズムに基づいてエントリーがクリアされます。295 ページの『最大セッション・キャッシュ・エントリー値』を参照してください。

注: このスタンザ・エントリーは、基本認証 (BA)、SPNEGO、および特定の形式の証明書認証のように、WebSEAL サーバーへの各要求に認証データを含む認証方式では無効です。これらの認証方式は、ユーザー・セッションが非アクティブまたは存続時間タイムアウトによって削除された場合、WebSEAL サーバーに対してそのユーザーを自動的に再認証します。その結果、非アクティブおよび存続時間タイムアウトの値のリセットが繰り返されます。

非アクティブ・タイムアウトの保持

場合によっては、特定のリソースに対する要求によってセッションの非アクティブ・タイムアウトが影響を受けないようにする方がよいことがあります。例えば、クライアント・ブラウザのバックグラウンドで実行中の Ajax スクリプトによってサーバーがポーリングされる場合、非アクティブ・タイムアウトを保持することができます。

ユーザー・セッションの非アクティブ・タイムアウトに影響すべきでないリソースを指定するために、セキュリティ・ポリシーを作成できます。このセキュリティ・ポリシーを定義するには、**preserve-inactivity-time** という名前の拡張属性を指定した保護オブジェクト・ポリシー (POP) を作成する必要があります。この POP は、非アクティブ・タイムアウトが要求に影響されないようにする必要がある任意のオブジェクトに付加することができます。この POP が付随しているオブジェクトのすべての子も、POP 条件を継承するという点に注意してください。

preserve-inactivity-time POP を作成および適用するには、以下のコマンドを使用します。

- **pdadmin pop create**
- **pdadmin pop modify**
- **pdadmin pop attach**

以下の例では、**preserve-inactivity-time** 拡張属性を持つ **robot** という名前の POP を作成し、それを **status.html** オブジェクトに付加します。

```
pdadmin> pop create robot
pdadmin> pop modify robot set attribute preserve-inactivity-time true
pdadmin> pop attach /WebSEAL/hostA/junction/status.html robot
```

このポリシーが適用されているときは、**status.html** に対して行われた要求がユーザー・セッションの非アクティブ・タイムアウトに影響を与えることはありません。

pdadmin pop コマンドについて詳しくは、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

333 ページの『古いセッション Cookie に対するカスタマイズ応答』も参照してください。

370 ページの『SMS の最終アクセス時間更新頻度の調整』も参照してください。

279 ページの『クライアント固有のセッション・キャッシュ・エントリーの存続時間値の設定』も参照してください。

並行セッションの制限

WebSEAL は、単一のユーザー・セッションでの並行要求数を制限するように構成できます。

並行セッション・スレッドのハード制限

ハード制限は、単一ユーザー・セッションで消費可能な最大並行スレッド数です。ユーザー・セッションがそのスレッド制限に達すると、WebSEAL はそのユーザー・セッションでのすべての新規要求の処理を停止し、クライアントにエラーを返します。

[server] スタンザの **concurrent-session-threads-hard-limit** 構成エントリーを使用して、セッションのハード制限を構成します。

このエントリーの値を指定しない場合は、ユーザー・セッションで消費できる並行スレッドの数に制限がなくなります。

例えば次の構成では、単一ユーザー・セッションでの最大並行スレッド数が 10 になります。

```
[server]
concurrent-session-threads-hard-limit = 10
```

並行セッション・スレッドのソフト制限

[server] スタンザの **concurrent-session-threads-soft-limit** 構成エントリーを使用して、セッションのソフト制限を構成できます。

ソフト制限は、WebSEAL が警告メッセージを生成するまで、単一ユーザー・セッションで消費可能な最大並行スレッド数です。WebSEAL は、**concurrent-session-threads-hard-limit** の構成値に達するまで、このセッションでの要求を処理し続けます。

例えば以下のエントリーが設定されている場合、WebSEAL は、ユーザー・セッションのスレッド数がソフト制限値の 5 を超えると警告を生成します。WebSEAL は、ハード制限の 10 に達するまで要求の処理を続けます。それ以上の要求があると、WebSEAL はそのクライアントにエラーを返します。

```
[server]
concurrent-session-threads-soft-limit = 5
concurrent-session-threads-hard-limit = 10
```

セッション・キャッシュの制限

制限:

レジストリーからユーザーを削除しても、WebSEAL セッション・キャッシュに入っているそのユーザーの資格情報は除去されません。アカウントが削除された時点で、そのユーザーがブラウザー・セッションをアクティブにしていると、既存のセッション・キャッシュ・エントリーを基にして、ブラウズが続けられます。

ユーザー・レジストリー内の現行情報に基づくユーザーの資格情報の再評価は、新しいログインが行われるか、セッション・キャッシュ・エントリーの有効期限が切れるまで、行われることはありません。WebSEAL セッション・キャッシュの内容は、ユーザーがブラウザー・セッションからログアウトしたときにクリアされます。

対応策:

アドミニストレーターは、削除されたユーザーのデフォルトの WebSEAL ACL ポリシーに、全探索 (T) 許可を外した明示的なエントリーを追加することによって、ドメイン内のユーザー・アクティビティの即時停止を強制することができます。また、コマンド行あるいは Security Access Manager 管理 API 機能を使用して、手動でセッションを終了させることもできます。679 ページの『ユーザー・セッションの終了』を参照してください。

第 16 章 フェイルオーバー・ソリューション

この章では、クラスター化環境でセッション状態を維持するフェイルオーバー Cookie ソリューションについて説明します。

この章で扱うトピックは以下のとおりです。

- 『フェイルオーバー認証の概念』
- 311 ページの『フェイルオーバー認証構成』
- 321 ページの『非スティッキー・フェイルオーバー環境のフェイルオーバー』
- 324 ページの『フェイルオーバー環境におけるパスワード変更操作』

フェイルオーバー Cookie および Session Management Server (SMS) は、クラスター化されたサーバー環境でセッション状態を維持するという同じ目的を果たすための代替ソリューションです。363 ページの『第 20 章 SMS を使用した WebSEAL の構成』も参照してください。

フェイルオーバー認証の概念

WebSEAL では、複製サーバー環境 (フォールト・トレラント環境) で WebSEAL サーバーが使用不可になったときに、クライアントと WebSEAL との間で認証セッションを保持する認証方式を提供します。このメソッドはフェイルオーバー認証 と呼ばれます。

このセクションは、以下のトピックで構成されています。

- 『フェイルオーバー環境』
- 305 ページの『フェイルオーバー Cookie』
- 305 ページの『フェイルオーバー認証のプロセス・フロー』
- 306 ページの『フェイルオーバー構成の例』
- 306 ページの『フェイルオーバー Cookie へのデータの追加』
- 309 ページの『フェイルオーバー Cookie からのデータの抽出』
- 310 ページの『ドメイン全体にわたるフェイルオーバー認証』

311 ページの『フェイルオーバー認証構成』も参照してください。

フェイルオーバー環境

フェイルオーバー Cookie は、実際にはセッション維持を目的としたメカニズムではなく、ユーザーを透過的に再認証するためのメカニズムです。フェイルオーバー認証は、クライアント要求がロード・バランシング・メカニズムによって複数の複製された WebSEAL サーバーに送信されるようなシナリオで最もよく使用されます。

複製されたサーバーの構成は同一です。複製されたサーバーには、WebSEAL 保護オブジェクト・スペース、junction データベース、および dynurl データベース (オプション) のレプリカ・コピーが入っています。

クライアントは、複製されたフロントエンド・サーバーの構成を認識しません。ロード・バランシング・メカニズムは、要求されたリソースの単一 POC です。ロード・バランサーは、使用可能なサーバーにクライアントを接続します。

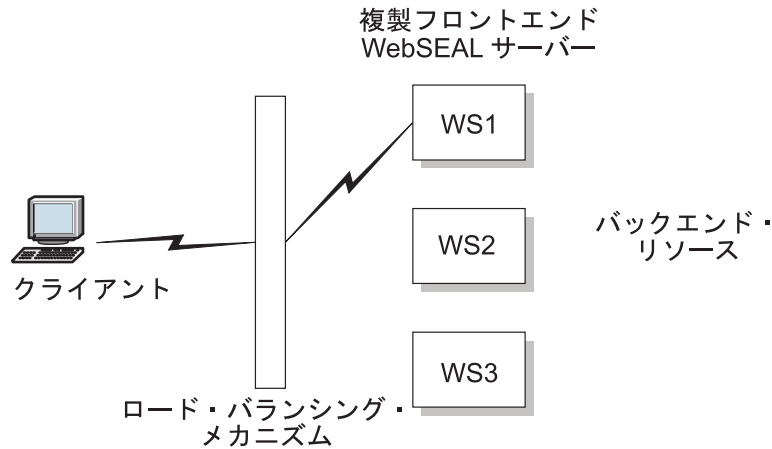


図 21. 複製 WebSEAL サーバーのフェイルオーバー

クライアントが接続されているサーバーが突然使用不可になった場合、ロード・バランサーは要求を他の複製サーバーの 1 つにリダイレクトします。このアクションは、元のセッション対資格情報のマッピングが失われる原因となります。クライアントは、この代替サーバーにとって新規であるため、通常は再度ログインする必要があります。

フェイルオーバー認証の目的は、元のクライアントとのセッションを保持している WebSEAL サーバーが突然使用不可になったときに、強制ログインを回避することです。フェイルオーバー認証により、クライアントは、別の WebSEAL サーバーに接続し、同じユーザー・セッション・データと同じユーザー資格情報を持った認証セッションを作成できます。

複製サーバー・デプロイメントにおけるフェイルオーバー認証は、2 つの便利な機能を提供します。

- ロード・バランシングによるパフォーマンスの向上
- WebSEAL サーバー間のクライアント・セッションのフェイルオーバー

参照:

- WebSEAL サーバーの複製について詳しくは、668 ページの『フロントエンド WebSEAL サーバーの複製』を参照してください。
- セッション・アフィニティー (非スティッキー) のない環境におけるフェイルオーバー・ソリューションの情報については、321 ページの『非スティッキー・フェイルオーバー環境のフェイルオーバー』を参照してください。

フェイルオーバー Cookie

WebSEAL は、フェイルオーバー *Cookie* を使用してユーザーのフェイルオーバー認証をサポートします。フェイルオーバー *Cookie* は、サーバー固有の *Cookie* すなわちドメイン *Cookie* にできます。フェイルオーバー *Cookie* には、以下の暗号化されたクライアント固有のデータが入っています。

- ユーザー名
- *Cookie* 作成のタイム・スタンプ
- オリジナルの認証方式
- 属性リスト

デフォルトでは、属性リストにはユーザーの現在の認証レベルが入ります。WebSEAL は、属性リストにさらに拡張属性を追加できるように構成できます。クライアントのセッション ID を拡張属性として保管するフェイルオーバー・ソリューションについては、321 ページの『非スティッキー・フェイルオーバー環境のフェイルオーバー』を参照してください。

Cookie は、クライアントが最初に接続したブラウザーに配置されます。最初の WebSEAL サーバーが一時的に使用不可になった場合、*Cookie* は代替サーバーに存在します。

複製された WebSEAL サーバーは、*Cookie* 情報を暗号化解除できる共通の鍵を共有します。代替の複製 WebSEAL サーバーは、この *Cookie* を受け取ると、*Cookie* を暗号化解除し、ユーザー名および認証方式を使用してクライアントの資格情報を再生成します。また、WebSEAL は、どの拡張属性でも *Cookie* からユーザー資格情報にコピーするように構成できます。

したがって、クライアントは、再ログインをプロンプトされることなく、レプリカ WebSEAL サーバーと新しいセッションを確立できます。

フェイルオーバー *Cookie* は、セッション状態の維持を目的としたメカニズムではありません。フェイルオーバー *Cookie* は、透過的にユーザーを再認証するためのメカニズムです。

注: フェイルオーバー *Cookie* は、HTTP または HTTPS のどちらかで使用できます。

フェイルオーバー認証のプロセス・フロー

次のステップで、フェイルオーバー認証のイベントの順序について説明します。

1. クライアント (ブラウザー) が保護リソースにアクセスしようとしています。クライアントの要求は、複製された WebSEAL サーバーへのアクセスをコントロールするロード・バランサーに行きます。
2. ロード・バランサーは、ターゲットの WebSEAL サーバーを選択し、ユーザー要求を転送します。
3. クライアントは、サポートされている認証方式のいずれか 1 つを使用して、WebSEAL から正常に認証されます。

4. WebSEAL は、クライアントの認証情報が入っているフェイルオーバー認証 Cookie を作成し、それをクライアント・ブラウザに送ります。
5. ブラウザーは、後続の要求のたびに、この Cookie を、ロード・バランサーを経由して WebSEAL に送ります。WebSEAL サーバーは、それぞれの要求を処理します。
6. オリジナルの WebSEAL サーバーがアクセスできないとロード・バランサーが判断した場合、クライアントの要求は、別の複製 WebSEAL サーバーにダイレクトされます。
7. 複製された WebSEAL サーバーは、ユーザーを認証しようとするときに必ずフェイルオーバー認証 Cookie の有無を検査するように構成されています。
8. 複製された WebSEAL サーバーは Cookie 中の情報を使用してクライアントとのセッションを確立します。このとき、クライアントに手動で再びログインするように求めることはありません。クライアントのセッション・データとユーザー資格情報がビルドされ、保護リソースに対する要求が処理されます。
9. ある WebSEAL サーバーから別の WebSEAL サーバーへのセッションの変更は、クライアントに透過的に行われます。WebSEAL サーバーには同一のリソースが組み込まれているため、クライアント・セッションは中断せずに継続されます。

フェイルオーバー構成の例

この例では、WebSEAL サーバーを、フォーム認証およびフェイルオーバー認証をサポートするように構成できます。

1. ユーザーはフォーム認証を使用して WebSEAL の認証を受けます。
2. WebSEAL サーバーは、フェイルオーバー認証 Cookie をそれぞれのクライアント (ブラウザ) に送ります。

Cookie データには、Cookie がフォーム認証環境で作成されたことが明記されています。

3. WebSEAL サーバーが使用不可になると、フェイルオーバー Cookie は 2 番目の WebSEAL サーバーに送られます。
4. 2 番目の WebSEAL サーバーはフェイルオーバー Cookie を受け取り、その Cookie を調べてユーザーの前の認証方式を判別します。
5. 2 番目の WebSEAL サーバーは、Cookie のデータを使用してユーザーを認証し、ユーザー資格情報を作成します。

構成手順については、以下の章を参照してください。

- 315 ページの『認証強度レベルの追加』

フェイルオーバー Cookie へのデータの追加

WebSEAL は、ユーザー・セッションからの特定のデータを、それぞれのフェイルオーバー認証 Cookie に自動的に追加します。また、WebSEAL を、資格情報キャッシュに維持されているクライアント・データにある追加情報を追加するよう構成できます。

デフォルトで、WebSEAL は、以下のデータをそれぞれの Cookie に追加します。

- **ユーザー名**

この名前は、ユーザー・レジストリーでユーザーを識別するのに使用した名前に対応します。

注: 認証済みユーザーが WebSEAL ユーザー切り替え機能を使用して別のユーザーの実効 ID を入手したときは、別のユーザーの ID は Cookie に追加されません。オリジナルの認証済みユーザーの ID だけが Cookie に追加されます。

- **認証方式**

WebSEAL がユーザーを認証するのに使用した認証方式。

- **Cookie 作成時刻**

Cookie が作成されたシステム時刻。

また、WebSEAL は、追加データが入った属性リストを作成します。デフォルトでは、属性リストには次の値が 1 つ入ります。

- **認証強度レベル**

ローカル WebSEAL サーバーで現在の認証方式に割り当てられた WebSEAL 認証強度レベル (これも整数値) に対応する整数値。認証強度 (ステップアップ認証ともいう) により、ユーザーは、ログアウトせずに、異なる認証方式で認証を受けることができます。

WebSEAL では、Cookie 属性リストに追加できる、次のような追加のユーザー・データが定義されます。

- **セッション存続時間タイム・スタンプ**

ユーザーが認証されると、WebSEAL は、WebSEAL セッション・キャッシュに入っているユーザー・エントリーの経過時間つまり存続時間をトラッキングします。セッション存続時間タイム・スタンプは現在時刻から成り、この時刻は、ユーザーのセッション・データがセッション・キャッシュに入っていられる最大時間までの秒数が加算されていきます。現行システム時刻がタイム・スタンプ値を超えると、WebSEAL は、セッション・キャッシュに入っているユーザーのエントリー (ユーザー資格情報を含む) を無効にします。

WebSEAL は、セッション存続時間タイム・スタンプを Cookie に追加するよう構成できます。このタイム・スタンプが Cookie に追加されると、セッション存続時間タイマーは、複数のフェイルオーバー・イベントにまたがって保存できます。WebSEAL アドミニストレーターは、クライアント・セッションが複製サーバーで確立されたときに、クライアントのセッション・タイマーをリセットするかしないかを選択することができます。

この機能の使用の成否は、複製 WebSEAL サーバー間でクロックが同期しているか否かに依存することに注意してください。クロック・スキューが大きくなると、セッションが予期せず終了する可能性があります。

- **セッション・アクティビティー・タイム・スタンプ**

セッション・アクティビティー・タイム・スタンプは、初期要求へ応答するサーバーでの作成時に、フェイルオーバー Cookie の属性として置かれる時間値です。

このタイム・スタンプは、WebSEAL セッション・キャッシュ用に維持されているセッション非アクティブ・タイムアウトとは異なります。フェイルオーバー Cookie のシステム・アクティビティー・タイム・スタンプは、現行システム時刻と最大時間を組み合わせることによって計算されます。タイム・スタンプは時間間隔で指定される頻度で更新されます。

Current®システム時刻

WebSEAL サーバーの現在時刻 (HH:MM 形式)。

値の例: 13:30

最大時間

ユーザーのセッションが非アクティブでいられる秒数 ([session], inactive-timeout)。

値の例: 600

時間間隔

フェイルオーバー認証 Cookie の更新と更新の間の秒数 ([failover], failover-update-cookie)。

値の例: 300

上記の例のフェイルオーバー Cookie のタイム・スタンプ値は、13:40 です。このセッション中の今後の要求がサーバー 1 からサーバー 2 にフェイルオーバーされると、サーバー 2 の時刻が 13:40 より前の場合のみ、サーバー 2 は要求を受け入れます。サーバー 2 の時刻が 13:40 以降である場合、サーバー 2 は要求を拒否し、サーバー 2 にログインするようにユーザーにプロンプトを出します。

フェイルオーバー Cookie のインターバルの設定は、パフォーマンスに影響を与えます。アドミニストレーターは、最適なパフォーマンスと、Cookie のタイム・スタンプの絶対正確性との間のバランスをとらなければなりません。タイム・スタンプを最も正確に保つには、ユーザーが要求を行うたびにフェイルオーバー Cookie を更新する必要があります。しかし、Cookie の内容の頻繁な更新は、必然的にオーバーヘッドを増やし、パフォーマンスを下げます。

それぞれのアドミニストレーターは、WebSEAL のデプロイメントに最適なインターバルを選択しなければなりません。ユーザー要求ごとに、フェイルオーバー Cookie を更新することが適切な場合があります。それ以外の場合は、アドミニストレーターが、フェイルオーバー Cookie のタイム・スタンプを更新しないことを選択することもできます。

• 追加拡張属性

アドミニストレーターは、カスタマイズされた属性のセットを、フェイルオーバー Cookie に挿入するよう WebSEAL を構成できます。属性は個々に指定することも、グループにまとめて指定することもできます。属性のグループを指定するには、構成ファイル・エントリーでワイルドカード・パターン・マッチングを使用します。

この機能は、特殊な属性をユーザー資格情報に挿入するために、カスタマイズされた認証モジュールも使用しているデプロイメントで使用する则便利です。これらの属性を WebSEAL 構成ファイルに指定することにより、アドミニストレータは、フェイルオーバー認証中に、これらの属性を再作成されたユーザー資格情報に必ず追加できるようになります。

注: フェイルオーバー認証 Cookie の最大サイズは、4 キロバイト (4096 バイト) です。

構成手順については、以下の章を参照してください。

- 315 ページの『認証強度レベルの追加』
- 316 ページの『セッション存続時間タイム・スタンプの追加』
- 316 ページの『セッション・アクティビティ・タイム・スタンプの追加』
- 317 ページの『アクティビティ・タイム・スタンプの更新間隔の追加』
- 318 ページの『追加の拡張属性』

フェイルオーバー Cookie からのデータの抽出

フェイルオーバー認証イベントが起これると、レプリカ WebSEAL サーバーはフェイルオーバー認証 Cookie を受け取り、デフォルトで、それぞれの Cookie から以下のデータを抽出します。

- ユーザー名
- 認証方式
- Cookie 作成時刻

WebSEAL は、まず、Cookie 作成時刻をシステム時刻から減算し、さらに、この値を、WebSEAL 構成ファイル・エントリーのフェイルオーバー Cookie 存続時間と比較して、この Cookie が有効かどうかを判別します。

Cookie の存続時間が超えている場合は、Cookie は有効ではなく、フェイルオーバー認証は試行されません。Cookie の存続時間が超えていない場合は、WebSEAL はユーザー名と認証方式を使用してユーザーを認証し、ユーザー資格情報を作成します。

次に、WebSEAL は構成設定値を検査して、追加の Cookie データを抽出して評価する必要があるかどうかを判別します。WebSEAL サーバーは、デフォルトでは、フェイルオーバー認証 Cookie から、この他の属性を抽出しないことに注意してください。抽出するその他の属性はそれぞれ、WebSEAL 構成ファイルに指定する必要があります。属性のグループを得るには、ワイルドカード・パターン・マッチングを使用できます。

WebSEAL は、以下の定義済み属性を抽出するように構成できます。

- **認証強度レベル**

この値が抽出されると、WebSEAL はこの値を使用して、指定された認証レベルを維持するのに必要な認証方式でユーザーの認証が行われるようにします。

WebSEAL は、認証強度レベルを、以下のさまざまな場所から入手できることに注意してください。

- フェイルオーバー Cookie
- フェイルオーバー認証ライブラリー
- クロスドメイン認証サービス
- 資格サービス

フェイルオーバー Cookie から抽出された認証強度レベルは、他の場所から得られたレベルより順位が優先します。

• セッション存続時間タイム・スタンプ

WebSEAL は、このタイム・スタンプを使用して、オリジナル・サーバーのセッション・キャッシュ内のユーザーのエントリーの有効期限が切れたかどうかを判別できます。有効期限が切れている場合、WebSEAL は、Cookie およびそのすべての潜在的な資格情報属性を破棄します。セッション存続時間は保存されず、ログインすることを求めるプロンプトがユーザーに出されます。

• セッション非アクティブ・タイム・スタンプ

WebSEAL は、このタイム・スタンプを使用して、オリジナル・サーバーのセッション・キャッシュ内のユーザーのエントリーが、長い時間非アクティブであったかどうかを判別できます。有効期限が切れている場合、WebSEAL は、Cookie およびそのすべての潜在的な資格情報属性を破棄します。セッション存続時間は保存されず、ログインすることを求めるプロンプトがユーザーに出されます。

注: これらのタイム・スタンプを正しく使用するには、さまざまな複製 WebSEAL サーバーのクロックが同期していることが必要です。クロックのスキューが大きくなると、セッションが突然終了するか非アクティブになります。

• 追加拡張属性

追加の拡張属性には、クロスドメイン認証サービスによって生成された属性などの、カスタマイズされた、ユーザー定義属性が含まれます。WebSEAL は、これらの属性をユーザー資格情報に追加します。

WebSEAL 構成ファイルに指定されていない属性は、無視され、抽出されません。さらに、アドミニストレーターは、フェイルオーバー Cookie の抽出中に、一定の属性は無視しなければならないと指定できます。無視はデフォルトの振る舞いですが、この指定は、例えば、ユーザー属性はフェイルオーバー Cookie からではなく、必ずユーザー・レジストリーから取得するようにする場合に役立ちます。

ドメイン全体にわたるフェイルオーバー認証

WebSEAL は、フェイルオーバー認証中に DNS ドメイン内のその他のすべての WebSEAL サーバーでできるように、フェイルオーバー認証 Cookie を使用可能にするオプションの構成をサポートします。この構成オプションにより、必ずしもロード・バランサーや複製 WebSEAL サーバーを持っていないデプロイメントで、フェイルオーバー認証 Cookie が使用できるようになります。

クライアント・セッションが、フェイルオーバー認証イベントを介して複製 WebSEAL サーバーに達するときは、クライアントは、同じセットの保護リソース

にアクセスし続けます。クライアント・セッションが、フェイルオーバー認証イベントを介して複製されていない WebSEAL サーバーに達するときは、別のセットのリソースがクライアントに使用可能になることが起こります。大規模なデプロイメントでは、DNS ドメイン内でのリソースのこのような区分化はよくあることです。この区分化は、パフォーマンス上の理由および管理上の目的で行うことができます。

ドメイン全体にわたるフェイルオーバー認証は、クライアントの要求が、ローカル WebSEAL サーバーでは使用できないリソースを要求することが必要になったときに、クライアントを別の WebSEAL サーバーにリダイレクトするために使用できます。この場合、クライアント (ブラウザー) は、別の WebSEAL サーバーにリダイレクトされます。受け取る側の WebSEAL サーバーは、フェイルオーバー認証 Cookie を探すように構成できます。WebSEAL サーバーはクライアントを認証しようとし、フェイルオーバー認証 Cookie を認識します。Cookie を使用することにより、WebSEAL サーバーは、ログイン情報を求めるプロンプトをクライアントに出す必要はなく、その代わりに、クライアントとのセッションを確立し、有効なユーザー資格情報のセットを作成することができます。

注: ドメイン規模のフェイルオーバー認証を使用可能にすると、WebSEAL デプロイメントに新たなセキュリティ・リスクが加わります。なぜなら、フェイルオーバー Cookie は WebSEAL サーバーと同じ DNS ドメインにある任意のサーバーに送信できるからです。アタッカーがドメイン内の任意の Web サーバーを制御したり、そのドメインの DNS サーバーを危険にさらしたりすることができれば、彼らはフェイルオーバー Cookie をハイジャックし、ユーザー名を使用することができます。

構成手順については、以下の章を参照してください。

- 319 ページの『ドメイン全体のフェイルオーバー Cookie の使用可能化』

フェイルオーバー認証構成

このセクションは、以下のトピックで構成されています。

- 312 ページの『フェイルオーバー認証の構成』
- 313 ページの『フェイルオーバー Cookie のプロトコル』
- 313 ページの『Cookie データを暗号化および暗号化解除する鍵ペアの生成』
- 314 ページの『フェイルオーバー Cookie 存続時間の指定』
- 314 ページの『Cookie スtringで使用する UTF-8 エンコードの指定』
- 315 ページの『認証強度レベルの追加』
- 315 ページの『欠落したフェイルオーバー Cookie の再発行』
- 316 ページの『セッション存続時間タイム・スタンプの追加』
- 316 ページの『セッション・アクティビティ・タイム・スタンプの追加』
- 317 ページの『アクティビティ・タイム・スタンプの更新間隔の追加』
- 318 ページの『追加の拡張属性』
- 318 ページの『抽出のための属性』
- 319 ページの『ドメイン全体のフェイルオーバー Cookie の使用可能化』

- 320 ページの『存続時間タイム・スタンプの妥当性検査』
 - 320 ページの『アクティビティ・タイム・スタンプの妥当性検査』
- 303 ページの『フェイルオーバー認証の概念』も参照してください。

フェイルオーバー認証の構成

フェイルオーバー認証に対応するように WebSEAL を構成することができます。

このタスクについて

フェイルオーバー認証を構成するには、以下のタスクを実行してください。

注:

これらのタスクに関連した構成エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

手順

1. WebSEAL サーバーを停止します。
2. フェイルオーバー認証を使用可能にするには、以下のタスクのそれぞれを実行してください。
 - a. 313 ページの『フェイルオーバー Cookie のプロトコル』
 - b. 313 ページの『Cookie データを暗号化および暗号化解除する鍵ペアの生成』
 - c. 314 ページの『フェイルオーバー Cookie 存続時間の指定』
 - d. 314 ページの『Cookie ストリングで使用する UTF-8 エンコードの指定』
 - e. 315 ページの『認証強度レベルの追加』
 - f. 315 ページの『欠落したフェイルオーバー Cookie の再発行』
3. 必要に応じて、複数のフェイルオーバー認証セッションにわたって、セッション状態を維持するように WebSEAL を構成できます。この構成がご使用のデプロイメントに適している場合は、以下の手順を実行してください。
 - a. 316 ページの『セッション存続時間タイム・スタンプの追加』
 - b. 316 ページの『セッション・アクティビティ・タイム・スタンプの追加』
 - c. 317 ページの『アクティビティ・タイム・スタンプの更新間隔の追加』
4. 必要に応じて、拡張属性をフェイルオーバー Cookie に追加するように、以下の手順を使用して WebSEAL を構成できます。
 - 318 ページの『追加の拡張属性』
5. フェイルオーバー Cookie に属性を追加するように WebSEAL を構成した場合は、Cookie を読み取るときに属性を抽出するように、以下の手順で WebSEAL を構成しなければなりません。
 - 318 ページの『抽出のための属性』
6. 必要に応じて、ドメイン内のどの WebSEAL サーバーでも使用できるように、フェイルオーバー認証 Cookie を使用可能にすることができます。この構成がご使用のデプロイメントに適している場合は、以下を参照してください。
 - 319 ページの『ドメイン全体のフェイルオーバー Cookie の使用可能化』

7. バージョン 6.0 より前のバージョンの WebSEAL サーバーで生成されたフェイルオーバー認証 Cookie と互換性を維持する必要がある場合は、以下の手順を実行してください。
 - a. 314 ページの『Cookie ストリングで使用する UTF-8 エンコードの指定』
 - b. 320 ページの『存続時間タイム・スタンプの妥当性検査』
 - c. 320 ページの『アクティビティ・タイム・スタンプの妥当性検査』
8. ご使用のデプロイメントに適用できるすべての手順が完了したら、WebSEAL サーバーを再始動します。

フェイルオーバー Cookie のプロトコル

デフォルトでは、フェイルオーバー認証 Cookie は使用不可になっています。フェイルオーバー Cookie を使用可能にするには、WebSEAL 構成ファイルを編集します。

[failover] スタンザに、フェイルオーバー Cookie による要求の処理方法を WebSEAL に指示する値を指定します。次の表に、有効な値を示します。

表 33. フェイルオーバー Cookie 用にサポートされているプロトコル

スタンザ・エントリー	説明
failover-auth = http	フェイルオーバー Cookie が HTTP プロトコルで使用可能になります。
failover-auth = https	フェイルオーバー Cookie が HTTPS (SSL) プロトコルで使用可能になります。
failover-auth = both	フェイルオーバー Cookie が HTTP プロトコルおよび HTTPS (SSL) プロトコルの両方で使用可能になります。

注: フェイルオーバー認証を HTTP または HTTPS のどちらかで使用可能にすると、すべての プロトコルで接続されているクライアントに Cookie が書き込まれます。**failover-auth** スタンザ・エントリーで指定される値は、フェイルオーバー認証イベント中に認証される Cookie が使用するプロトコルを示しています。

Cookie データを暗号化および暗号化解除する鍵ペアの生成

このタスクについて

Cookie データを保護する鍵ペアを生成するには、LMIを使用します。WebSEAL がこのユーティリティを提供します。フェイルオーバー Cookie のデータを暗号化および暗号化解除できる対称鍵ペアを生成できます。

注:

- 特定のロード・バランシング環境 (フェイルオーバー用に構成された環境) 用に生成された鍵ペア (Cookie データの暗号化および暗号化解除に使用) を他のロード・バランシング環境で再使用しないでください。常に、フェイルオーバー認証用に構成されたロード・バランシング環境ごとに固有キーのペアを生成します。

- フェイルオーバー認証 Cookie を暗号化するように WebSEAL を構成していない場合に、フェイルオーバー認証を使用可能にした場合、WebSEAL はエラーを生成し、開始することを拒否します。フェイルオーバー認証 Cookie は、暗号化しなければなりません。

手順

1. LMI を使用して鍵ファイル (例: `ws.key`) を生成します。鍵ファイルを作成するには、「SSO キー」管理ページを使用します。このページにアクセスするには、「セキュア・リバース・プロキシ設定 (Secure - Reverse Proxy Settings)」>「グローバル・キー」>「SSO キー」にアクセスします。
2. WebSEAL 構成ファイルを編集します。**[failover]** スタンザで、鍵ファイルを指定します。

```
[failover]
failover-cookies-keyfile = keyfile_name
```
3. 鍵格納ファイルを、残りの複製サーバーのそれぞれに手動でコピーします。
4. それぞれの複製サーバーで、WebSEAL 構成ファイルを編集し、**[failover]** スタンザの **failover-cookies-keyfile** に正しいパス名を指定します。

フェイルオーバー Cookie 存続時間の指定

このタスクについて

WebSEAL 構成ファイルで、フェイルオーバー Cookie 存続時間の値を指定します。

手順

WebSEAL 構成ファイルを編集します。フェイルオーバー Cookie に、有効な存続時間 (分) を指定します。デフォルトの存続時間は 60 分です。以下に例を示します。

```
[failover]
failover-cookie-lifetime = 60
```

注: WebSEAL は標準 HTTP フェイルオーバー Cookie に含まれている **expires** 属性を利用しません。代わりに、トークン自体に含まれている有効期限値を利用します。その結果、ブラウザの終了などによってセッションが終了した場合や、トークンの有効期限が切れた場合に、Cookie の存続期間が切れます。

Cookie スtringで使用する UTF-8 エンコードの指定

このタスクについて

Cookie 内のユーザー名または資格情報属性が、WebSEAL サーバーが使用しているものと同じコード・ページでエンコードされていないときは、UTF-8 を使用してください。デフォルトで、WebSEAL サーバーは UTF-8 エンコードを使用します。WebSEAL デプロイメント内のすべての WebSEAL サーバーが UTF-8 エンコードを使用しているときは、この値はデフォルト設定の「yes」のままにしておきます。

手順

WebSEAL 構成ファイルを編集します。WebSEAL が、フェイルオーバー Cookie 内のストリングに UTF-8 エンコードを使用するかどうかを指定します。

```
[failover]
use-utf8 = yes
```

デフォルト値は「yes」です。

認証強度レベルの追加

このタスクについて

認証強度レベルをフェイルオーバー認証 Cookie で指定するには、認証レベルを WebSEAL 構成ファイルに追加します。

手順

以下のように **AUTHENTICATION_LEVEL** スタンザ・エントリーを使用します。

```
[failover-add-attributes]
AUTHENTICATION_LEVEL = add
```

AUTHENTICATION_LEVEL の実際の値は、WebSEAL が内部でトラッキングする整数です。ユーザーがこのスタンザに整数を指定する必要はありません。

欠落したフェイルオーバー Cookie の再発行

プロキシ環境では、有効なセッションを持ったクライアントがフェイルオーバー Cookie を失う場合があります。このようなクライアントは、最初の WebSEAL システムとのセッションを引き続き維持できます。しかし、フェイルオーバー Cookie がないと、クライアントは新規システムにフェイルオーバーできません。

WebSEAL 構成ファイルの **[failover]** スタンザにある **reissue-missing-failover-cookie** スタンザ・エントリーを使用して、クライアントが、フェイルオーバー認証が使用可能な場合にセッション中にフェイルオーバー Cookie を常に保有するようにできます。有効な値は、「yes」(使用可能) および「no」(使用不可) です。

フェイルオーバー Cookie の再発行メカニズムは、デフォルトでは使用不可になっています。以下に例を示します。

```
[failover]
reissue-missing-failover-cookie = no
```

reissue-missing-failover-cookie = yes の場合、WebSEAL は、クライアントの WebSEAL セッション・キャッシュ・エントリーにクライアント用に生成されたフェイルオーバー Cookie を保管します。前の Cookie の内容がキャッシュ・エントリーに既に保管されている場合、それらは削除され、新規の Cookie データに置き換えられます。

クライアントが、WebSEAL サーバーに後続の要求を作成し、その要求にフェイルオーバー Cookie を提供しない場合、WebSEAL は以下の条件に基づいて、クライアントに対する応答でキャッシュされている元のフェイルオーバー Cookie を再発行します。

- フェイルオーバー cookie の再発行メカニズムが使用可能であること。
`reissue-missing-failover-cookie = yes`
- クライアントが有効なセッションを保持していること。
- このクライアント・タイプでフェイルオーバー認証が使用可能であること。
- このクライアントのフェイルオーバー Cookie が、このクライアントのセッション・キャッシュ・エントリーに保管されていること。
- 他のメカニズムがこの要求に対して新規のフェイルオーバー Cookie を生成していないこと。

セッション存続時間タイム・スタンプの追加

WebSEAL は、以下の値を結合してセッション存続時間タイム・スタンプを計算します。

- 現行システム時刻。
- WebSEAL 資格情報キャッシュの中にエントリーが存在することを許される最大存続時間 (秒数)。

この最大存続時間 (秒数) が、WebSEAL 構成ファイルの **[session]** スタンザに指定されます。

```
[session]
timeout = 3600
```

この値をフェイルオーバー認証 Cookie に追加するには、手動で WebSEAL 構成ファイルを編集して、次のエントリーを追加します。

```
[failover-add-attributes]
session-lifetime-timestamp = add
```

フェイルオーバー・インシデントが発生すると、セッションの存続時間値を使用して、ユーザーのセッションが存続できる残り時間を判別します (セッション存続時間値はフェイルオーバー時にリセットされません)。セッション存続時間を超えると、ユーザーはログインしなければなりません。

この属性は、ワイルドカード・マッチングを使用して設定できないことに注意してください。完全なエントリー **session-lifetime-timestamp** を入力する必要があります。

セッション・アクティビティー・タイム・スタンプの追加

このタスクについて

WebSEAL は、以下の 3 つの値を合計してセッション・アクティビティー・タイム・スタンプを計算します。

- 現行システム時刻。

例: 13:30

- ユーザーのセッションが非アクティブでいられる最大時間 (秒数)。

非アクティブ・キャッシュ・エントリーの最大存続時間は、WebSEAL 構成ファイルの **[session]** スタンザに設定されます。次に例を示します。

```
[session]
inactive-timeout = 600
```

デフォルト値は 600 秒です。

- フェイルオーバー認証 Cookie の更新と更新の間の時間間隔 (秒数)。

この値は、WebSEAL 構成ファイルの **[failover]** スタンザに次のように設定されます。例:

```
[failover]
failover-update-cookie = 300
```

デフォルト値は -1 秒です。詳細については、『アクティビティー・タイム・スタンプの更新間隔の追加』を参照してください。

この例のタイム・スタンプ値は 13:50 です。

手順

WebSEAL 構成ファイルを編集し、以下のエントリーを追加します。

```
[failover-add-attributes]
session-activity-timestamp = add
```

注: この属性をワイルドカード・マッチングで設定することはできません。エントリー **session-activity-timestamp** を正確に入力する必要があります。

注: **failover-update-cookie** をゼロより大きい数値に設定するときは、**session-activity-timestamp = add** も必ず設定してください。
session-activity-timestamp = add を設定しなかった場合、WebSEAL はユーザー・アクセスごとに、フェイルオーバー Cookie をデコードします。この繰り返されるアクションは、パフォーマンスに悪い影響を与えます。

アクティビティー・タイム・スタンプの更新間隔の追加

必要に応じて、フェイルオーバー Cookie 内のセッション・アクティビティー・タイム・スタンプは、ユーザーのセッション中に更新することができます。

このエントリーには、フェイルオーバー Cookie のアクティビティー・タイム・スタンプを更新するのに使用される間隔 (秒数) を指定する整数値が入ります。

デフォルト・エントリーは、以下のとおりです。

```
[failover]
failover-update-cookie = -1
```

failover-update-cookie が 0 に設定されていると、最後のアクティビティー・タイム・スタンプが各要求で更新されます。

failover-update-Cookie が 0 より小さい整数 (負の数値) に設定されていると、最後のアクティビティ・タイム・スタンプが更新されることはありません。

failover-update-Cookie が 0 より大きい整数に設定されていると、Cookie 内のセッション・アクティビティ・タイム・スタンプが、この秒数のインターバルで更新されます。

このスタンザ・エントリーに選択された値が、パフォーマンスに影響を与えます。306 ページの『フェイルオーバー Cookie へのデータの追加』を参照してください。

注: failover-update-cookie をゼロより大きい数値に設定するときは、session-activity-timestamp = add も必ず設定してください。
session-activity-timestamp = add を設定しなかった場合、WebSEAL は各ユーザー・アクセスごとに、フェイルオーバー Cookie をデコードします。この繰り返されるアクションは、パフォーマンスに悪い影響を与えます。316 ページの『セッション・アクティビティ・タイム・スタンプの追加』を参照してください。

追加の拡張属性

必要に応じて、WebSEAL は、ユーザー資格情報にある指定した拡張属性のコピーを、フェイルオーバー認証 Cookie に入れるよう構成できます。デフォルトでは、拡張属性は構成されていません。

拡張属性を追加するには、エントリーを、WebSEAL 構成ファイルの **[failover-add-attributes]** スタンザに追加します。構文は次のようになります。

```
[failover-add-attributes]
attribute_pattern = add
```

attribute_pattern は、特定の属性名でもかまいませんし、あるいは、複数の属性名に一致する、大文字小文字を区別しないワイルドカード式でもかまいません。例えば、プレフィックスが **tagvalue_** になっているすべての属性を指定するには、以下のエントリーを追加します。

```
[failover-add-attributes]
tagvalue_* = add
```

スタンザ・エントリーの順序は重要ではありません。**[failover-add-attributes]** スタンザの中で早く現れるルールが、スタンザの後のほうに入れられるルールよりも優先します。

ワイルドカード・パターンに一致しない属性、あるいは明示的に指定されていない属性は、フェイルオーバー Cookie に追加されません。

抽出のための属性

必要に応じて、WebSEAL を、フェイルオーバー認証 Cookie から属性を抽出し、これをユーザー資格情報に入れるように構成できます。デフォルトでは、抽出のために構成されている拡張属性はありません。

抽出する属性は、WebSEAL 構成ファイルの **[failover-restore-attributes]** スタンザで宣言されます。構文は次のようになります。

```
[failover-restore-attributes]
attribute_pattern = {preserve|refresh}
```

値 `preserve` は、WebSEAL に、属性を抽出して、これを資格情報に追加するように指示するものです。

値 `refresh` は、WebSEAL に、属性を無視し、Cookie から抽出しないように指示するものです。

`attribute_pattern` は、特定の属性名でもかまいませんし、あるいは、複数の属性名に一致する、大文字小文字を区別しないワイルドカード式でもかまいません。例えば、`tagvalue_` というプレフィックスのすべての属性を抽出するには、以下のエントリーを追加します。

```
[failover-restore-attributes]
tagvalue_* = preserve
```

値 `preserve` で指定したパターンに一致しない属性は、フェイルオーバー認証 Cookie から抽出されません。

スタンザ・エントリーの順序は重要ではありません。**[failover-restore-attributes]** スタンザの中で早く現れるルールが、スタンザの後のほうに入れられるルールよりも優先します。

以下の属性は、ワイルドカード・パターンでマッチングされませんが、抽出のために明示的に定義する必要があります。

- 認証レベル

```
[failover-restore-attributes]
AUTHENTICATION_LEVEL = preserve
```

- セッション存続時間タイム・スタンプ

```
[failover-restore-attributes]
session-lifetime-timestamp = preserve
```

- セッション・アクティビティー・タイム・スタンプ

```
[failover-restore-attributes]
session-activity-timestamp = preserve
```

ドメイン全体のフェイルオーバー Cookie の使用可能化

このタスクについて

フェイルオーバー認証 Cookie を、Cookie を作成した WebSEAL サーバーと同じドメイン内のどの WebSEAL サーバーでも使用できるようにできます。この機能は、**[failover]** スタンザのスタンザ・エントリーで制御されます。

デフォルトでは、ドメイン全体のフェイルオーバー Cookie 機能は使用不可にされます。

```
[failover]
enable-failover-cookie-for-domain = no
```

手順

ドメイン全体のフェイルオーバー Cookie 機能を使用可能にするには、**enable-failover-cookie-for-domain** を、次に示すように「yes」に設定します。

```
[failover]
enable-failover-cookie-for-domain = yes
```

このスタンザ・エントリーを使用可能にした場合の影響については、310 ページの『ドメイン全体にわたるフェイルオーバー認証』を参照してください。

存続時間タイム・スタンプの妥当性検査

必要に応じて、WebSEAL サーバーを、各フェイルオーバー認証 Cookie がセッション存続時間タイム・スタンプを持つことを必要とする ように構成できます。セッション存続時間タイム・スタンプは、デフォルトでは必須ではありません。デフォルトの構成ファイルのエントリーは次のようになっています。

```
[failover]
failover-require-lifetime-timestamp-validation = no
```

このスタンザ・エントリーは、主に、WebSEAL の前のバージョンとの互換性のために使用されます。バージョン 5.1 より前の WebSEAL サーバーによって作成されたフェイルオーバー認証 Cookie には、このタイム・スタンプがありません。バージョン 5.1 より前の WebSEAL サーバーで作成されたフェイルオーバー Cookie との互換性が必要な場合は、このエントリーを「no」に設定します。

- この値が「no」で、さらに、セッション存続時間タイム・スタンプがフェイルオーバー Cookie から欠落しているとき、受け取り側のサーバーは、この Cookie を有効 と見なします。
- この値が「yes」で、さらに、セッション存続時間タイム・スタンプがフェイルオーバー Cookie から欠落しているとき、受け取り側のサーバーは、この Cookie を無効 と見なします。
- この値が「no」または「yes」であり、セッション存続時間タイム・スタンプがフェイルオーバー Cookie に存在しているとき、受け取り側のサーバーはタイム・スタンプを評価します。タイム・スタンプが無効な場合、認証は失敗します。タイム・スタンプが有効な場合、認証プロセスは先に進みます。

注: セッション存続時間タイム・スタンプは、セッション・アクティビティー・タイム・スタンプとは別に構成されます。

アクティビティー・タイム・スタンプの妥当性検査

必要に応じて、WebSEAL サーバーを、各フェイルオーバー認証 Cookie がセッション・アクティビティー・タイム・スタンプを持つことを必要とする ように構成できます。セッション・アクティビティー・タイム・スタンプは、デフォルトでは必須ではありません。デフォルトの構成ファイルのエントリーは次のようになっています。

```
[failover]
failover-require-activity-timestamp-validation = no
```

このスタンザ・エントリーは、主に、WebSEAL の前のバージョンとの互換性のために使用されます。

- この値が「no」で、さらに、セッション・アクティビティー・タイム・スタンプがフェイルオーバー Cookie から欠落しているとき、受け取り側のサーバーは、この Cookie を有効 と見なします。
- この値が「yes」で、さらに、セッション・アクティビティー・タイム・スタンプがフェイルオーバー Cookie から欠落しているとき、受け取り側のサーバーは、この Cookie を無効 と見なします。
- この値が「no」または「yes」であり、セッション・アクティビティー・タイム・スタンプがフェイルオーバー Cookie に存在しているとき、受け取り側のサーバーはタイム・スタンプを評価します。タイム・スタンプが無効な場合、認証は失敗します。タイム・スタンプが有効な場合、認証プロセスは先に進みます。

注: セッション・アクティビティー・タイム・スタンプは、セッション継続時間タイム・スタンプとは別に構成されます。

非スティッキー・フェイルオーバー環境のフェイルオーバー

このセクションは、以下のトピックで構成されています。

- 『非スティッキー・フェイルオーバーの概念』
- 323 ページの『非スティッキー・フェイルオーバー・ソリューションの構成』
- 324 ページの『既存の WebSEAL 機能と連動したフェイルオーバー Cookie の使用』

非スティッキー・フェイルオーバーの概念

WebSEAL フェイルオーバー認証は、クライアント要求がロード・バランシング・メカニズムによって複数の複製された WebSEAL サーバーに送信されるフォールト・トレラント環境に適したソリューションです (303 ページの『フェイルオーバー認証の概念』を参照)。それぞれのレプリカ・サーバーが同じ内容と構成を保有しています。クライアントが接続されているサーバーが突然使用不可になった場合、ロード・バランサーが要求を他の複製サーバーの 1 つに宛先変更します。

WebSEAL は、最初のユーザー認証中に暗号化されたフェイルオーバー Cookie を発行することによって、このようなフェイルオーバー・イベントを処理できます。レプリカ・サーバーは、このフェイルオーバー Cookie を暗号化および暗号化解除するために使用する共通の暗号鍵を共有します。この Cookie は、別の WebSEAL レプリカに提示される場合、新規のレプリカ上でのユーザーの認証、ユーザー資格情報の作成、および新規セッションの作成に必要な十分な情報を (レプリカ自体の ID の証明と共に) 含んでいます。フェイルオーバーが発生すると、ユーザーは追加のログイン・プロンプトを受け取りません。

フォールト・トレラント環境では、ロード・バランサーは要求の負荷の物理的な分散を管理します。ここでの説明では、「スティッキー」および「非スティッキー」という用語を使用して、クライアントと特定のサーバーの間の接続を維持する (または維持しない) ためのロード・バランシング・メカニズムの機能を説明します。

- **スティッキー負荷分散**では、クライアントと特定のサーバー間の接続が維持されます。この状態は、ステートフルとも呼ばれ、サーバー・アフィニティを監視することを指します。
- **非スティッキー分散**では、クライアントと特定のサーバー間の接続が維持されません。この状態は、ステートレスとも呼ばれ、サーバー・アフィニティを維持しないことを指します。

スティッキー環境のシナリオでは、ロード・バランサーは、クライアントにセッション中 1 つのレプリカと通信させます。このレプリカに障害が発生するというまれなケースでは、すべての後続の要求は、ロード・バランサーが決定した別のレプリカに送信されます。

非スティッキー環境では、ロード・バランサーはクライアントを 1 つのレプリカに縛り付けません。ユーザー・セッションの実行中、複数の要求を使用可能な任意のレプリカ・サーバーにいつでも送信できます。

いずれの環境においても、サーバーはセッション情報を共有しません。クライアントとサーバー間のセッション状態を維持するのは WebSEAL の担当です。WebSEAL は、クライアント・ブラウザにあるセッション Cookie を使用してセッション状態を維持します。

クライアント要求が別のレプリカ・サーバーに切り替えられても、フェイルオーバー Cookie では追加のログインは必要ありません。ただし、新規のレプリカには他のレプリカのセッション状態がまったくわからないため、新規のセッション Cookie を作成して以前のセッション Cookie と置き換えます。非スティッキーなロード・バランシング環境では、新規 Cookie の発行と新規セッションの確立を頻繁に行うため、ユーザー・セッションの応答性と WebSEAL のパフォーマンスの両方が著しく悪化します。

このパフォーマンス上の問題は、すべてのレプリカがクライアントに発行された元のセッション Cookie を再使用することによって軽減できます。WebSEAL がユーザーの元のセッション ID を拡張属性としてフェイルオーバー Cookie に保管するように構成できます。

クライアントの要求を処理する各レプリカは、元のセッション ID を使用してセッション・キャッシュ・エントリーを作成します。後続のレプリカに戻る間に、フェイルオーバー Cookie 属性の ID とセッション Cookie の ID を比較することによって、サーバー ID をセキュアな方法で検査します。この ID の突き合わせが一致した場合、サーバーは既存のセッション・キャッシュ・エントリーを使用し、ブラウザに対して新規のセッション Cookie を発行しません。

注: 可能な場合は、ロード・バランサーでセッション・アフィニティを維持するように構成します。セッション・アフィニティによって、パフォーマンスが改善され、ユーザー・エクスペリエンスが向上し、WebSEAL 構成が簡単になります。

非スティッキー・フェイルオーバー・ソリューションの構成

このタスクについて

以下の構成ステップによって、WebSEAL によるクライアントの元のセッション ID の再使用を可能にして、非スティッキー・ロード・バランシング環境におけるフェイルオーバー認証応答およびパフォーマンスを改善します。WebSEAL は、ID を拡張属性としてフェイルオーバー Cookie に保管することによって元のセッション ID を再使用します。

フェイルオーバー Cookie にあるユーザーの元のセッション ID を組み込む機能を使用可能にするには、WebSEAL 構成ファイルの **[failover]** スタンザ内の **failover-include-session-id** スタンザ・エントリーを「yes」に設定します。

```
[failover]
failover-include-session-id = yes
```

非スティッキー・フェイルオーバー・ソリューションを使用可能にした場合 (**failover-include-session-id = yes**)、以下の 4 つのスタンザ・エントリーを正しく構成する必要があります。これらの設定のいずれかに誤りがある、WebSEAL が開始エラーを報告し、開始は失敗します。

手順

1. 非スティッキー・フェイルオーバー・ソリューションでは、HTTPS を使用してセッション状態を維持するために、SSL セッション ID ではなく WebSEAL セッション Cookie の使用が必要です。WebSEAL 構成ファイルの **[session]** スタンザ内の **ssl-id-sessions** スタンザ・エントリーが「no」(デフォルト) に設定されていることを確認します。

```
[session]
ssl-id-sessions = no
```

2. フェイルオーバー Cookie に拡張属性として存在するユーザーの元のセッション ID をエンコードするには、WebSEAL 構成ファイルの **[failover-add-attributes]** スタンザ内の **tagvalue_failover_amweb_session_id** スタンザ・エントリーを「add」に設定します。

```
[failover-add-attributes]
tagvalue_failover_amweb_session_id = add
```

3. ユーザー・セッションが初めて別のレプリカに切り替わるとき、WebSEAL (そのレプリカ上の) はフェイルオーバー Cookie に含まれる情報を使用して、そのユーザーの資格情報およびセッション・キャッシュ・エントリーを作成する必要があります。(フェイルオーバー Cookie 内にエンコードされた) セッション ID がユーザー資格情報に必ず追加されるようにするには、WebSEAL 構成ファイルの **[failover-restore-attributes]** スタンザ内の **tagvalue_failover_amweb_session_id** スタンザ・エントリーを「preserve」に設定します。

```
[failover-restore-attributes]
tagvalue_failover_amweb_session_id = preserve
```

4. 資格情報のリフレッシュ機能によって、ユーザーは **pdadmin** コマンドを発行して、オンデマンドでユーザー資格情報の内容を更新できます (254 ページの『資格情報リフレッシュ』を参照)。非スティッキー・フェイルオーバー・パフォーマンス・ソリューションで資格情報のリフレッシュ中に使用されるセッション ID 属性を保存するには、WebSEAL 構成ファイルの **[credential-refresh-**

attributes] スタンザ内の **tagvalue_failover_amweb_session_id** スタンザ・エントリーを「preserve」に設定する必要があります。

```
[credential-refresh-attributes]  
tagvalue_failover_amweb_session_id = preserve
```

既存の WebSEAL 機能と連動したフェイルオーバー Cookie の使用

以下の情報は、非スティッキー・フェイルオーバー・ソリューションが他の WebSEAL 機能に与える影響について説明したものです。

- **Switch-user**

ユーザー切り替え機能では、フェイルオーバー認証はサポートされません。したがって、非スティッキー・フェイルオーバー・ソリューションもサポートされません。

- **認証方式**

非スティッキー・フェイルオーバー・ソリューションは、サポートされる他の WebSEAL 認証方式には影響を与えません。

- **再認証**

再認証はユーザーのセッション ID を変更しないため、非スティッキー・フェイルオーバー・ソリューションは再認証に影響を与えません。

- **認証強度 (ステップアップ)**

認証強度はユーザーのセッション ID を変更しないため、非スティッキー・フェイルオーバー・ソリューションは認証強度ポリシー (ステップアップ) に影響を与えません。

- **資格情報リフレッシュ**

WebSEAL 構成ファイルの **[credential-refresh-attributes]** スタンザ内の **tagvalue_failover_amweb_session_id** スタンザ・エントリーによって、資格情報のリフレッシュ操作中、セッション ID 情報をユーザーの資格情報内に保存することができます。しかし、資格情報リフレッシュのコマンドを使用して、複数のレプリカ・サーバーへの資格情報リフレッシュを制御することはできません。サーバー・クラスター環境で資格情報リフレッシュ操作を実行する場合は、レプリカ・セットのレプリカ・メンバーごとに資格情報リフレッシュのコマンドを発行する必要があります。

フェイルオーバー環境におけるパスワード変更操作

認証中のパスワード変更操作は、非スティッキー・ロード・バランシング環境では悪影響を与える場合があります。例えば、ユーザーは期限切れのパスワード・フォームを受け取り、フォームに必要なパスワード変更情報をすべて入力します。入力済みフォームを送信するとき、ロード・バランサーは異なるレプリカ・サーバーに接続します。この新規サーバーは元のサーバーとの前の接続を認識していないため、ユーザーにログインを求めるプロンプトを表示します。ユーザーは旧パスワードを入力し、再度期限切れパスワード・フォームを受け取ります。

WebSEAL 構成ファイルの **[acct-mgt]** スタンザ内の **change-password-auth** スタンザ・エントリーによって、パスワード変更操作中の追加ログイン要求を回避することができます。change-password-auth = yes と設定すると、新規のレプリカ・サーバーはパスワード変更要求内の既存の認証情報（ユーザー名、元のパスワード、新規パスワード）を使用してユーザーを認証したり、ユーザーのパスワードを変更したりできます。

フェイルオーバー環境でこの制御されたパスワード変更操作を使用可能にするには、以下のように設定します。

```
[acct-mgt]
change-password-auth = yes
```

バージョン 6.0 より前のバージョンの WebSEAL との互換性のため、デフォルトの設定は「no」です。

第 17 章 非クラスター環境でのセッション状態

この章では、クライアントと単一 WebSEAL サーバー (非クラスター・サーバー環境) 間のセッション状態を管理するための基本的な概念と手順を説明します。

複数の WebSEAL サーバーがリソースの保護を提供している環境については、363 ページの『第 20 章 SMS を使用した WebSEAL の構成』を参照してください。

個別のユーザー・セッションやすべてのユーザー・セッションを終了するには、675 ページの『バックエンド・サーバーのユーザー・セッション管理』を参照してください。

この章で扱うトピックは以下のとおりです。

- 『非クラスター環境でのセッション状態の維持』
- 331 ページの『セッション Cookie』
- 333 ページの『古いセッション Cookie に対するカスタマイズ応答』
- 336 ページの『HTTP ヘッダーを使用したセッション状態の維持』
- 339 ページの『Microsoft Office アプリケーションとのセッションの共用』

非クラスター環境でのセッション状態の維持

このセクションは、以下のトピックで構成されています。

- 『SSL を介したセッション状態情報のコントロール』
- 328 ページの『異なるトランスポートでの同一セッション鍵の使用』
- 329 ページの『有効なセッション鍵データ・タイプ』
- 330 ページの『有効なセッションのタイムアウト値』
- 330 ページの『use-same-session に対する Netscape 4.7x の制限』

SSL を介したセッション状態情報のコントロール

WebSEAL 構成ファイルの `[session]` スタンザにある `ssl-id-sessions` スタンザ・エントリーにより、HTTPS を介してアクセスするクライアント用のログイン・セッションを維持するために、SSL セッション ID を使用するか別のセッション鍵データ・タイプを使用するかをコントロールできます。

スタンザ・エントリー値が「yes」に設定されている場合、すべての認証方式に SSL セッション ID が使用されます。例:

```
[session]
ssl-id-sessions = yes
```

スタンザ・エントリー値が「no」(デフォルト) に設定されている場合、ほとんどの認証方式にセッション Cookie が使用されます。例:

```
[session]
ssl-id-sessions = no
```

このスタンザ・エントリーを「no」に設定すると、HTTPS を介してアクセスするクライアントについて以下の条件が発生します。

- セッション状態の維持には、SSL セッション ID は決して使用されません。
- HTTP ヘッダーを使用して認証するクライアントの場合は、HTTP ヘッダーがセッション ID データとして使用されます。
- IP アドレスを使用して認証するクライアントの場合は、IP アドレスがセッション ID データとして使用されます。
- Cookie は、他のすべての方式を使用して認証するクライアントとのセッションを維持するのに使用されます。

329 ページの『有効なセッション鍵データ・タイプ』を参照してください。

異なるトランスポートでの同一セッション鍵の使用

クライアントが、あるタイプのトランスポート (例えば HTTP) を介して認証し、セッションを確立し、そして別のタイプのトランスポート (例えば HTTPS) を介して接続するときに、WebSEAL が同じセッションを使用するように、または使用しないように構成することができます。

WebSEAL 構成ファイルの `[session]` スタンザにある `use-same-session` スタンザ・エントリーでは、以下の 2 つの選択項目が提供されています。

• いいえ

クライアントがあるトランスポートを介して認証し、セッションを確立し、その後別のトランスポートを介して接続する場合、クライアントは再度認証を行う必要があります。2 番目のセッション鍵を使用して別のセッションが作成されます。この 2 つのセッションは、WebSEAL セッション・キャッシュに個別に維持されます。将来の接続で使用する適切なセッション鍵は、クライアントが使用するトランスポートによって決定されます。

```
[session]
use-same-session = no
```

• はい

クライアントがあるトランスポートを介して認証し、セッションを確立し、その後別のトランスポートを介して接続する場合、クライアントは、最初のトランスポート用に作成したものと同一のセッションと対応するセッション鍵を使用します。クライアントは、再度認証を行う必要はありません。

```
[session]
use-same-session = yes
```

このスタンザ・エントリーを「yes」に設定すると、以下の条件が発生します。

- すべてのトランスポート・タイプで HTTP ヘッダーを使用してアクセスするクライアントのセッションを維持するには、HTTP ヘッダーが使用されます。
- IP アドレスを使用してアクセスするクライアントのセッションを維持するには、IP アドレスが使用されます。
- その他すべての認証方式のセッションを維持するには、セッション Cookie が使用されます。

- **ssl-id-sessions** 構成は無視されます。したがって、結果の動作は、**ssl-id-sessions** が「no」に設定されている場合と同じになります。

HTTP クライアントは、セッション・データとして使用できる SSL セッション ID がないので、このロジックは重要です。

- Cookie は HTTP クライアントと HTTPS クライアントのどちらにも使用できるので、「セキュア」 Cookie 属性でフラグは立てられません。

『有効なセッション鍵データ・タイプ』を参照してください。

有効なセッション鍵データ・タイプ

認証方式ごとに WebSEAL で使用されるセッション鍵データ・タイプを構成することができます。

認証方式で使用するセッション鍵データ・タイプは、以下の構成項目の特定の組み合わせによって決まります。

- **ssl-id-sessions** スタンザ・エントリーの設定。
- **use-same-session** スタンザ・エントリーの設定。
- **[session-http-headers]** スタンザに定義されたヘッダー名。

テーブルの凡例:

- **C** (セッション Cookie)、**H** (HTTP ヘッダー)、**IP** (IP アドレス)、**SSL** (SSL セッション ID)
- **S-I-S** (ssl-id-sessions); **U-S-S** (use-same-session); **S-H-H** ([session-http-headers])
- スタンザ・エントリーがテーブル・ヘッダーに表示される場合、エントリーの値は「yes」です。それ以外の場合は、値は「no」となります。

HTTPS クライアントに対する有効なセッション鍵データ・タイプ								
認証方式	- - -	S-I-S - -	- S-H-H -	- - U-S-S	S-I-S S-H-H -	S-I-S - U-S-S	- S-H-H U-S-S	S-I-S S-H-H U-S-S
BA	C	SSL	H	C	SSL	C	H	H
CDSSO	C	SSL	H	C	SSL	C	H	H
証明書	C	SSL	H	C	SSL	C	H	H
外部 authn インター フェース	C	SSL	H	C	SSL	C	H	H
フェイルオ ーバー Cookie	C	SSL	H	C	SSL	C	H	H
LTPA Cookie	C	SSL	H	C	SSL	C	H	H
フォーム	C	SSL	H	C	SSL	C	H	H

HTTP クライアントに対して有効なセッション鍵データ・タイプ				
認証方式	- - -	- S-H-H -	- - U-S-S	- S-H-H U-S-S
BA	C	H	C	H
CDSSO	C	H	C	H
証明書	C	H	C	H
外部 認証 インターフェース	C	H	C	H
フェイルオーバー Cookie	C	H	C	H
LTPA Cookie	C	H	C	H
フォーム	C	H	C	H

有効なセッションのタイムアウト値

ssl-id-sessions が「yes」に設定されていると、セッションの実際のタイムアウトは、いくつかの異なる値によって決定されます。

セッション・キャッシュ・エントリーの存続時間タイムアウトは、**[session]** スタンザの中の **timeout** エントリーに設定されます。

セッションの非アクティブ・タイムアウトは、同じスタンザの **inactive-timeout** エントリーによって設定されます。

SSL タイムアウトは **[ssl]** スタンザの中に設定され、ここで、**ssl-v2-timeout** と **ssl-v3-timeout** スタンザ・エントリーの両方が宣言されます。

ssl-id-sessions = **yes** であるときは、実際の有効セッション・タイムアウトは、以下のタイムアウト設定のそれぞれに設定された値のうちの最低 の値に設定されます。

```
[session]
timeout
inactive-timeout
```

```
[ssl]
ssl-v2-timeout
ssl-v3-timeout
```

use-same-session に対する Netscape 4.7x の制限

問題:

WebSEAL に対して行われた要求に URL 内のポート番号 (例えば、<http://webseal:80>) が組み込まれていると、Netscape Navigator バージョン 4.7x 上の **use-same-session** 機能が失敗します。

説明:

WebSEAL がデフォルトの HTTP/HTTPS ポート用に構成されていて、URL 内のポート番号が組み込まれていないと、要求は成功します。WebSEAL が非デフォルト・ポートに構成されていて、`use-same-session = yes` 構成オプションが使用可能になっていると、要求は失敗します。

Netscape 4.7x は、非標準ポート番号を持つホスト名が、別のポート番号を持つホスト名と同じドメインにあるとは考えません。例えば、以下にアクセスした場合、
`https://hostname:443`

WebSEAL により Cookie が設定されます。後でアクセスすると、
`http://hostname:80`

ドメイン:80 はドメイン:443 と同じではないので、Netscape は Cookie を送りません。

対応策:

Netscape Navigator を、バージョン 6.2 以上にアップグレードしてください。

セッション Cookie

このセクションは、以下のトピックで構成されています。

- 『セッション Cookie の概念』
- 332 ページの『セッション Cookie の使用に関する条件』
- 332 ページの『セッション Cookie 名のカスタマイズ』
- 333 ページの『各要求を使用したセッション Cookie の送信』

セッション Cookie の概念

クライアントとサーバーの間のセッション状態を保持する方式の 1 つは、Cookie を使用して、このセッション情報を保持することです。サーバーは、特定のクライアントのセッション鍵を Cookie にパッケージして、それをクライアントのブラウザに送ります。新規要求ごとに、ブラウザは (セッション鍵を保有した) Cookie をサーバーに送り返すことにより、自身を再識別します。

セッション Cookie は、クライアントで使用されているブラウザが SSL セッションについて再折衝を行うまでの時間が極めて短い場合に考えられるソリューションの 1 つです。例えば、Microsoft Internet Explorer ブラウザーのバージョンによっては、2 分から 3 分ごとに SSL セッションについて再折衝します。

セッション Cookie は、Cookie を生成したマシン以外のマシンに受け渡すことができないサーバー固有の Cookie です。セッション Cookie は、ブラウザを使用して、クライアントが以前に認証した単一で固有のサーバーとそのセッション Cookie 自体を再識別させることができます。セッション Cookie を使用している場合、WebSEAL は、クライアントに別のログインを求めるプロンプトを出す必要はありません。

セッション Cookie に保管されたセッション鍵には、サーバーのセッション・キャッシュのインデックスに使用する乱数 ID (「鍵」) のみが含まれています。セッション Cookie で公開される情報は、他には何もありません。

セッション Cookie の使用に関する条件

セッション Cookie には、以下の基本的な条件が適用されます。

- セッション Cookie にはセッション情報のみが含まれ、識別情報は含まれていません。
- セッション Cookie は、ブラウザー・メモリーにのみ存在します (ディスク上のブラウザー Cookie jar には書き込まれません)。
- セッション Cookie の存続時間は限られています。
- セッション Cookie は、サーバー固有の Cookie です。ブラウザーは、Cookie が作成されたのと同じホストへの要求でのみ、この Cookie を送信できます。
- クライアント・ブラウザーは、Cookie を受け入れるか、拒否するかのどちらかに構成することができます。クライアント・ブラウザーがセッション Cookie を拒否し、その後ログインに成功すると、WebSEAL は、クライアントから出されるそれぞれの要求ごとに、ユーザーを再認証して新しいセッションを確立しなければなりません。ただし、基本認証 (BA) を使用する場合、WebSEAL は BA ヘッダー情報を使用してユーザーを再認証するので、再ログインするようにユーザーにプロンプトが出されることはありません。ただし、再認証およびセッションの作成によるオーバーヘッドにより、サーバーのパフォーマンスが下がることがあります。

セッション Cookie 名のカスタマイズ

ユーザーは、WebSEAL セッション Cookie の名前をカスタマイズできます。WebSEAL 構成ファイルには、TCP および SSL 接続で使用されるセッション Cookie のさまざまなデフォルト名が記述されています。

```
[session]
tcp-session-cookie-name = PD-H-SESSION-ID
ssl-session-cookie-name = PD-S-SESSION-ID
```

デフォルトの Cookie 名の変更に関する条件は、次のとおりです。

- Cookie 名は、英数字でなければなりません。
- Cookie 名は、固有でなければなりません。
- TCP および SSL 通信の両方に同じ Cookie を使用するには、`use-same-session=yes` を構成します。
- これらのスタンザ・エントリーは、ホストおよびドメインの両方のタイプの Cookie に影響を与えます。

各要求を使用したセッション Cookie の送信

このタスクについて

Cookie を使用してセッション状態を維持している場合、Cookie は、ログイン成功の後で 1 回だけブラウザーに送られます。しかし、一部のブラウザーでは、同時に保管できるメモリー内 Cookie の数に制限があります。

環境によっては、アプリケーションは、1 ドメインにつき多数のメモリー内 Cookie をクライアント・システムに入れることがあります。このような場合、構成済みの WebSEAL セッション Cookie、フェイルオーバー Cookie、または LTPA Cookie はすぐに他の Cookie で置き換えられてしまうことがあります。

セッション Cookie を使用するように WebSEAL を構成すると、WebSEAL 構成ファイルの `[session]` スタンザにある `resend-webseal-cookies` スタンザ・エントリーも設定できます。このスタンザ・エントリーにより、セッション Cookie をもともと含んでいた要求に対するすべての応答用に、WebSEAL よりセッション Cookie がブラウザーに再送されます。この方法で、セッション Cookie がブラウザー・メモリー内に確実に残されるようにすることができます。

resend-webseal-cookies スタンザ・エントリーのデフォルトの設定は、「no」です。

```
[session]
resend-webseal-cookies = no
```

手順

WebSEAL がセッション Cookie の有無について各要求を検査し、その Cookie を対応する応答に含めることができるようにするには、このスタンザ・エントリーを「yes」に構成します。

```
[session]
resend-webseal-cookies = yes
```

注: **resend-webseal-cookies** スタンザ・エントリーでは、フェイルオーバー Cookie、e-community Cookie、および LTPA Cookie に対しても同じ結果が生成されます。

古いセッション Cookie に対するカスタマイズ応答

このセクションは、以下のトピックで構成されています。

- 『セッションの除去および古いセッション Cookie の概念』
- 335 ページの『古いセッション Cookie に対するカスタマイズ応答の使用可能化』

セッションの除去および古いセッション Cookie の概念

ユーザーが `/pkmslogout` コマンドを使用してセッションをログアウトすると、WebSEAL セッションのキャッシュにあるこのユーザーのエントリーは自動的に除去されます。

このセッションのセッション Cookie がユーザーのブラウザに残っていると、古い失効した Cookie になります。失効した Cookie は、WebSEAL セッション・キャッシュの既存エントリーにマップしなくなります。ユーザーから保護オブジェクトに対する後続の要求が出されると、WebSEAL は認証を要求し、ログイン・フォームを戻します。これらの条件下での新規要求に対する応答は、ユーザーの予想通りのものです。

不明な理由でユーザーのセッションが WebSEAL セッションのキャッシュから除去された場合、ユーザーのブラウザに残存している元のセッションの Cookie は、失効した Cookie になります。失効した Cookie は、WebSEAL セッション・キャッシュの既存エントリーにマップしません。WebSEAL からセッションが除去される理由としては、セッション・タイムアウト、セッション強制終了、セッション終了などがあり、これらはユーザーに不明な場合があります。

ユーザーが保護オブジェクトを要求すると、WebSEAL は認証を要求し、ログイン・フォームを戻します。これらの条件下での新規要求に対するこの応答は、ユーザーにとって予期しないものである可能性があります。

予期しないログイン・プロンプトの理由の説明となる追加情報が含まれるように、ログイン応答をカスタマイズすることができます。カスタマイズ応答を提供するには、以下のステップを実行します。

1. セッション・キャッシュのいずれの既存エントリーにもマップしない失効したセッション Cookie を WebSEAL が受け取った場合には、必ずカスタム・ログイン応答をトリガーする。

『カスタム・ログイン応答のトリガー』を参照してください。

2. `/pkmslogout` コマンドを使用した標準ログアウトの際にブラウザからセッション Cookie を除去するように、WebSEAL を構成します。

『通常ログアウト時のブラウザからの Cookie の除去』を参照してください。

カスタム・ログイン応答のトリガー

WebSEAL の標準ログイン・フォームには、`OLDSESSION` というマクロが含まれています。`OLDSESSION` マクロは、デフォルトのブランク、または値 1 を取ることができます。セッション・キャッシュのいずれの既存エントリーにもマップしない失効したセッション Cookie を WebSEAL が受け取ると、`OLDSESSION` マクロの値が 1 に設定されます。

適切な WebSEAL ログイン・フォーム (`login.html` など) を使用して、以下のアクションを実行できます。

- `OLDSESSION` マクロの値を読み取る。
- マクロの値が 1 になった場合に、ユーザーへのカスタム応答をトリガーする。

カスタム応答により、非アクティブが原因でセッションが終了した可能性があることがユーザーに通知されます。

通常ログアウト時のブラウザからの Cookie の除去

デフォルトでは、WebSEAL は、セッション終了の原因にかかわらず、終了中にクライアント・システムから Cookie を除去しません。終了処理は、クライアントの

ログアウトまたはサーバー側のいずれかによって開始される可能性があります。つまり、セッション終了後にユーザーが別の要求を実行するときには、OLDSESSION マクロの値が必ず 1 に設定されます。その場合、カスタム・ログイン応答を起動することはできません。

手順

ユーザーがログアウトするときに、対応する応答で Cookie を送信することにより、クライアント・システムからの Cookie の除去を試行するように WebSEAL を構成します。

通常、失効した Cookie を WebSEAL が受け取るのは、終了したセッションがユーザーによって開始されていなかった場合のみです。この場合はカスタム・ログイン応答が必要です。

重要: 通常は成功しますが、その一方で、Cookie を除去するように WebSEAL を構成していてもクライアント・システムから Cookie が除去されない場合があります。例えば、`/pkmslogout` を要求して WebSEAL で正常にセッションが終了した後に、ネットワークの問題が原因でログアウト応答が送信されない場合が挙げられます。このような場合は、失効したセッション Cookie がクライアント・システムに残ってしまいます。

このため、セキュリティに影響する決定は、Cookie の有無や OLDSESSION マクロの値に基づいてはなりません。

注: ブラウザーがクローズされた際のセッション Cookie の除去は通常のブラウザーの動作であるため、意図的にログアウトした場合に失効した Cookie がユーザーに残されることはありません。ユーザーがブラウザー・アプリケーションを閉じて意図的にログアウトした後、そのユーザーによる次の要求のときに OLDSESSION マクロは設定されません。

しかし、ユーザー・セッション・キャッシュ・エントリは WebSEAL サーバー (Session Management Server) に残ります。存続時間または非アクティブ・タイムアウトのためにキャッシュ・エントリが期限切れになるまで、`max-concurrent-web-sessions` ポリシー設定の対象としてカウントされ続けます。

古いセッション Cookie に対するカスタマイズ応答の使用可能化

古いセッション Cookie に対するカスタマイズ応答を使用可能にするように WebSEAL を構成することができます。

手順

1. 標準の方法でログアウトしたユーザーのブラウザーからセッションの Cookie を除去するように WebSEAL を構成する。

WebSEAL 構成ファイルの `[session]` スタンザの `logout-remove-cookie` スタンザ・エントリは、標準のログアウトを実行したユーザーのブラウザーからのセッション Cookie の除去を制御します。

ユーザーは、コマンド行に `/pkmslogout` コマンドを入力することで標準の方法でログアウトします。

値 `yes` を指定すると、標準の方法でログアウトしたユーザーのブラウザからの Cookie の除去を試行するように WebSEAL が設定されます。例: `[session] logout-remove-cookie = yes`。

2. 以下の操作を実行するように該当の WebSEAL ログイン・フォーム (`login.html` など) をカスタマイズします。

- `OLDSESSION` マクロの値を読み取る。
- マクロの値が `1` に設定されている場合に、ユーザーに対するカスタム応答を生成する。

以下のいずれのツールを使用しても、ログイン・フォームの `OLDSESSION` マクロを検査することができます。

- Javascript
- HTML Meta タグ
- ローカル応答リダイレクト

詳細については、103 ページの『ローカル応答リダイレクト』を参照してください。

タスクの結果

バージョン 6.0 より前のバージョンの WebSEAL との互換性:

デフォルト設定の `logout-remove-cookie = no` では、標準の方法でログアウトしたユーザーのブラウザから Cookie を除去しないように WebSEAL が設定されます。例:

```
[session]
logout-remove-cookie = no
```

デフォルト値が `no` である理由は、バージョン 6.0 より前のバージョンの WebSEAL との互換性のためです。

HTTP ヘッダーを使用したセッション状態の維持

このセクションは、以下のトピックで構成されています。

- 『HTTP ヘッダー・セッション鍵の概念』
- 337 ページの『セッション状態を維持するための HTTP ヘッダーの構成』
- 338 ページの『MPA からの要求を求めるためのセットアップ』
-

HTTP ヘッダー・セッション鍵の概念

WebSEAL では、使用する認証方式とは別に HTTP ヘッダーをセッション鍵として使用したセッション状態の維持に対するサポートを提供しています。

例えば、同時モバイル装置やインターネット・ユーザー・サポートを許可するには、Tivoli Federated Identity Manager 環境では、WebSEAL が、事前に提供された HTTP ヘッダーを使用してワイヤレス装置クライアントのセッション状態を維持する必要があります。

このシナリオでは、モバイル装置ユーザーは、認証済み多重プロキシ・エージェント (MPA) ゲートウェイを介して WebSEAL が保護するイントラネットに接続しています。WAP ゲートウェイは Liberty 対応プロキシ (LEP) として動作します。LEP は、Liberty Alliance Project (LAP) によって作成されたネットワーク規格です。

クライアントとのセッション状態は、Mobile Station Integrated Services Digital Network (MSISDN) HTTP ヘッダーを介して維持および管理されます。セッション鍵として使用される HTTP ヘッダーは、要求が認証済み多重プロキシ・エージェント (MPA) を介してプロキシ化されたときにのみ WebSEAL により受け入れられます。

セッション状態を維持するための HTTP ヘッダーの構成

このタスクについて

セッション状態を維持するために HTTP ヘッダーを構成するには、WebSEAL 構成ファイルの `[session-http-headers]` スタンザにヘッダー名を指定します。

各ヘッダーは、トランスポートごとにリストされます。同じヘッダーが、両方のトランスポートにリストされる場合があります。有効なトランスポートには「HTTP」や「HTTPS」などがあります。次の構文を使用します。

```
[session-http-headers]
header-name = http|https
```

例:

```
[session-http-headers]
entrust-client = http
entrust-client = https
```

HTTP ヘッダーのセッション鍵の構成に関する条件は、次のとおりです。

- HTTP ヘッダーをセッション状態の維持に使用できるようにするには、以下を設定する必要があります。

```
[session]
ssl-id-sessions = no
```

- `ssl-id-sessions = yes` である場合、`[session-http-headers]` スタンザは無視されます。MPA サポートが使用可能である場合、例外が発生します。

```
[mpa]
mpa = yes
```

- 認証済み多重プロキシ・エージェント (MPA) を介してプロキシ化された要求では HTTP ヘッダーのみを受け入れるように、WebSEAL を構成する必要があります。338 ページの『MPA からの要求を求めるためのセットアップ』を参照してください。
- セッションの維持に使用するすべてのヘッダーをリストします。
- ヘッダー・リストを、トランスポートごとに 20 エントリー以下に制限します。
- ヘッダー名には、コロン (:) 文字を含めないでください。
- HTTP ヘッダーは、トランスポートごとに使用可能または使用不可に設定できます。

HTTP ヘッダーを使用してセッション状態を確立するためのプロセス・フローは、次のとおりです。

- セッション状態の維持では、セッション Cookie は HTTP ヘッダーより優先順位が常時高くなります。

要求を受信すると、WebSEAL はまずセッション Cookie を検索し、その後継続して構成済み HTTP ヘッダーを探します。

受信した要求に WebSEAL セッション Cookie が含まれている場合、WebSEAL は構成済み HTTP ヘッダーを探しません。

- 要求 (セッション Cookie を含まないもの) に **[session-http-headers]** スタンザのエントリーと一致する HTTP ヘッダーがある場合、そのクライアントのセッション状態の維持にその HTTP ヘッダーが使用されます。
- **[session-http-headers]** スタンザには、複数のヘッダーを入力できます。最初に一致する HTTP ヘッダーが検出されると、そのヘッダーが既存のキャッシュ・エントリーへの鍵であるかどうかにかかわらず、WebSEAL は要求の検索を停止します。

例えば、2 つのヘッダーがヘッダー A、ヘッダー B の順序で構成されており、ヘッダー B を使用してセッションが確立していて、ヘッダー A は何らかの理由で、同じクライアントの後の要求に追加されているとします。そして、WebSEAL は、**[session-http-headers]** スタンザを検索し、ヘッダー A との一致を検出します。この場合、セッション・キャッシュにあるそのクライアントの既存のエントリーはヘッダー B に基づいているため、WebSEAL は既存のセッション・キャッシュ・エントリーを検出せず、ユーザーに認証を求めるプロンプトを出します。

- **[session-http-headers]** スタンザにエントリーが存在しない場合、WebSEAL はセッション Cookie を使用してセッション状態を維持します。
- `ssl-id-sessions = no` が設定されているけれど、受信した要求内で構成済み HTTP ヘッダーが検出されなかった場合、WebSEAL はセッション Cookie を使用してセッションを維持します。

MPA からの要求を求めるためのセットアップ

セッション状態を維持するために HTTP ヘッダーを使用すると、セッション鍵が盗まれたりユーザー・セッションがスプーフされるリスクがあります。モバイル装置を使用するクライアントがいる環境では、MSISDN 電話番号はセッション鍵の値として使用されます。大容量サイズでランダムな性質をもつセッション Cookie のキー値と異なり、電話番号は小さくより予測可能であるため、セキュアな形式ではありません。

セキュア WebSEAL 環境では、HTTP ヘッダー・セッション鍵は、要求が認証済み多重プロキシ・エージェント (MPA) を介してプロキシ化されるときにのみ有効になります。WebSEAL 構成ファイルの **[session]** スタンザにある **require-mpa** スタンザ・エントリーを使用すれば、ユーザーはこの要件をコントロールできます。

「yes」を設定すると、WebSEAL は、認証済み多重プロキシ・エージェント (MPA) を介してプロキシ化された要求からの HTTP ヘッダーのみを受け入れます。

す。WebSEAL は、プロキシ化されたクライアント接続を受け入れる前に、ゲートウェイ自体を認証する必要があります。例:

```
[session]
require-mpa = yes
```

「no」を設定すると、WebSEAL は、すべての条件で HTTP ヘッダーを受け入れることができます。例:

```
[session]
require-mpa = no
```

MPA を使用した WebSEAL のインプリメンテーションでは、次の条件を厳守する必要があります。

- 競合を避けるため、MPA では、その MPA を介して WebSEAL にアクセスするクライアントと同一のセッション鍵を使用できません。

例えば、MPA がセッション Cookie を使用してセッションを維持している場合、クライアント・セッションは、別の手段によって維持する必要があります。

- 競合を避けるため、MPA では、その MPA を介して WebSEAL にアクセスするクライアントと同一の認証方式を使用できません。

例えば、MPA がフォーム認証を使用している場合、クライアントは、外部認証インターフェースなど、その他の手段を使用して認証を行う必要があります。典型的なシナリオでは、MPA では基本認証 (BA) または証明書認証が使用され、クライアントでは外部認証インターフェースが使用されます。

注: HTTP ヘッダー認証をサポートする以前のリリースから 6.1 以降に WebSEAL をアップグレードすると、**require-mpa** のデフォルト値が *no* から *yes* に変わります。HTTP ヘッダー認証を使用している場合に、アップグレードすると、**require-mpa** が *no* に設定されていない限り、認証に失敗します。

Microsoft Office アプリケーションとのセッションの共用

Microsoft Office アプリケーションとのセッション情報の共用をブラウザーに指示するように WebSEAL を構成できます。セッション情報の共用により、Microsoft Office アプリケーションがユーザーを再認証する必要がなくなります。

- 『Microsoft Office アプリケーションとのセッション共用の概要』
- 340 ページの『一時セッション・キャッシュの構成』
- 342 ページの『Microsoft Office アプリケーションとの共用セッションの構成』

Microsoft Office アプリケーションとのセッション共用の概要

クライアント・セッションを維持するために、Cookie を使用するように WebSEAL を構成することができます。セキュリティ上の理由から、WebSEAL は非永続 Cookie を使用します。Internet Explorer および Microsoft Office は永続 Cookie の共有のみが可能なため、Microsoft Office アプリケーションは、デフォルトでは WebSEAL ユーザー・セッションを共用できません。

短期的な 1 回使用の持続セッション Cookie を作成するように WebSEAL を構成することができます。この Cookie は、索引を一時セッション・キャッシュに保管し

ます。WebSEAL はその索引を使用して、標準セッション・キャッシュ内の対応するセッションを見つけます。一時キャッシュ・エントリーは 1 回限りの使用のために作成され、WebSEAL インスタンス間では共有されません。Microsoft Office アプリケーションは、この 1 回使用の持続 Cookie を使用して、対応するユーザー・セッションを Internet Explorer から見つけることができます。

/pkmstempsession URI の要求は、この一時セッション Cookie の作成をトリガーします。/pkmstempsession 要求に、ターゲット・リダイレクト URL を組み込むことができます。/pkmstempsession 要求の処理が完了すると、WebSEAL はクライアントをこの URL にリダイレクトします。リダイレクト URL が指定されていない場合、WebSEAL はデフォルトの結果ページをクライアントに返します。

`http://<server>/pkmstempsession?url=<requested_resource>`

説明:

`<server>`

WebSEAL サーバーの完全修飾ホスト名。

`<requested_resource>`

ターゲット・リソースの場所。

例えば、Microsoft Office 文書の場合は /server/test.doc です。

注: 要求のリソース URL には、必要に応じて照会ストリング引数を含めることができます。これらの引数は、作成された WebSEAL リダイレクト要求内に変更されずに残ります。

1 回使用の持続 Cookie を作成するには、/pkmstempsession URI に要求を送信します。この Cookie の作成は、クライアントがコンテキストを WebSEAL から Microsoft Office に切り替える前に行う必要があります。Microsoft Office 環境での 2 つの一般的なユース・ケースの構成について詳しくは、342 ページの『Microsoft Office アプリケーションとの共用セッションの構成』を参照してください。

一時セッション・キャッシュの構成

セッション共用機能で WebSEAL が使用する一時セッション・キャッシュを構成できます。

一時セッション・キャッシュと関連がある WebSEAL 構成エントリーは 3 つあります。それらのエントリーを使用して、キャッシュ・エントリーの存続時間の制御、一時セッション Cookie の名前の指定、およびデフォルトの結果ページの構成を行うことができます。

- 『一時セッション・キャッシュのエントリーの存続時間の構成』
- 341 ページの『一時セッション Cookie の名前の構成』
- 341 ページの『一時キャッシュ応答ページの構成』

一時セッション・キャッシュのエントリーの存続時間の構成

このタスクについて

一時セッション・キャッシュは、1 回使用の短時間セッションを格納します。WebSEAL はこのキャッシュを使用して、1 回使用の短時間セッションと標準セッ

セッションの間にマップする中間セッションを作成します。一時セッション・キャッシュに対する索引が、永続 Cookie としてクライアントに返されます。

一時セッション・キャッシュのエントリーの最大存続時間 (秒) を設定するには、WebSEAL 構成ファイルの **[session]** スタンザの **temp-session-max-lifetime** エントリーを使用します。

手順

temp-session-max-lifetime の値を 0 に設定すると、一時セッション・キャッシュの使用が無効になります。

例

以下の例では、最大存続時間を 10 秒に構成しています。

```
[session]
temp-session-max-lifetime = 10
```

一時セッション Cookie の名前の構成

このタスクについて

WebSEAL は、**/pkmstempsession** 管理ページに対する要求への応答として、1 回使用の短時間セッション Cookie を作成します。この Cookie は、一時セッション・キャッシュに対する索引を格納します。この索引は、標準セッション・キャッシュの対応するセッションを検索するために、後で WebSEAL が使用します。

手順

一時セッション Cookie の名前を指定するには、WebSEAL 構成ファイルの **[session]** スタンザの **temp-session-cookie-name** エントリーを使用します。

注: この構成項目は、一時セッション・キャッシュが使用可能である場合にのみ有効です。キャッシュを使用可能にするには、**[session]** スタンザの **temp-session-cookie-name** エントリーにゼロ以外の値を設定する必要があります。

例

以下の例では、PD-TEMP-SESSION-ID という Cookie 名を構成します。

```
[session]
temp-session-cookie-name = PD-TEMP-SESSION-ID
```

一時キャッシュ応答ページの構成

このタスクについて

デフォルトの管理ページを設定するには、WebSEAL 構成ファイルの **[acct-mgt]** スタンザの **temp-cache-response** エントリーを使用します。pkmstempsession 要求でリダイレクト URL が指定されていない場合、WebSEAL は pkmstempsession 要求の処理後にこのページを返します。

デフォルトでは、WebSEAL は **temp_cache_response.html** を返します。


```
[acct-mgt]  
temp-cache-response = temp_cache_response.html
```

例

以下の例に、代表的な要求を示します。

```
http://<server>/pkmstempsession?url=<requested_resource>
```

要求に **url** 引数が含まれていない場合、WebSEAL は、**temp-cache-response** 構成エントリーで指定されたページを返します。

Microsoft Office アプリケーションとの共用セッションの構成

WebSEAL は、一時セッション・キャッシュを使用するように構成し、Internet Explorer と Microsoft Office 製品の両方が同じユーザー・セッションを参照できるようにすることができます。

/pkmstempsession 管理ページに対する要求により、一時セッション Cookie が作成されます。Internet Explorer と Microsoft Office 製品の両方がこのセッション Cookie にアクセスできます。

Ajax 要求を使用して、この管理ページを自動的に要求し、持続セッション Cookie の作成をトリガーできます。このプロセスにより、Internet Explorer を介してアクセスされるアプリケーションとのセッション共用を実現できます。

Microsoft SharePoint 2007 サーバー

Microsoft SharePoint 2007 サーバーの JavaScript を変更して、SharePoint リソースにアクセスするときに Internet Explorer と Microsoft Office が同じセッションを共用できるようにすることができます。

このタスクについて

Microsoft SharePoint サーバーとセッションを共用するには、セッションを共用するための持続セッション Cookie をいつ作成するかを SharePoint 管理者が決定しなければなりません。目的は、要求されたリソースのコンテキスト切り替えの直前に /pkmstempsession 管理ページを要求することです。

SharePoint がこのコンテキスト切り替えより前に WebSEAL に通知を送信することはありません。しかし、SharePoint サーバーにカスタム JavaScript ファイルを作成して、要求されたリソースにアクセスする前に Ajax 要求を WebSEAL に自動的に送信させることができます。この Ajax HTTP 要求で、一時セッション・キャッシュのセッション Cookie を収集できます。core.js の代わりにこのカスタム JavaScript ファイルを使用するように SharePoint を構成する必要があります。

注: SharePoint サーバーの core.js ファイルを直接更新してはなりません。このファイルを直接更新することは、SharePoint の変更としてサポートされているものではありません。代わりに、カスタム JavaScript ファイルおよびカスタム・マスター・ページを作成して core.js の必要な機能をオーバーライドしてください。カスタム・マスター・ページは、更新した JavaScript 関数をサイト収集で使用するよう構成するために使用します。

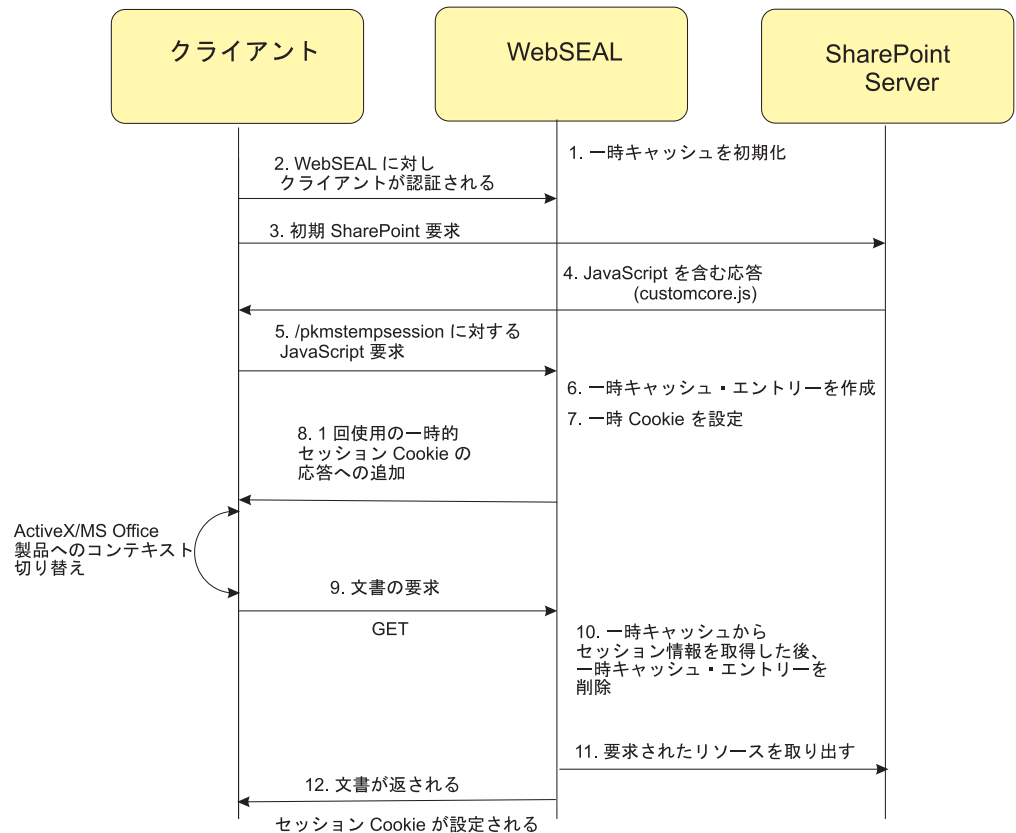


図 22. Microsoft SharePoint サーバーとの WebSEAL セッションの共用

図 22 に、一時セッション・キャッシュの操作シーケンスを示します。

1. WebSEAL は、開始すると一時キャッシュを初期化します。
2. クライアントが WebSEAL サーバーに認証されます。
3. クライアントが SharePoint リソースの要求を送信します。
4. SharePoint が、JavaScript (customcore.js) を含む応答をクライアントに返信します。
5. SharePoint リソース要求を送信する前に、更新された JavaScript が /pkmstempsession の Ajax HTTP 要求を送信します。
6. WebSEAL が、一時キャッシュにセッションのエントリーを追加します。
7. WebSEAL が、1 回使用の短時間 Cookie にセッション情報を設定します。
8. WebSEAL が、クライアント・ブラウザに返す応答にこの 1 回使用の Cookie を追加します。
9. クライアントがコンテキストを MS Office/ActiveX に切り替え、アプリケーションが GET メソッドを使用して SharePoint 文書要求を送信します。
10. WebSEAL が一時キャッシュからセッション情報を取得した後、そのエントリーを削除して、Cookie を再度使用できないようにします。
11. WebSEAL が、要求されたリソースを SharePoint サーバーから取り出します。
12. WebSEAL が、セッション Cookie を設定して文書をクライアントに返します。

手順

1. `core.js` ファイルをコピーします。`core.js` ファイルを含むフォルダーにこのコピーを貼り付け、`customcore.js` に名前変更します。

デフォルトの SharePoint 2007 インストール済み環境では、`core.js` ファイルは以下の場所にあります。

```
C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\LAYOUTS\core.js
```

2. カスタム JavaScript ファイル (`customcore.js`) を編集用に開きます。
3. `/pkmstempsession` ページに対するバンド外 Ajax HTTP 要求を実行するようにこの JavaScript を変更する必要があります。ActiveX コントロールを使用して、Office 文書を開くすべてのリンクに以下の JavaScript を追加します。特に、SharePoint サーバー上の `customcore.js` ファイルで以下の関数が始まる場所に新しい JavaScript を追加してください。

- `function DispEx(...)`
- `function createNewDocumentWithProgIDCore(...)`

注: それ以外の JavaScript 関数をすべて `customcore.js` から削除し、オーバーライドしようとする関数のみが残るようにします。不要な関数を削除すると、処理の効率が向上します。

JavaScript の例

```
var cookieRequest = false;
try {
    cookieRequest = new XMLHttpRequest();
}
catch (trymicrosoft) {
    try {
        cookieRequest = new ActiveXObject("Msxml2.XMLHTTP");
    } catch (othermicrosoft) {
        try {
            cookieRequest = new ActiveXObject("Microsoft.XMLHTTP");
        } catch (failed) {
            cookieRequest = false;
        }
    }
}
if (cookieRequest) {
    var url = "/pkmstempsession";
    cookieRequest.open("GET", url, false);
    cookieRequest.send(null);
    if (cookieRequest.status != 200){
        alert("ERROR: Single-Signon Cookie Request Failed!,Application may not load Document");
    }
}
```

4. SharePoint サイトのマスター・ページとして `default.master` ページを使用する場合は、`default.master` ページのコピーを作成します。このコピーを `custom.master` に名前変更します。

デフォルトの SharePoint 2007 インストール済み環境では、`default.master` ファイルは以下の場所にあります。

```
C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\GLOBAL\default.master
```

5. custom.master ページに、customcore.js ファイルを使用するための行を追加します。以下のように、この新しいエントリーは、core.js を参照する既存のエントリーの直後に挿入してください。

```
<SharePoint:ScriptLink language="javascript" name="core.js" Defer="true"
  runat="server"/>
<SharePoint:ScriptLink language="javascript" name="customcore.js" Defer="true"
  runat="server"/>
```

注: 元の core.js ファイルに対する参照は、このカスタム・マスター・ページに残しておく必要があります。Defer="true" プロパティを設定して、元の core.js ファイルをオーバーライドする必要があります。

6. custom.master ページを保存し、サイトのマスター・ページ・ギャラリーにアップロードします。
7. 以下のようにしてサイトのデフォルトのマスター・ページとして custom.master ページを設定します。
 - a. SharePoint のサイトにログインします。
 - b. 「サイトの操作」 > 「サイトの設定」 > 「すべてのサイト設定の変更」 > 「外観」 > 「マスター ページ」にアクセスします。
 - c. 「このサイトおよびこのサイトを継承するすべてのサイトで使用するマスター ページを指定する」を選択します。
 - d. ドロップダウン・リストから **custom.master** を選択します。

タスクの結果

Defer="true" の設定により、SharePoint 機能のロード順序が指定されます。

Defer="true" を指定した 2 つのエントリーを組み込むことで、実質的にそれらの JavaScript ファイルがマージされます。SharePoint は延期されたスクリプトを順に実行するため、customcore.js ファイルに対応するエントリーは、core.js ファイルに対応するエントリーの後にマスター・ページのソースに組み込まれます。

SharePoint は最初に core.js を実行し、次にオーバーライド用の customcore.js スクリプトを実行します。

注: カスタム JavaScript 関数を使用して core.js をオーバーライドする場合は、以下の考慮事項に注意してください。

- 処理中に送信されるスクリプトの量を最小限に抑えるために、カスタマイズするメソッドのみを customcore.js ファイルに残してください。
- SharePoint 製品をアップグレードしたりパッチをインストールしたりする場合は、更新された core.js に基づいてカスタム・スクリプトを検討し、オーバーライド用のスクリプトを引き続き適用できることを確認してください。

第 5 部 Session Management Server

第 18 章 Session Management Server (SMS) の概要

この章では、クラスター化されたサーバー環境でセッション状態を維持するための Session Management Server (SMS) ソリューションについての概要を提供します。

この章で扱うトピックは以下のとおりです。

- 『フェイルオーバー環境』
- 350 ページの『Session Management Server (SMS)』
- 351 ページの『サーバー・クラスター、レプリカ・セット、およびセッション・レルム』
- 351 ページの『SMS プロセス・フロー』
- 353 ページの『複数の DNS ドメインでのセッション共有』

フェイルオーバー Cookie および Session Management Server (SMS) は、クラスター化されたサーバー環境でセッション状態を維持するという同じ目的を果たすための代替ソリューションです。303 ページの『第 16 章 フェイルオーバー・ソリューション』も参照してください。

Session Management Server の管理上の考慮事項および操作の説明については、『*IBM Security Access Manager for Web 共用セッション管理 管理ガイド*」(SC23-6509-02) を参照してください。

フェイルオーバー環境

Session Management Server ソリューションは、クライアント要求を、2 つ以上の WebSEAL 複製サーバーにロード・バランシング・メカニズムによって分散させるシナリオで、最も一般的に使用されます。

複製されたサーバーは同一です。複製されたサーバーには、WebSEAL 保護オブジェクト・スペース、junction データベース、および dynurl データベース (オプション) のレプリカ・コピーが入っています。

クライアントは、複製されたフロントエンド・サーバーの構成を認識しません。ロード・バランシング・メカニズムは、要求されたリソースの単一 POC です。ロード・バランサーは、使用可能なサーバーにクライアントを接続します。

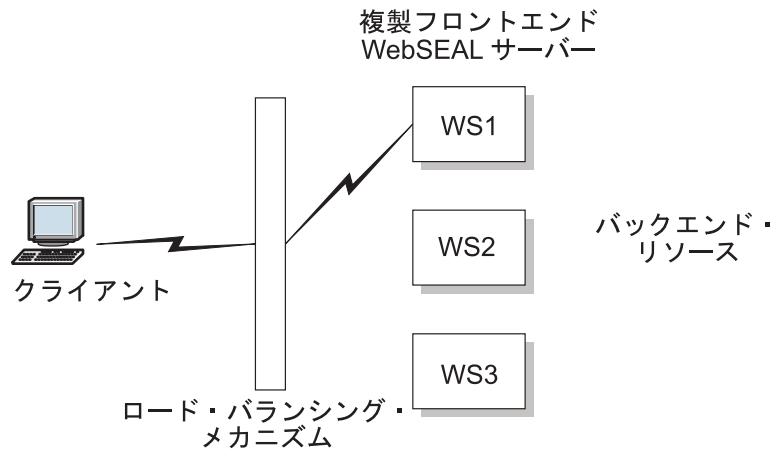


図 23. 複製 WebSEAL サーバーのフェイルオーバー

クライアントが接続されているサーバーが突然使用不可になった場合、ロード・バランサーは要求を他の複製サーバーの 1 つにリダイレクトします。このアクションは、元のセッション対資格情報のマッピングが失われる原因となります。クライアントは、この代替サーバーにとって新規であるため、通常は再度ログインする必要があります。

Session Management Server ソリューションの目的は、クライアントとのセッションがもともとあった WebSEAL サーバーが急に使用できなくなった場合の強制ログインを回避することです。Session Management Server ソリューションにより、クライアントは別の WebSEAL サーバーに接続し、同じユーザー・セッション・データと同じユーザー資格情報を持った認証セッションを作成できます。

注: 可能な場合は、ロード・バランサーでセッション・アフィニティーを維持するように構成します。セッション・アフィニティーによって、パフォーマンスが改善され、ユーザー・エクスペリエンスが向上し、WebSEAL 構成が簡単になります。

Session Management Server (SMS)

Session Management Server (SMS) は、クラスター化された WebSEAL サーバー環境の集中セッション・リポジトリとして動作する独立サービスです。

SMS の主要機能は、分散セッション・キャッシュとして動作することです。

共用セッション管理には、以下のような利点があります。

- クラスター化された Web セキュリティー・サーバー全体のセッションを管理する分散セッション・キャッシュを提供する。
- ログイン・ヒストリー情報を維持する中央ポイントを提供する。
- 複製 Web セキュリティー・サーバー環境で発生する、セッションの非アクティブ・タイムアウトおよび存続時間タイムアウトによる整合性の問題を解決する。
- 複製 Web セキュリティー・サーバー間に、セキュアなフェイルオーバーとシングル・サインオンを提供する。
- ユーザー 1 人あたりに許可されている並行セッションの最大数をコントロールできる。

- 同一 DNS ドメイン内にある他の Web サイト間のシングル・サインオンの機能を提供する。
- ハードウェアまたはソフトウェアの障害発生時に、サーバー環境のパフォーマンスと高可用性を保護する。
- アドミニストレーターが WebSEAL サーバーでセッションを表示および修正できる。

サーバー・クラスター、レプリカ・セット、およびセッション・レルム

サーバー・クラスターには、次の 2 種類があります。

- 複数のサーバーが完全に同じコンテンツ (Web サイト) をユーザーに提示する。

Session Management Server (SMS) のメイン・ユーザーは、レプリカ・セットというグループに組織化された複製 Web セキュリティー・サーバーです。レプリカ・セットは、同一の構成と保護 Web スペースを持つ複数のサーバーで構成されているため、レプリカ・セットのあるメンバーによって作成されたクライアント・セッションを別のメンバーが修正せずに使用することができます。

レプリカ・セットは、ロード・バランシングや高可用性など、パフォーマンス上の利点を提供することができます。

- 複数のサーバーが、同じではないが関連するコンテンツをユーザーに提示する。

これらの Web サイトでは、同一のコンテンツは表示されませんが、通常サーバー間でシングル・サインオン要件を設けて Security Access Manager ユーザー・レジストリーおよびポリシー・サーバーを共有します。

レプリカ・セットのグループは、セッション・レルムと呼ばれます。最大並行セッション・ポリシーや、資格情報の変更に影響を与えるポリシーなど、特定のポリシーはセッション・レルム全体に一貫して適用できます。ユーザーおよびアドミニストレーターの立場で見ると、複数のセッションが、セッション・レルム全体で 1 つのエンティティーとして存在しています。

SMS プロセス・フロー

次の図は、WebSEAL が Session Management Server (SMS) を使用するように構成された環境での、セッション管理の基本的なプロセス・フローを示しています。この例には、以下の条件が含まれています。

- WebSEAL 1 と WebSEAL 2 は、レプリカの仮想ホスト (vhostA) で構成される。
- レプリカの仮想ホストは、レプリカ・セットに属する。

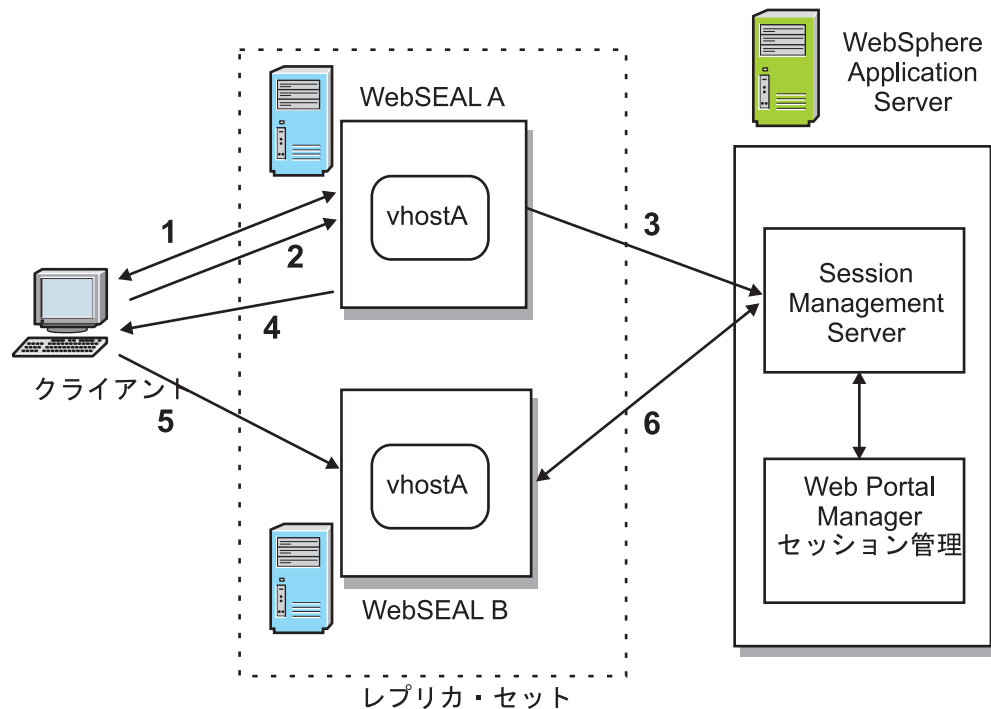


図 24. WebSEAL/SMS のプロセス・フロー

1. ユーザーは **vhostA** の Web スペースにある保護オブジェクトに対する要求を出します。WebSEAL A はこの要求をインターセプトしてユーザー用のローカル・キャッシュ・エントリを作成します。WebSEAL A は、ユーザーにログインするようにプロンプトを出します。
2. ユーザーは、認証データを WebSEAL に提供します。WebSEAL は、クライアントの資格情報を使用してローカル・セッションのキャッシュ・エントリを更新します。

ローカル・セッションのキャッシュを維持することにより、その特定の WebSEAL サーバーで今後リソースの要求があった場合にパフォーマンスが向上します。

3. WebSEAL A は、新規セッションおよび関連する資格情報を SMS に通知します。SMS は、この情報を自身のデータベースに維持します。
4. WebSEAL A は、ユーザーのブラウザにセッション Cookie を送信します。
5. **vhostA** のリソースの追加要求が、同じユーザーによって、同じセッション Cookie を使用して出されると、レプリカ・セット内の別のサーバー (WebSEAL B) にフェイルオーバーされます。
6. セッション Cookie を使用して、WebSEAL B は、このユーザーが認証済みかどうか SMS に判断を求めます。SMS はユーザーのキャッシュ付き資格情報を送信して応答します。

WebSEAL B はこの資格情報を使用してユーザーを信用し、リソースへの要求を許可して先に進みます。ユーザーに再度ログインを求めるプロンプトは出されません。

複数の DNS ドメインでのセッション共有

複数ドメイン環境で操作する場合は、各種の DNS ドメインの WebSEAL サーバーにセッション ID を通信するため、異なるメカニズムを使用する必要があります。

このタスクについて

SMS セッション・キャッシュに対する索引は、ドメイン Cookie で格納および送信されます。クライアントはこの Cookie を、ソース DNS ドメイン外にあるサーバーに提供しません。

外部認証インターフェース (EAI) の拡張機能を使用すると、EAI アプリケーションから WebSEAL に、認証データでなく SMS セッション ID を提供できます。詳しくは、275 ページの『外部認証インターフェース HTTP ヘッダーのリファレンス』を参照してください。このセッション ID は、SMS セッション・キャッシュに対するセッション・インデックスに対応します。EAI アプリケーションがセッション ID を受け取るメカニズムは、EAI アプリケーションのインプリメンテーションに応じて異なります。例えば、以下を構成できます。

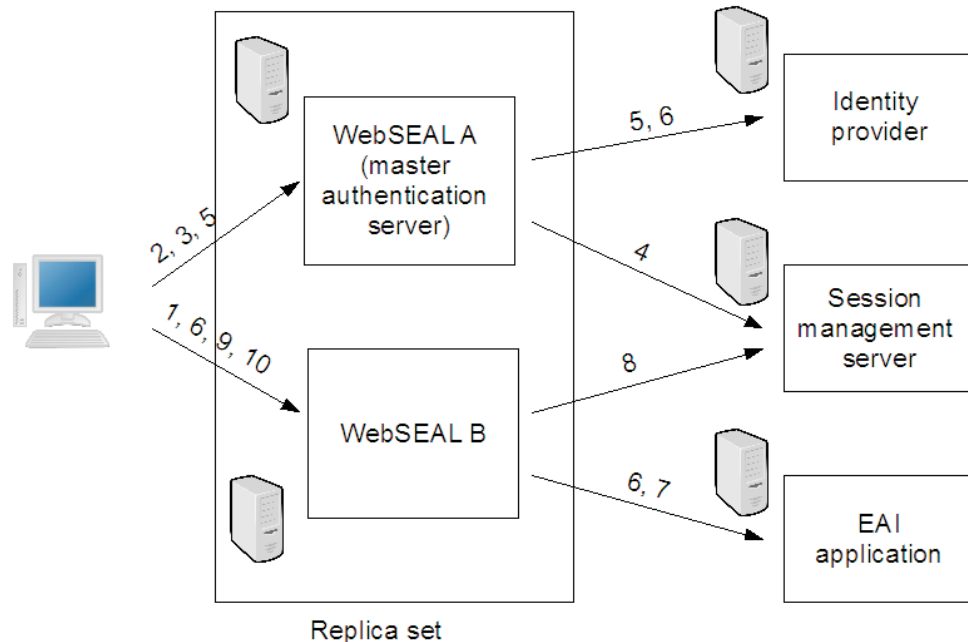
- SAML アサーションの一部として ID を提供する Tivoli Federated Identity Manager。
- セッション ID を SAML アサーションに追加するアイデンティティ・プロバイダー。サービス・プロバイダーは SAML アサーションからセッション ID を取得して、この情報を WebSEAL に返します。

このメカニズムを機能させるには、1 次認証サーバーと同じ DNS ドメイン内で、単一の WebSEAL サーバーまたは複数のロード・バランシング Web サーバーを指定します。この WebSEAL サーバーは認証されたセッションを確立します。また、WebSEAL サーバーはアイデンティティ・プロバイダーをホストし、アイデンティティ・プロバイダーは SMS セッション ID を EAI アプリケーションに返します。

SMS セッション ID をアイデンティティ・プロバイダー・アプリケーションで使えるようにするには、675 ページの『バックエンド・サーバーのユーザー・セッション管理』を参照してください。

次の図は、WebSEAL が Session Management Server を使用するように構成された複数ドメイン環境での、セッション管理の基本的なプロセス・フローを示しています。この例は、以下の条件に基づいています。

- WebSEAL A は 1 次認証サーバーとして構成されています。
- WebSEAL A および B は異なる DNS ドメインに配置されています。
- いずれの WebSEAL サーバーでもセッションは確立されません。



手順

1. ユーザーは WebSEAL B の Web スペースにある保護オブジェクトに対する要求を出します。WebSEAL B は以下の処理を実行します。
 - 要求をインターセプトします。
 - ユーザーのローカル・キャッシュ・エントリーを作成します。
 - 1 次認証サーバーの背後にあるアイデンティティ・プロバイダー・アプリケーションにリダイレクトを返信します。このリダイレクトは、ログイン・フォーム内のメタヘッダーとして設定したり、local-response-redirect 機能を使用して指定したりすることができます。
2. アイデンティティ・プロバイダーは、認証が必要な保護されたアプリケーションです。WebSEAL A:
 - 要求をインターセプトします。
 - ユーザーのローカル・キャッシュ・エントリーを作成して、その中に要求されたアイデンティティ・プロバイダーの URL を格納します。
 - ユーザーにログインするようプロンプトを出します。
3. ユーザーは WebSEAL に認証データを提供し、WebSEAL はクライアントの資格情報を使用してローカル認証を実行し、ローカル・セッション・キャッシュ・エントリーを更新します。
4. WebSEAL A は、新規セッションおよび関連する資格情報を Session Management Server に通知します。Session Management Server は自身のデータベースにこの情報を維持します。
5. WebSEAL A は以下を送信します。
 - セッション Cookie (ユーザーのブラウザーに送信)
 - リダイレクト (キャッシュ内のアイデンティティ・プロバイダーの URL に送信)

6. アイデンティティ・プロバイダーは以下の処理を実行します。
 - HTTP 要求ヘッダーから SMS セッション ID を抽出します。
 - WebSEAL B の背後で junction された EAI アプリケーションにリダイレクトを送信します。EAI アプリケーションにセッション ID を送信するためのメカニズムは、インプリメンテーション固有のものです。
7. EAI アプリケーションは以下の処理を実行します。
 - アイデンティティ・プロバイダーから提供された SMS セッション ID の場所を検出します。
 - この ID を、適切な HTTP ヘッダーで WebSEAL B に返します。
8. WebSEAL B は対応するセッションを Session Management Server から取得します。
9. WebSEAL B は以下を送信します。
 - セッション Cookie (ユーザーのブラウザーに送信)
 - リダイレクト (キャッシュ内の元の URL に送信)
10. WebSEAL B:
 - ステップ 9 で設定したセッション Cookie を使用して、適切なローカル・セッションおよび資格情報を検出します。
 - リソースに関する要求の処理を許可します。
 - ユーザーに再ログインを求めるプロンプトは出しません。

第 19 章 SMS を使用した WebSEAL のクイック・スタート・ガイド

この章では、Session Management Server (SMS) を使用するように WebSEAL を構成するための要件の要約を示します。この章の目的は、アドミニストレーターが単純な WebSEAL および SMS の統合のシナリオをセットアップし、環境を検証し、SMS が提供する機能をより理解する方法を説明することです。363 ページの『第 20 章 SMS を使用した WebSEAL の構成』に、SMS 構成オプションのより詳しい説明があります。

この章で扱うトピックは以下のとおりです。

- 『SMS を使用した WebSEAL の構成の要約』

SMS を使用した WebSEAL の構成の要約

このセクションでは、読者が Session Management Server に関連する概念を十分に理解していることを前提としています。Session Management Server の概念について詳しくは、以下を参照してください。

- 349 ページの『第 18 章 Session Management Server (SMS) の概要』。
- *IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*

ご使用の環境に合わせてこの構成を調整する方法については、以下のセクションを参照してください。363 ページの『第 20 章 SMS を使用した WebSEAL の構成』

この構成の要約では、WebSEAL 環境の以下の要件を前提とします。

- WebSEAL へのフォーム認証

Session Management Server を使用した環境ではフォーム認証は必要ありません。ただし、基本認証 (BA) (デフォルトの WebSEAL 認証方式) は、セッション強制終了で使用するには適切ではありません。

- WebSEAL と、Security Access Manager 証明書を使用する Session Management Server の間の相互認証 (SSL)
- 最大並行セッション・ポリシーの施行
- エンド・ユーザーに対して、そのユーザーのアカウントのログイン・ヒストリーを表示

1. 情報の収集

Session Management Server を使用するように WebSEAL を構成するには、情報を収集する必要があります。

以下の詳細な情報が必要です。

- Session Management Server をホストする SSL Web サーバーのホスト名およびポート番号
- WebSEAL サーバーが使用するレプリカ・セット (複数可)

- 他の Security Access Manager サーバーを認証するために WebSEAL が使用する鍵データベースおよび stash ファイル

これらの値を、WebSEAL 構成ファイル内の **[ssl]**、**ssl-keyfile** および **[ssl]**、**ssl-keyfile-stash** スタンザ・エントリーからコピーします。これらのスタンザ・エントリーおよび値は、構成ファイル内で以下のように記述されています。

```
[ssl]
ssl-keyfile = default-webseald.kdb
ssl-keyfile-stash = default-webseald.sth
```

- 基本認証ヘッダーを使用して SMS に対して認証する場合に使用されるユーザーおよびパスワード (存在する場合)

2. WebSEAL 構成ファイル設定

以下の WebSEAL 構成ファイル内のスタンザおよびスタンザ・エントリーのリストは、Session Management Server を使用するように WebSEAL を構成するために指定する必要があるオプションの完全なセットを示しています。必要な部分で、これらのオプションを設定するように WebSEAL 構成ファイルを編集してください。オリジナルの WebSEAL 構成ファイル・テンプレートに基づき、オプションの一部がデフォルトで設定される場合があります。

```
[session]
logout-remove-cookie = yes
dsess-enabled = yes
prompt-for-displacement = yes
register-authentication-failures = yes
dsess-last-access-update-interval = 60
standard-junction-replica-set = replica_set_for_standard_junctions

[replica-sets]
replica-set = replica_set_for_standard_junctions
replica-set = replica_set_for_a_virtual_host_junction
... repeat for all of the replica sets in which
this WebSEAL server participates ...

[dsess]
dsess-sess-id-pool-size = 125
dsess-cluster-name = SMS cluster name

[dsess-cluster]
server = [0-9], <URL>
response-by = 60
handle-pool-size = 10
handle-idle-timeout = 240
timeout = 30
```

3. Security Access Manager CA 証明書のインポート

手順

- Security Access Manager 認証局 (CA) 証明書を、WebSEAL クライアントおよび junction 証明書データベースにインポートします。デフォルト・クライアントおよび junction 証明書鍵データベースは、WebSEAL 構成ファイル内の **[ssl]**、**webseal-cert-keyfile** スタンザ・エントリーを使用して指定されます。**webseal-cert-keyfile** スタンザ・エントリーのデフォルト値は以下のとおりです。

```
[ssl]
webseal-cert-keyfile = pdsrv.kdb
```

- あるいは、junction SSL サーバー証明書の検証に別の鍵ストアを使用するように WebSEAL を構成することもできます。オプションの junction 証明書鍵データベースは、WebSEAL 構成ファイル内の **[junction]**、**jct-cert-keyfile** スタンザ・エントリーを使用して指定されます。junction 用の別のオプション鍵データベースは、構成ファイル内ではコメント化されています。このオプションが使用可能な場合、**jct-cert-keyfile** スタンザ・エントリーのデフォルト値は以下のとおりです。

```
[junction]
jct-cert-keyfile = pdjct.kdb
```

注: ローカル管理インターフェース (LMI) を使用して、**webseal-cert-keyfile** スタンザ・エントリーで指定された鍵データベースに CA 証明書をインポートします。LMI で証明書を管理するには、「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「グローバル・キー」>「SSL 証明書」にアクセスします。

4. WebSEAL サーバーを再始動する

手順

- WebSEAL サーバーを再始動します。
- WebSEAL サーバー・ログ・ファイルで、始動プロセス中に発生したエラーまたは警告があるかどうかを確認します。一部の警告メッセージは、正常な場合にも出され、問題を示すものではありません。例:

```
WebSEAL received notification that the SMS session cache
for replica-set "replica-set" was cleared.
All local reference to sessions are being discarded to
synchronize the local session cache with the SMS session cache.
```

懸念の原因となるその他の警告またはエラーが発生した場合は、問題を調査し、解決してください。

5. 仮想ホストの junction の作成

このタスクについて

仮想ホストを使用している場合、仮想ホスト junction を作成します。

手順

- **-z** オプションを使用して、各仮想ホスト junction のレプリカ・セットの名前を指定します。

6. Session Management Server の junction

手順

1. Session Management Server への標準の WebSEAL junction または仮想ホスト junction を作成します。例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task server create -t ssl -h sms_server_name  
-p sms_server_port -D DN_of_SMS_Web_server /sms
```

2. SMS-Session-Index HTTP ヘッダーの値を拡張属性として渡すように、junction を変更します。例:

```
pdadmin> object modify /WebSEAL/server_object/sms set attribute  
HTTP-Tag-Value session_index=SMS-Session-Index
```

3. lastLogin.jsp ページへのアクセスのみを許可する ACL を付加します。例:

```
pdadmin> acl create noaccess  
pdadmin> acl modify noaccess set group iv-admin TcmdbsvaBR1r  
pdadmin> acl modify noaccess set any-other T  
pdadmin> acl modify noaccess set unauth T  
pdadmin> acl attach /WebSEAL/<server-object>/sms noaccess  
  
pdadmin> acl create authonly  
pdadmin> acl modify authonly set user sec_master TcmdbsvaBR1r  
pdadmin> acl modify authonly set group iv-admin TcmdbsvaBR1r  
pdadmin> acl modify authonly set any-other Tr  
pdadmin> acl modify authonly set unauth T  
pdadmin> acl attach /WebSEAL/server_object/sms/DSess/lastLogin.jsp authonly
```

7. 最大並行セッション・ポリシーの設定

このタスクについて

ご使用の環境の最大並行セッション・ポリシーを設定します。例:

```
pdadmin> policy set max-concurrent-web-sessions displace
```

8. 構成のテスト

このタスクについて

構成をテストします。以下の手順を実行中に問題が発生した場合は、ブラウザーまたは WebSEAL サーバー・ログ・ファイルに表示されたエラーを調べて、問題を訂正します。

手順

1. 以下のようにして、ログインが正しく機能することをテストします。

- 以下の URL にアクセスします。

```
https://webseal
```

ログインのプロンプトが出されます。

- テスト・ユーザーとしてログインします。

Web サイトへのアクセスが許可されます。

2. 最大並行セッション・ポリシーが施行されていることを確認します。

- 別のマシンから、以下の URL にアクセスします。

```
https://webseal
```

ログインのプロンプトが出されます。

- 正しいパスワードを使用して、テスト・ユーザーとしてログインします。

古いセッションを強制終了するように求めるプロンプトが出されます。

- 「既存のログインの終了」リンクをクリックします。

Web サイトへのアクセスが許可されます。

- 元のブラウザーに戻り、以下の URL に再度アクセスします。

`https://webseal`

ブラウザーが Web サイトのフロントページをキャッシュに入れた場合、シフト・キーを押し下げてページを再ロード (最新表示) しなければならない場合があります。

元のセッションが強制終了されたため、ログインのプロンプトが出されます。

3. ログイン・ヒストリーが機能していることを確認します。

- 以下の URL にアクセスして、Web サイトからログアウトします。

`https://webseal/pkmslogout`

ログアウトしたことを示すページが戻されます。

- 以下の URL にアクセスします。

`https://webseal/sms/DSess/lastLogin.jsp`

ログインのプロンプトが出されます。

- 正しくないパスワードを使用して、テスト・ユーザーとしてログインします。

ログインのプロンプトが再度出されます。

- 今度は正しいパスワードを使用して、テスト・ユーザーとしてログインします。

最後にログインした時間と、それ以降に 1 度ログインが失敗したことを示す「最終ログイン (Last login)」ページが表示されます。

4. 以下のようにして ACL が正しく付加されていることを確認します。

- 以下の URL にアクセスします。

`https://webseal/sms/DSess/services/DSess`

アクセスが禁止されていることを示すページが表示されます。

第 20 章 SMS を使用した WebSEAL の構成

この章では、WebSEAL と Session Management Server (SMS) ソリューションを統合するための構成オプションについて説明します。

この章で扱うトピックは以下のとおりです。

- 『WebSEAL の SMS の構成』
- 365 ページの『レプリカ・セット構成』
- 370 ページの『SMS の最終アクセス時間更新頻度の調整』
- 370 ページの『SMS 通信タイムアウトの構成』
- 372 ページの『SMS パフォーマンスの構成』
- 373 ページの『WebSEAL および SMS の SSL 構成』
- 375 ページの『最大並行セッション・ポリシー』
- 381 ページの『セッション・レルム内でのシングル・サインオン』
- 385 ページの『ログイン・ヒストリーの構成』

WebSEAL の SMS の構成

このセクションは、以下のトピックで構成されています。

- 『Session Management Server (SMS) の構成』
- 『WebSEAL の SMS の使用可能化と使用不可化』
- 364 ページの『Session Management Server のクラスターとロケーションの指定』
- 365 ページの『最大並行セッション・ポリシー値の検索』

Session Management Server (SMS) の構成

このタスクについて

Session Management Server (SMS) をインストールし、クラスター化されたサーバー環境に合わせて構成するには、「*IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*」を参照してください。

WebSEAL の SMS の使用可能化と使用不可化

このタスクについて

WebSEAL 構成ファイルの `[session]` スタンザ内の `dsess-enabled` スタンザ・エントリを使用して、WebSEAL で Session Management Server (SMS) を使用可能または使用不可に設定します。

注: 句「dsess」は、「分散セッション」を意味します。

クラスター化された WebSEAL サーバー環境で SMS を使用可能にする場合、セッションの Cookie を使用して分散セッション状態の情報を維持します。SMS の使用

中は、**[session]** の **ssl-id-sessions** スタンザ・エントリーは適用されません。

手順

- ユーザー・セッションを維持するために、WebSEAL を使用可能にして Session Management Server (SMS) を使用するには、値「yes」を入力します。例:

```
[session]
dsess-enabled = yes
```

- ユーザー・セッションを維持するための Session Management Server (SMS) の WebSEAL による使用を不可にするには、値「no」(デフォルト)を入力します。例:

```
[session]
dsess-enabled = no
```

Session Management Server のクラスターとロケーションの指定

このタスクについて

SMS を使用した構成エントリーは、WebSEAL 構成ファイルの **[dsess]** スタンザおよび **[dsess-cluster]** スタンザ内にあります。

手順

- Session Management Server のロケーションを指定するには、**[dsess]** スタンザの **dsess-cluster-name** エントリー内でクラスター名を定義します。例:

```
[dsess]
dsess-cluster-name = dsess
```

- 次に、対応する **[dsess-cluster:<cluster-name>]** スタンザ内でクラスターの詳細を定義します。 **server** エントリーを使用して、Session Management Server のロケーション (URL) を指定します。例:

```
[dsess-cluster:dsess]
server = http://sms.example.com/DSess/services/DSess
```

注: SMS サーバーのクラスターを定義するためのデフォルト・パラメーターおよびデフォルト値は、**[dsess-cluster]** スタンザ内で指定します。

フェイルオーバー構成に複数の SMS がインストールされているアーキテクチャーでは、この構成エントリーのインスタンスを複数作成する必要があります。フェイルオーバーおよびロード・バランシング用に複数のサーバー・エントリーを指定することができます。これらのサーバー・エントリーの完全なセットによって、SMS クラスターのメンバーシップが定義されます。

- オプションとして、URL の前に 1 から 9 の数字を含めることによって、各サーバーの優先順位を指定することができます。この数字が存在する場合、数字はクラスター内のサーバーの優先順位を表します (9 が最大、0 が最小)。優先順位が指定されていない場合、優先順位 9 であるものとします。 **server** エントリーで URL に HTTPS プロトコルを指定する場合は、SMS と SSL 通信を行うように WebSEAL を構成しなければなりません。373 ページの『WebSEAL および SMS の SSL 構成』を参照してください。

最大並行セッション・ポリシー値の検索

このタスクについて

最大並行セッション・ポリシー (`pdadmin policy set max-concurrent-web-sessions`) を使用すると、Session Management Server (SMS) で管理される分散セッション環境内で、各ユーザーが一度に接続できるセッション数をコントロールできます。デフォルトでは、このポリシーは使用可能にされています。

```
[session]
enforce-max-sessions-policy = yes
```

アドミニストレーターは、このポリシーを特定のユーザーに対して適用すること、このセキュア・ドメインに登録済みのすべてのユーザーに対してグローバルに適用することもできます。ポリシーは Security Access Manager ユーザー・レジストリーに保管されます。ポリシーを SMS 環境の認証プロセスで実行するには、ポリシーをレジストリーから検索して、各ユーザーの資格情報に拡張属性として保管する必要があります。

レプリカ・セット構成

レプリカ・セットは、同一の構成および保護 Web スペースを持つ複数のサーバーで構成されています。レプリカ・セットのあるメンバーによって作成されたクライアント・セッションは、別のメンバーが修正せずに使用することができます。

サーバー環境で使用するレプリカ・セットごとに名前を指定する必要があります。慣例により、レプリカ・セット名は Web サイトで使用される DNS 名 (完全修飾ホスト名) と一致する必要があります。例えば、Web サイトの DNS 名が `www.example.com` の場合、Web サイトのレプリカ・セット名も `www.example.com` にする必要があります。

- 最初に、各レプリカ・セット名を、Session Management Server の構成時に初期定義しなければなりません。IBM Security Access Manager for Web: 共用セッション管理 管理ガイド を参照してください。
- 次に、レプリカ・セット名を、レプリカ・セットに参加する各 WebSEAL インスタンスの構成ファイルにそれぞれ指定する必要があります。さらに、それぞれの junction または仮想ホストを該当するレプリカ・セットに割り当ててください。標準 junction および仮想ホストをレプリカ・セットに割り当てる方法はいくつかあります。以下を参照してください。
 - 『複数のレプリカ・セットに参加する WebSEAL の構成』。
 - 366 ページの『標準 junction のレプリカ・セットへの割り当て』。
 - 366 ページの『レプリカ・セットに割り当てられる仮想ホスト』。

複数のレプリカ・セットに参加する WebSEAL の構成

WebSEAL が参加する各レプリカ・セットは、WebSEAL 構成ファイルの `[replica-sets]` スタンザにリストされている必要があります。

例

WebSEAL が `vhostA.example.com`、`vhostB.example.com`、および `www.example.com` という名前のレプリカ・セットに参加する場合、スタンザは以下のように構成されます。

```
[replica-sets]
replica-set = vhostA.example.com
replica-set = vhostB.example.com
replica-set = www.example.com
```

標準 junction のレプリカ・セットへの割り当て

このタスクについて

設計によって、WebSEAL インスタンスのすべての標準 `junction` は、WebSEAL 構成ファイルの `[session]` スタンザの `standard-junction-replica-set` スタンザ・エントリーの指定どおりに 1 つのレプリカ・セットに割り当てられます。

SMS を使用するには、`standard-junction-replica-set` スタンザ・エントリーの値が `[replica-sets]` スタンザにもリストされている必要があります。`standard-junction-replica-set` の値が `[replica-sets]` スタンザに存在しない場合、WebSEAL は始動しません。

```
[session]
standard-junction-replica-set = replica-set-name
```

```
[replica-sets]
replica-set = replica-set-name
```

例

```
[session]
standard-junction-replica-set = www.example.com
```

```
[replica-sets]
replica-set = www.example.com
```

レプリカ・セットに割り当てられる仮想ホスト

標準 `junction` とは対照的に、仮想ホストは個別に異なるレプリカ・セットに割り当てることができます。これには、`pdadmin server task virtualhost create` コマンドの `-z` オプションを指定します。

SMS がオペレーション中の場合、`-z` オプションによって、仮想ホスト `junction` 上のセッションを管理するレプリカ・セットを指定します。SMS を使用するように WebSEAL を構成する場合、仮想ホスト `junction` のレプリカ・セットは、`-z` オプションを使用して指定できます。レプリカ・セットごとに異なる仮想ホスト `junction` を割り当てる必要があります。`-z` オプションを使用しない場合、WebSEAL はレプリカ・セットの名前として、`junction` の仮想ホスト名を使用します。

また、この仮想ホストで使用するレプリカ・セットの名前は、以下のように、WebSEAL インスタンスの構成ファイルの `[replica-sets]` スタンザ内の `replica-set` スタンザ・エントリーによって定義しなければなりません。

```
[replica-sets]
replica-set = replica-set-name
```

条件の例:

- 仮想ホスト・タイプ: TCP
- 仮想ホストが常駐するマシンのホスト名: abc.example.com
- 仮想ホスト名: vh1.example.com
- このレプリカ・セットに属する仮想ホスト: vh1.example.com
- 仮想ホスト junction ラベル: vhost-vh1-http

仮想ホスト作成コマンドの例 (1 行で入力します):

```
pdadmin> server task default-webseald-webseal.example.com virtualhost create  
-t tcp -h abc.example.com -v vh1.example.com -z vh1.example.com vhost-vh1-http
```

レプリカ・セットの WebSEAL 構成例:

```
[replica-sets]  
replica-set = vh1.example.com
```

レプリカ・セット構成の例

以下に、レプリカ・セット構成の例を示します。

ネットワーク体系:

- 下記のように、3 つの WebSEAL サーバーがあって、それぞれ 1 つの WebSEAL インスタンス (「デフォルト」) が構成されているとします。
 - WebSEAL-A: webseal-A.example.com
 - WebSEAL-B: webseal-B.example.com
 - WebSEAL-C: webseal-C.example.com
- WebSEAL インスタンスを、2 つの仮想ホストと 1 つの標準 junction にそれぞれ構成します。
- 仮想ホストおよび標準 junction リソースは、以下のように、レプリカのバックエンド・サーバー上に常駐します。
 - 仮想ホスト vhostA.example.com は、バックエンド・サーバー a.example.com に常駐
 - 仮想ホスト vhostB.example.com は、バックエンド・サーバー b.example.com に常駐
 - 標準 junction /jctC は、バックエンド・サーバー c.example.com に常駐
- Junction の指定:
 - 仮想ホスト・ラベル **vhost-A** は、仮想ホスト vhostA.example.com の junction を表す
 - 仮想ホスト・ラベル **vhost-B** は、仮想ホスト vhostB.example.com の junction を表す
 - Junction /jctC は、バックエンド・サーバー c.example.com の標準 junction
- WebSEAL サーバーおよび複製バックエンド・サーバー全体で、スター型トポロジを使用します。

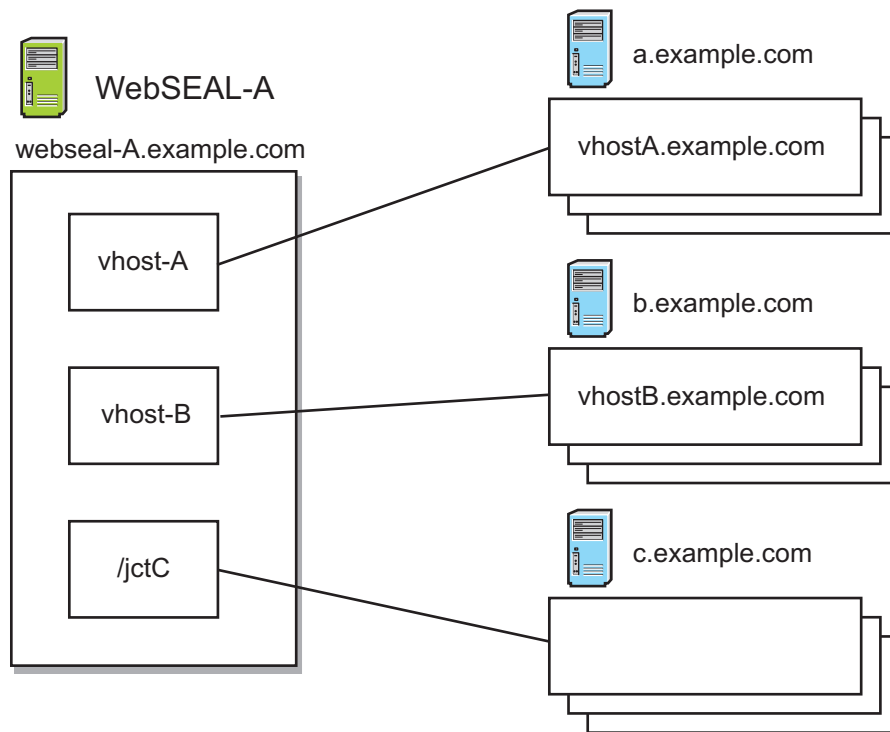


図 25. 単一 WebSEAL サーバーの Junction 構成

junction 構成:

- SMS 構成には、vhostA.example.com、vhostB.example.com、および www.example.com の 3 つのレプリカ・セットが含まれます。
- WebSEAL 構成ファイルには、この 3 つのレプリカ・セットがすべて **[replica-sets]** スタンザにリストされます。
- **vhost-A** junctions は、レプリカ・セット vhostA.example.com に、**-z** 仮想ホスト junction オプションを使用して割り当てられます。
- **vhost-B** junctions は、レプリカ・セット vhostB.example.com に、**-z** 仮想ホスト junction オプションを使用して割り当てられます。
- 標準 junction (/ および **/jctC** junctions を含む) は、WebSEAL 構成ファイルの **[session]**、**standard-junction-replica-set** スタンザ・エントリを使用して、レプリカ・セット www.example.com に割り当てられます。

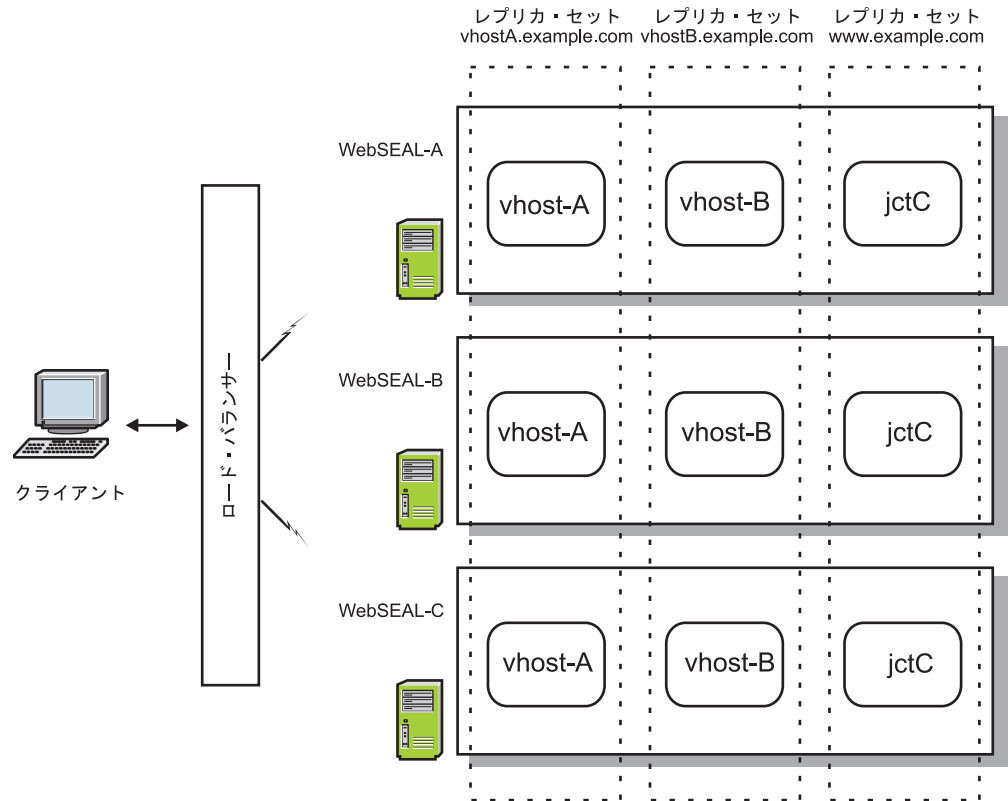


図 26. レプリカ・セット構成

WebSEAL-A (webseal-A.example.com) の構成例:

- レプリカ・セット構成:

```
[session]
standard-junction-replica-set = www.example.com
```

```
[replica-sets]
replica-set = www.example.com
replica-set = vhostB.example.com
replica-set = vhostA.example.com
```

- バックエンド・サーバー a.example.com に配置されている仮想ホスト vhostA.example.com の仮想ホスト junction (ラベル: **vhost-A**) を次のように作成します (1 行に入力)。

```
pdadmin> server task default-webseald-webseal1.example.com virtualhost create
-t tcp -h a.example.com -v vhostA.example.com -z vhostA.example.com vhost-A
```

- バックエンド・サーバー b.example.com に配置されている仮想ホスト vhostB.example.com の仮想ホスト junction (ラベル: **vhost-B**) を次のように作成します (1 行に入力)。

```
pdadmin> server task default-webseald-webseal1.example.com virtualhost create
-t tcp -h b.example.com -v vhostB.example.com -z vhostB.example.com vhost-B
```

- バックエンド・サーバー c.example.com の標準 junction **jctC** を次のように作成します (1 行に入力)。

```
pdadmin> server task default-webseald-webseal1.example.com create
-t tcp -h c.example.com /jctC
```

WebSEAL-B および WebSEAL-C の構成も同様に行います。

SMS の最終アクセス時間更新頻度の調整

WebSEAL 構成ファイルの **[session]** セクション内の **dsess-last-access-update-interval** スタンザ・エントリーを使用して、WebSEAL が SMS での最新アクセス時にセッションを更新する頻度を指定します。Session Management Server (SMS) を使用しているときに、セッションの非アクティブ・タイムアウトを調整したり、セッションの非アクティブ・ポリシー (**reauth-for-inactive = yes**) に基づいて再認証を構成する場合は、この値を調整する必要があります。

値を小さくすると、更新が SMS へより頻繁に送信されることになりますが、より正確な非アクティブ・タイムアウト・トラッキングが可能になります。1 秒未満の値は指定できません。デフォルト値は 60 秒です。例:

```
[session]
dsess-last-access-update-interval = 60
```

例として、次の構成について検討してください。

```
[session]
inactive-timeout = 600
dsess-last-access-update-interval = 60
```

これらの構成値では、WebSEAL サーバーへのユーザーの最終アクセスの後、540 秒から 600 秒までの間に、ユーザーのセッションに「非アクティブ」のフラグが、SMS で立てられる可能性があります。

dsess-last-access-update-interval パラメーターに小さい値を設定すると、WebSEAL サーバーのパフォーマンスに深刻な影響が生じる可能性があるためお勧めできません。

278 ページの『外部認証インターフェースによる再認証』も参照してください。

299 ページの『キャッシュ・エントリーの非アクティブ・タイムアウト値』も参照してください。

SMS 通信タイムアウトの構成

このセクションは、以下のトピックで構成されています。

- 『SMS 応答タイムアウトの構成』
- 371 ページの『ブロードキャスト・イベントの接続タイムアウトの構成』

SMS 応答タイムアウトの構成

このタスクについて

WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **timeout** スタンザ・エントリーを使用すると、Session Management Server からの応答に対して WebSEAL が待機可能な時間 (秒数) を指定できます。デフォルト値は 30 秒です。例:

```
[dsess-cluster]
timeout = 30
```

タイムアウトの限度時間が経過しても SMS からの応答がない場合、WebSEAL は SMS が使用不可であると想定します。この場合、以下のアクションが取られます。

- SMS がリカバリーしたかまたはバックアップがオンラインになったかを確認するために個別の WebSEAL サーバー・スレッドが 60 秒ごとに SMS に接続を試みます。
- WebSEAL サーバー上でのセッション作成またはセッションへのアクセスのすべての試みで、WebSEAL サーバーから HTTP の「503 サービスが使用できません」エラー・ページ (38b9a4b1.html) を受信します。このページは、86 ページの『HTML サーバー応答ページの変更』で説明したように、エラー状況「0x38b9a4b1」のエラー・ページを作成するとカスタマイズできます。

SMS がリカバリーすると、WebSEAL は、停止が一時的なネットワーク障害によるものか、また、SMS サーバーが再始動したかどうかを判別しようとします。SMS サーバーが再始動していた場合は、ローカルの WebSEAL セッション・キャッシュは消去されます。WebSEAL サーバー上のすべてのセッションは削除されます。このアクションによって、クラスター内のすべての WebSEAL サーバーと SMS のセッション全体で同期化が維持されます。SMS の高可用性構成の複製について詳しくは、「*IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*」を参照してください。

ブロードキャスト・イベントの接続タイムアウトの構成

このタスクについて

クラスター化された一部のサーバー・アーキテクチャーでは、WebSEAL サーバーと SMS を実行中の WebSphere のインストールとの間にファイアウォールをインプリメントする場合があります。ファイアウォールは通信の流れを一方向に制限することがよくあります。WebSEAL は、ファイアウォールを介してセッション情報を SMS に送信します。SMS からブロードキャスト・イベントを追加で受信するには、WebSEAL はファイアウォールを介した別の接続をオープンする必要があります。ファイアウォールのタイムアウト・ポリシーによって、WebSEAL が SMS からのブロードキャスト・イベントを待機している間にこの接続がシャットダウンされる場合があるからです。

手順

WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **response-by** スタンザ・エントリーを使用すると、WebSEAL が SMS クラスターのブロードキャスト・イベントを受信するために SMS との接続をオープンし続ける時間 (秒数) を指定できます。タイムアウト値に到達すると、WebSEAL は新規の接続を再作成します。

```
[dsess-cluster]
response-by = 60
```

この接続をオープンし続ける条件を最適にするために、**response-by** スタンザ・エントリー値は、ファイアウォール内部のタイムアウト値より小さく設定してください。

SMS パフォーマンスの構成

このセクションは、以下のトピックで構成されています。

- 『事前割り振りセッション ID の最大数』
- 『ハンドル・プール・サイズの構成』

事前割り振りセッション ID の最大数

WebSEAL 構成ファイルの **[dsess]** スタンザ内の **dsess-sess-id-pool-size** スタンザ・エントリーは、セッション ID プールに事前割り振りされるセッション ID の最大数を指定します。このスタンザ・エントリーは SMS クラスタで使用され、SMS が使用可能な場合は必須です。

```
[dsess]
dsess-sess-id-pool-size = 125
```

ハンドル・プール・サイズの構成

WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **handle-pool-size** スタンザ・エントリーは、dsess クライアントでいつでも維持される、アイドル状態の Simple Access Object Protocol (SOAP) ハンドルの最大数を指定します。このハンドルは、予想される SMS との交換で使用されます。デフォルト値は 10 です。

例:

```
[dsess-cluster]
handle-pool-size = 10
```

ほとんどの環境では、このデフォルト値を変更する必要はありません。

すべてのハンドルが使用中のために WebSEAL と SMS 間の通信が低速の場合は、この値を増やすことができます。この値を変更する場合は、WebSEAL サーバーに予約されている各ハンドルのファイル記述子に注意してください。値を高くすると、ファイル記述子を必要とする他の WebSEAL の機能の正常な動作を妨げる場合があります。

handle-pool-size の最大値は、WebSEAL サーバーを実行するプラットフォームの種類と、多様な WebSEAL の構成オプションの両方に応じて異なります。ほとんどの環境では、**handle-pool-size** は 25 ハンドルを超えないようにしてください。

SMS 認証

SMS に基本認証ヘッダーを提供するように WebSEAL を構成できます。

BA ヘッダーは **[dsess-cluster]** スタンザ内の **basic-auth-user** 構成エントリーおよび **basic-auth-passwd** 構成エントリーで定義されたユーザー名とパスワードで構成されています。

WebSEAL および SMS の SSL 構成

[dsess-cluster] server スタンザ・エントリーで URL に HTTPS プロトコルを指定する場合は、SMS と SSL 通信を行うように WebSEAL を構成しなければなりません。WebSEAL は、クライアント証明書を持つ SMS を認証することができます。

SMS と SSL 通信を行う WebSEAL の構成では、以下の情報を WebSEAL に提供する必要があります。

- SMS の SSL サーバー証明書に署名するために使用する CA 証明書。
- SMS の SSL サーバー証明書に含まれる DN。
- WebSEAL が SMS の認証で使用する必要がある SSL クライアント証明書。

SMS を使用する SSL 接続の初期化時に使用する、追加の GSKit 属性を構成することもできます。

このセクションは、以下のトピックで構成されています。

- 『WebSEAL 鍵データベースの構成』
- 374 ページの『SSL 証明書の識別名 (DN) の指定』
- 375 ページの『SMS 接続用の GSKit 構成』

WebSEAL 鍵データベースの構成

このタスクについて

WebSEAL は、SMS との SSL 通信で使用する、以下のクライアント・サイドの証明書と CA ルート証明書を、鍵データベース・ファイルに保管します。

- CA ルート証明書は、SMS から返されるサーバー証明書を検証するために使用します。
- クライアント・サイドの証明書は、SMS を相互認証用に構成する場合に WebSEAL で使用します。

手順

- 鍵データベース・ファイルを指定するには、WebSEAL 構成ファイルの [dsess-cluster] スタンザ内の **ssl-keyfile** スタンザ・エントリーを使用します。例:

```
[dsess-cluster]
ssl-keyfile = key-file-name
```

WebSEAL と SMS の間の通信に Security Access Manager の SSL 証明書が使用されている場合を除き、他の WebSEAL 鍵ファイルとは別の鍵ファイルを **ssl-keyfile** の値に使用してください。

- 鍵データベース stash ファイル (データベース・ファイルにアクセスするためのパスワード情報が含まれる) を指定するには、WebSEAL 構成ファイルの [dsess-cluster] スタンザ内の **ssl-keyfile-stash** スタンザ・エントリーを使用します。例:

```
[dsess-cluster]
ssl-keyfile-stash = key-file-name
```

- クライアント・サイドの証明書のラベル名を指定するには、WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **ssl-keyfile-label** スタンザ・エントリーを使用します。例:

```
[dsess-cluster]
ssl-keyfile-label = label-name
```

SSL 証明書の識別名 (DN) の指定

このタスクについて

WebSEAL 鍵データベース・ファイルに保管されている CA ルート証明書は、SMS から受け取った証明書が本物であることを検証します。さらに、証明書の DN 値をチェックすることで、WebSEAL によって受け取られた SMS からのサーバー証明書が予想通りの証明書であることを確認できます。

受け入れ済み証明書の DN 値を指定するには、WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **ssl-valid-server-dn** スタンザ・エントリーを使用します。

例

```
[dsess-cluster]
ssl-valid-server-dn = DN-value
```

サーバー証明書の DN 値の取得

このタスクについて

WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **ssl-valid-server-dn** では、WebSEAL との通信中に SMS が送信する有効なサーバー証明書で検出される DN 値が必要になります。

DN 値は、SMS アドミニストレータから直接取得できます。

あるいは、以下の手順に従って、この値を間接的に判別することもできます。

手順

1. WebSEAL 用に SMS を使用可能にする。
[session] dsess-enabled = yes
2. SMS が SSL 用に構成されていることを確認する。SMS への URL には、以下のように HTTPS プロトコルが必要です。
[dsess-cluster] server = https://server/DSess/services/DSess
3. WebSEAL 構成ファイルの **[dsess-cluster]** スタンザ内の **ssl-keyfile**、**ssl-keyfile-stash**、および **ssl-keyfile-label** の各スタンザ・エントリーの構成手順に従います。373 ページの『WebSEAL 鍵データベースの構成』を参照してください。
4. **ssl-valid-server-dn** スタンザ・エントリーにテスト値を入力します。例:
[dsess-cluster] ssl-valid-server-dn = test
5. WebSEAL サーバーを再始動します。
6. WebSEAL から次のエラー・メッセージが戻されます。

The DN contained within the server certificate, <DN>, is not a configured DN.

メッセージにリストされている DN は、SMS で提示された証明書の DN です。

この値を使用して、**ssl-valid-server-dn** スタンザ・エントリーに正しい値を指定してください。

7. 正しい SSL サーバーと通信しているかを検証するには、エラー・メッセージで戻された DN 値を SMS アドミニストレーターで確認します。

SMS サーバー証明書の DN が正しい値であることを確認したら、**ssl-valid-server-dn** スタンザ・エントリーの値にこの DN を使用します。

SMS 接続用の GSKit 構成

GSKit での SSL 接続の作成を制御するために使用できるよう、いくつかの GSKit 属性が用意されています。WebSEAL は、SSL 接続を初期化するときに特定の GSKit 属性を使用するように構成できます。

[dsess-cluster] スタンザの **gsk-attr-name** 構成エントリーにより、SMS との接続を初期化するときに WebSEAL が使用する GSKit 属性を制御します。この構成エントリーは、複数回指定することができます。必要な GSKit 属性それぞれを、新規エントリーとして含めてください。

```
[dsess-cluster]
gsk-attr-name = {enum | string | number}:id:value
```

注: [ssl] スタンザには、クライアントおよび junction Web サーバーとの接続で使用する類似の構成エントリーが存在します。

これらの構成エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

最大並行セッション・ポリシー

このセクションは、以下のトピックで構成されています。

- 『最大並行セッション・ポリシーの設定』
- 379 ページの『最大並行セッション・ポリシーの実行』
- 380 ページの『ユーザーおよび最大並行セッション・ポリシーの切り替え』

最大並行セッション・ポリシーの設定

このタスクについて

Session Management Server で管理される分散セッション環境内で、各ユーザーが一度に接続できるセッション数をコントロールできます。**pdadmin policy set max-concurrent-web-sessions** コマンドを使用して、この並行セッションの最大数を指定します。

アドミニストレータは、このポリシーを特定のユーザーに対して適用すること、このセキュア・ドメインに登録済みのすべてのユーザーに対してグローバルに適用することもできます。379 ページの『ユーザーごとの設定とグローバル設定』を参照してください。

WebSEAL 構成ファイルの [session] スタンザ内の **enforce-max-sessions-policy** スタンザ・エントリを使用して、特定の WebSEAL インスタンスで **max-concurrent-web-sessions** ポリシーを実行するかどうか制御します。379 ページの『最大並行セッション・ポリシーの実行』を参照してください。

pdadmin ポリシーのコマンド構文 (それぞれ 1 行で入力)

```
policy set max-concurrent-web-sessions {unset|number|displace|unlimited}  
[-user username]
```

```
policy get max-concurrent-web-sessions [-user username]
```

pdadmin ポリシー・セットの引数の説明:

- **未設定**

max-concurrent-web-sessions ポリシーを使用不可にします。この設定では、ポリシーに含まれる値はありません。ユーザーに対して有効なポリシーは、**unlimited** 設定と同じです。

unset 設定がデフォルト・ポリシーです。

例 (グローバル設定):

```
pdadmin> policy set max-concurrent-web-sessions unset
```

- **number**

ユーザー 1 人あたりに許容される並行セッション数を指定します。ユーザーは、この数を超えてセッションを確立することはできません。

例 (グローバル設定):

```
pdadmin> policy set max-concurrent-web-sessions 2
```

この値を超えてログインが試行されると、エラー応答ページ (38b9a41f.html 「追加ログインが拒否されました」) がユーザーに戻されます。

- **制限無し**

ユーザー 1 人あたりに無制限の並行セッション数を許可します。

例 (グローバル設定):

```
pdadmin> policy set max-concurrent-web-sessions unlimited
```

- **displace**

max-concurrent-web-sessions ポリシーのセッションの値を「1」に強制することによって、1 度に 1 つのアクティブ・セッションのみにユーザーを制限します。

例 (グローバル設定):

```
pdadmin> policy set max-concurrent-web-sessions displace
```

追加のログイン試行に対する応答は、WebSEAL 構成ファイルの **[session]** スタンザ内の **prompt-for-displacement** によって管理されます。

『対話式強制終了』および 378 ページの『非対話式強制終了』を参照してください。

対話式強制終了

WebSEAL 構成ファイルの **[session]** スタンザ内の **prompt-for-displacement** スタンザ・エントリーを使用して、**max-concurrent-web-sessions displace** ポリシーを超過した場合に、適切なアクションを促すプロンプトをユーザーに出すかどうかを決定します。このセクションでは、ユーザーに適切なアクションを促すプロンプトを出す対話式オプション (**prompt-for-displacement = yes**) について説明します。

構成の例は、次のようになります。

- ポリシー設定 (グローバル例):

```
pdadmin> policy set max-concurrent-web-sessions displace
```

- プロンプトの設定:

```
[session]
prompt-for-displacement = yes
```

2 回目のログイン試行時に、ユーザーは `too_many_sessions.html` 応答ページを受け取ります。ユーザーは、このページの内容をカスタマイズできます。このページのデフォルトのメッセージは、次のとおりです。

```
You are already logged in from another client. Do you want to terminate
your existing login or cancel this new login request?
```

```
Terminate existing login
Cancel this new login
```

アクションの説明:

- **Terminate existing login**

終了のアクションによって、WebSEAL の **/pkmsdisplace** 機能が呼び出されます。この機能は、既存 (元) のログインを強制終了して、そのユーザーの新規セッションを作成します。さらに、このユーザーを透過的にログインさせて、要求された URL にユーザーをリダイレクトします。

注: **pkmsdisplace** 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

ユーザーの元のブラウザーに残っている元のセッション Cookie が古い「失効した」Cookie となって、WebSEAL セッション・キャッシュの既存エントリーにマップしなくなります。ユーザーが元の (古い) ログイン・セッションから別の保護リソースにアクセスしようとする、WebSEAL は標準のログイン・フォームで認証を要求し、応答します。

このフォームに含まれる **OLDSESSION** マクロでは値が「1」に設定されていますが、これは、WebSEAL セッション・キャッシュ内のどのエントリーにも一致しなくなった古い (「失効した」) Cookie が要求に含まれていることを示しています。OLDSESSION マクロの値は、カスタマイズしたユーザーへの応答のトリガ

ー・メカニズムとして使用することができます。このカスタム応答では、セッションが有効でなくなった理由をより正確にユーザーに説明することができます。

この機能について詳しくは、333 ページの『古いセッション Cookie に対するカスタマイズ応答』を参照してください。

- この新規ログインの取り消し

取り消しアクションは、WebSEAL `/pkmslogout` 機能呼び出しします。この機能は、現在のログイン試行をクローズして、標準の WebSEAL ログアウト・ページをユーザーに戻します。元の (古い) ログイン・セッションは、リソースへのアクセスを継続できます。

前提条件: 最大並行セッション・ポリシーが追加構成で使用可能になっている必要があります。379 ページの『最大並行セッション・ポリシーの実行』を参照してください。

非対話式強制終了

WebSEAL 構成ファイルの `[session]` スタンザ内の **prompt-for-displacement** スタンザ・エントリーを使用して、**max-concurrent-web-sessions displace** ポリシーを超過した場合に、適切なアクションを促すプロンプトをユーザーに出すかどうかを決定します。このセクションでは、ユーザーに適切なアクションを促すプロンプトを出さない非対話式オプション (`prompt-for-displacement = no`) について説明します。

構成の例は、次のようになります。

- ポリシー設定 (グローバル例):

```
pdadmin> policy set max-concurrent-web-sessions displace
```

- プロンプトの設定:

```
[session]
prompt-for-displacement = no
```

2 回目のログイン試行時に、元の (古い) ログイン・セッションが自動的に終了されます。プロンプトは出されません。ユーザーには新規セッションが作成され、ユーザーはこの新規セッションに透過的にログインします。元の (古い) セッションは無効になります。

ユーザーの元のブラウザーに残っている元のセッション Cookie が古い「失効した」Cookie となって、WebSEAL セッション・キャッシュの既存エントリーにマップしなくなります。ユーザーが元の (古い) ログイン・セッションから別の保護リソースにアクセスしようとする、WebSEAL は標準のログイン・フォームで認証を要求し、応答します。

このフォームに含まれる `OLDSESSION` マクロでは値が「1」に設定されていますが、これは、WebSEAL セッション・キャッシュ内に存在するいずれのエントリーにも一致しなくなった古い (「失効した」) Cookie が、要求に含まれていることを示しています。`OLDSESSION` マクロの値は、カスタマイズしたユーザーへの応答のトリガー・メカニズムとして使用することができます。このカスタム応答では、セッションが有効でなくなった理由をより正確にユーザーに説明することができます。

この機能について詳しくは、333 ページの『古いセッション Cookie に対するカスタマイズ応答』を参照してください。

ユーザーごとの設定とグローバル設定

pdadmin policy コマンドは、特定のユーザーに対して設定する (**-user** オプションを使用) ことも、グローバルに設定する (**-user** オプションを使用しない) こともできます。ユーザー固有の設定は、すべて、ポリシーのグローバル設定をオーバーライドします。

ポリシーを使用不可にすることもできます (**unset** 引数を使用)。この設定では、ポリシーに含まれる値はありません。

例:

グローバル最大並行 Web セッション・ポリシーをユーザー 1 人あたり 1 セッション作成します。このポリシーの例外として、ユーザー **brian** には、最大並行 Web セッション・ポリシーとして 4 セッションを提供します。

```
pdadmin> policy set max-concurrent-web-sessions 1
pdadmin> policy set max-concurrent-web-sessions 4 -user brian
```

```
pdadmin> policy get max-concurrent-web-sessions
Maximum concurrent web sessions: 1
pdadmin> policy get max-concurrent-web-sessions -user brian
Maximum concurrent web sessions: 4
```

ユーザー **brian** に対する特定の最大並行 Web セッション・ポリシーを設定解除します。これで、ユーザー **brian** には、最大並行 Web セッション・ポリシーが適用されなくなります。ただし、1 セッションという現在のグローバル最大並行 Web セッション・ポリシーが、ユーザー **Brian** に対しても有効となります。

```
pdadmin> policy set max-concurrent-web-sessions unset -user brian
```

```
pdadmin> policy get max-concurrent-web-sessions -user brian
Maximum concurrent web sessions: unset
```

グローバル最大並行 Web セッション・ポリシーは設定解除されています。これで、ユーザー **brian** を含むすべてのユーザーに最大並行 Web セッション・ポリシーが適用されなくなります。ただし、全ユーザーに対して有効なポリシーは **unlimited** 設定と同じです。

```
pdadmin> policy set max-concurrent-web-sessions unset
```

```
pdadmin> policy get max-concurrent-web-sessions
Maximum concurrent web sessions: unset
```

最大並行セッション・ポリシーの実行

このタスクについて

WebSEAL 構成ファイルの **[session]** スタンザ内の **enforce-max-sessions-policy** スタンザ・エントリーを使用して、特定の WebSEAL インスタンスで

max-concurrent-web-sessions ポリシーを実行するかどうか制御します。

手順

- この WebSEAL インスタンスで **max-concurrent-web-sessions** ポリシーを実行するように設定するには、値「yes」(デフォルト)を入力します。例:

```
[session]
enforce-max-sessions-policy = yes
```

- この WebSEAL インスタンスで **max-concurrent-web-sessions** ポリシーを実行しないように設定するには、値「no」を入力します。例:

```
[session]
enforce-max-sessions-policy = no
```

注: このスタンザ・エントリーは、ユーザー環境のセッションを管理するように Session Management Server を構成済みの場合にのみ有効です。

```
[session]
dsess-enabled=yes
```

デフォルトでは、次のように、分散セッション環境にあるすべてのシステムでこのポリシーを実行します。

```
[session]
enforce-max-sessions-policy = yes
```

- 同一環境にある特定の WebSEAL インスタンスの **enforce-max-sessions-policy** スタンザ・エントリーを変更して、**max-concurrent-web-sessions** ポリシーの適用を次のように使用不可にすることができます。

```
[session]
enforce-max-sessions-policy = no
```

enforce-max-sessions-policy = no に設定されているこれらの WebSEAL サーバーにアクセスするユーザーには、ログイン・セッション数の制限はありません。

最大並行セッション・ポリシーの設定については、375 ページの『最大並行セッション・ポリシーの設定』を参照してください。

注: 最大並行セッション・ポリシーは、セッション・レルム単位で実行されます。詳しくは「*IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*」を参照してください。

例

最大並行セッション・ポリシーを「1」にグローバル指定するには、**pdadmin policy set** コマンドを使用します。

```
pdadmin> policy set max-concurrent-web-sessions 1
```

ユーザーおよび最大並行セッション・ポリシーの切り替え

アドミニストレーターがユーザー切り替えを使用して、別のユーザー名を偽名として使用した場合、Session Management Server のセッションはユーザー切り替えアド

ミニストレーターに属するとみなされます。最大並行セッション・ポリシーは、ユーザー切り替えアドミニストレーターに適用され、偽名を使用されたユーザーには適用されません。

例えば、最大並行セッション・ポリシー数が 1 のユーザー「brian」が WebSEAL サーバーにログインしている場合も、ユーザー切り替えアドミニストレーターは偽名「brian」を使用することができます。ユーザー「brian」の最大並行セッション・ポリシーは偽名を使用するセッションに適用されません。

セッション・レルム内でのシングル・サインオン

このセクションでは、同じセッション・レルム内の複数のレプリカ・セットに属するサーバー間のセッション共用によって実現される、シングル・サインオン・ソリューションについて説明します。トピックは次のとおりです。

- ・ 『セッション・レルムおよびセッション共用の概念』
- ・ 382 ページの『セッション共用の構成』

セッション・レルムおよびセッション共用の概念

セッション・レルム は、セッションを共用するように構成された Web セキュリティー・サーバーの集まりです。

セッション共用を使用すると、並行セッション制限およびセッション終了を実現しながら、セッション・レルム内のすべてのサーバーでシングル・サインオンを実現することができます。ユーザーはセッション・レルム内の任意のサーバーにログオンしたり、再認証しなくてもセッション・レルム内の別のサーバーにアクセスすることができます。

例えば、会社の メイン Web サイト **www.example.com** にユーザーとしてログインしたとします。**www.example.com** サイトは、すべての WebSEAL サーバーが **www.example.com** レプリカ・セットに属している WebSEAL クラスタで処理されます。

WebSEAL は **.example.com** のドメイン・セッション Cookie をユーザー（認証されたユーザー）に提供するように構成されています。

セッションの後半では、会社の販売部門のメイン Web サイト **sales.example.com** にアクセスします。**sales.example.com** サイトは、すべての WebSEAL サーバーが **sales.example.com** レプリカ・セットに属している WebSEAL クラスタで処理されます。

Session Management Server (SMS) の構成では、セッション・レルムへのレプリカ・セットの割り当ても行います。この例では、2 つのレプリカ・セット **www.example.com** と **sales.example.com** が、同じセッション・レルムのメンバーとなるように構成されています。

sales.example.com WebSEAL クラスターはユーザーのドメイン・セッション Cookie を使用して、**www.example.com** でのユーザーのセッション情報を取得します。このセッション情報があれば、再認証は要求されず、シングル・サインオンが実現されます。

セッション共用を機能させるには、以下の条件がすべて満たされている必要があります。

- セッション・レルム内のすべてのサーバーのセッション存続時間および非アクティブ・タイムアウトの値は、同じである必要があります。
- セッション・レルム内のすべてのサーバーの認証構成およびポリシーには、互換性がなければなりません。

互換性のない構成の例として、以下の構成を考えます。

- **www.example.com** はフォーム認証用に構成されている。
- **test.example.com** は EAI 認証用に構成されている。
- リソース **www.example.com/action.jsp** は、再認証を必要とする POP によって保護されている。

ユーザーが **test.example.com** にログオンしてから、**www.example.com** にアクセスした場合は、**www.example.com** のほとんどのリソースにアクセスできます。ただし、ユーザーは **www.example.com** 上で EAI 再認証を実行できないため、**www.example.com/action.jsp** にはアクセスできません。

セッション共用の構成

このタスクについて

セッション共用を構成するには、以下のステップを実行します。

- 同じセッション・レルム内のセッション共用に参加するすべてのレプリカ・セットを、単一の SMS セッション・レルムに割り当てます。

『セッション・レルムへのレプリカ・セットの割り当て』を参照してください。

- セッション Cookie に同じ名前を使用するように、セッション・レルム内のすべての Web セキュリティー・サーバーを構成します。

383 ページの『セッション Cookie 名の構成』を参照してください。

- セッション・レルム内のすべての Web セキュリティー・サーバーに、ドメイン・セッション Cookie に使用する DNS ドメイン・リストを構成します。

384 ページの『DNS ドメインの構成』を参照してください。

セッション・レルムへのレプリカ・セットの割り当て

このタスクについて

単一の SMS セッション・レルムに、参加するすべてのレプリカ・セットを割り当てます。

セッション・レルムの構成については、「*IBM Security Access Manager for Web 共用セッション管理 管理ガイド*」を参照してください。

このステップで WebSEAL 構成を変更する必要はありません。

セッション Cookie 名の構成

このタスクについて

セッション Cookie に同じ名前を使用するように、セッション・レルム内のすべての Web セキュリティー・サーバーを構成します。

WebSEAL セッションに使用する Cookie 名は、WebSEAL 構成ファイルの **[session]** スタンザ内の **tcp-session-cookie-name** および **ssl-session-cookie-name** スタンザ・エントリーで指定されます。例えば、以下のようになります (TCP および SSL セッションのデフォルト WebSEAL Cookie 名)。

```
[session]
tcp-session-cookie-name = PD-H-SESSION-ID
ssl-session-cookie-name = PD-S-SESSION-ID
```

332 ページの『セッション Cookie 名のカスタマイズ』も参照してください。

同じセッション・レルム内で WebSEAL および Plug-in for Web Servers セキュリティー・サーバーを使用している場合は、すべてのサーバーで同じセッション Cookie 名が使用されていることを確認します。デフォルトでは、WebSEAL および Plug-in for Web Servers では異なる Cookie 名が使用されます。以下の構成を選択することができます。

- Plug-in for Web Servers Cookie 名を使用するように、デフォルトの WebSEAL Cookie 名構成を変更する。
- WebSEAL Cookie 名を使用するように、デフォルトの Plug-in for Web Servers Cookie 名構成を変更する。
- WebSEAL と Plug-in for Web Servers の両方の構成にカスタム Cookie 名を使用する。

例

以下の URL にある WebSEAL で保護された Web サイトがあるとします。

`https://www.example.com`

また、以下の URL にある Web サイトが Web サーバーのプラグインによって保護されているとします。

`https://test.example.com`

この 2 つの Web サイト間のセッション共有を構成するには、Web サーバーのプラグインで使われるデフォルト Cookie 名と一致するように、WebSEAL 構成ファイル内のデフォルト・セッション Cookie 名を変更します。

```
[session]
tcp-session-cookie-name = AM-DSESS-SESSION-ID
ssl-session-cookie-name = AM-DSESS-SECURE-SESSION-ID
```

DNS ドメインの構成

このタスクについて

同じセッション・レルム内のすべての Web セキュリティー・サーバーに、ドメイン・セッション Cookie に使用する DNS ドメイン・リストを構成します。

WebSEAL のセッションの Cookie は、Session Management Server (SMS) を使用する WebSEAL 環境で使用される必須のセッション ID データ・タイプです。通常、セッションの Cookie は、サーバー特定 (またはホスト) の Cookie です。ブラウザは、最初に Cookie を送信したサーバーにホスト・タイプ Cookie を戻すだけです。

同じ DNS ドメイン内の複数のクラスターでセッション共用を使用可能にするには、ドメイン・タイプ Cookie を使用する必要があります。

WebSEAL 構成ファイルの **[session-cookie-domains]** スタンザ内の **domain** スタンザ・エントリーに参加するドメインを定義します。

domain スタンザ・エントリーをアンコメントして、値を指定して使用すると、WebSEAL セッション Cookie は自動的にドメイン・タイプ Cookie になります。例：

```
[session-cookie-domains]
domain = example1.com
domain = example2.com
```

WebSEAL は、ユーザーが接続している Web サイトおよび **[session-cookie-domains]** スタンザに基づいて、ドメイン・セッション Cookie に使用するドメインを決定します。セッション・レルムに使用するドメインと一致するエントリーを、このスタンザに追加します。

例として、3 つの仮想ホストを持つ WebSEAL サーバーを考えます。

```
www.example.com
www.abc.ibm.com
www.tivoli.com
```

この例に適用される条件は、以下のとおりです。

- **www.example.com** サイトが「**example.com**」セッション・レルムに参加する。
- **www.abc.ibm.com** サイトが「**abc.ibm.com**」セッション・レルムに参加する。
- **www.tivoli.com** サイトはその他の Web セキュリティー・サーバーとセッションを共用しないため、セッション・レルムに割り当てられない。

example.com と **abc.ibm.com** の両方のドメインにドメイン・セッション Cookie を構成するには、参加するそれぞれの Web セキュリティー・サーバーに以下の構成オプションを設定します。

```
[session-cookie-domains]
domain = example.com
domain = abc.ibm.com
```

ログイン・ヒストリーの構成

このタスクについて

Session Management Server (SMS) は、ユーザーの最後のログイン時刻や、最後のログイン成功以降に試行されたログイン失敗の回数に関する情報を記録するように構成できます。この情報をログイン時にユーザーに表示するには、カスタマイズされた JavaServer Page (JSP) を使用します。この情報は、そのアカウントで発生したあらゆるアクティビティに関するアラートをユーザーに表示します。

ログイン・ヒストリーを構成するには、以下のステップを実行します。

手順

1. Session Management Server (SMS) を使用するように WebSEAL を構成します。
363 ページの『WebSEAL の SMS の構成』を参照してください。
2. Session Management Server (SMS) にログイン・ヒストリー・データベースを構成します。
「*IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*」を参照してください。
3. ログイン失敗時に WebSEAL が Session Management Server (SMS) に通知できるように設定します。
『ログイン失敗の通知の使用可能化』を参照してください。
4. 適切な junction オプションを使用して、SMS への junction を作成します。
386 ページの『Session Management Server への junction の作成』を参照してください。
5. ログイン・ヒストリー JSP へのアクセスを許可します。
386 ページの『ログイン・ヒストリー JSP へのアクセスの許可』を参照してください。
6. ログイン・ヒストリー JavaServer Page (JSP) をカスタマイズし、ご使用の Web サイトと統合します。
387 ページの『ログイン・ヒストリーを表示するための JSP のカスタマイズ』を参照してください。

ログイン失敗の通知の使用可能化

このタスクについて

ログイン失敗時に Session Management Server (SMS) に通知するように WebSEAL を構成します。SMS は、この情報に基づいてログイン・ヒストリーを生成し、次のログインの成功時にこの情報をユーザーに表示します。

ログイン失敗を通知できるように WebSEAL を設定するには、**[session]** スタンザ内の **register-authentication-failures** スタンザ・エントリーの値を「yes」に設定します。

デフォルト設定は「no」です (ログイン失敗時の通知は使用不可です)。

例

```
[session]  
register-authentication-failures = yes
```

Session Management Server への junction の作成

ログイン・ヒストリー JavaServer Page (JSP) は Session Management Server (SMS) と同じサーバーに配置する必要があります。

手順

1. SMS および JSP への標準 WebSEAL junction を作成します

例:

```
pdadmin> server task server create -t tcp -h sms.example.com /sms
```

2. 次のように、HTTP ヘッダー **SMS-Session-Index** を使用して、資格情報属性 **tagvalue_session_index** を junction に渡すように WebSEAL を構成します。

例:

```
pdadmin> object modify /WebSEAL/www.example.com/sms set attribute  
HTTP-Tag-Value session_index=SMS-Session-Index
```

ログイン・ヒストリー JSP へのアクセスの許可

このタスクについて

ログイン情報を表示する JSP を除き、Session Management Server マシン上のリソースにどのユーザーもアクセスできないようにする必要があります。Session Management Server マシンへの junction に適切な ACL を適用する必要があります。

手順

1. 以下の **pdadmin** コマンドを使用して、SMS マシン上のすべてのリソースへのアクセスを拒否する ACL を、SMS への Junction に作成して、付加します。

```
pdadmin> acl create noaccess  
pdadmin> acl modify noaccess set group iv-admin TcmdbsvaBR1r  
pdadmin> acl modify noaccess set any-other T  
pdadmin> acl modify noaccess set unauth T
```

```
pdadmin> acl attach /WebSEAL/server-object/sms noaccess
```

2. 以下のコマンドを使用して、認証されたユーザーに JSP (lastLogin.jsp) への一般的な「読み取り」アクセスを許可する明示的な ACL を作成して、付加します。

```
pdadmin> acl create authonly  
pdadmin> acl modify authonly set user sec_master TcmdbsvaBR1r  
pdadmin> acl modify authonly set group iv-admin TcmdbsvaBR1r  
pdadmin> acl modify authonly set any-other Tr  
pdadmin> acl modify authonly set unauth T
```

```
pdadmin> acl attach /WebSEAL/server-object/sms/DSess/lastLogin.jsp authonly
```

3. その他のアプリケーションが Session Management Server マシンでホストされている場合は、ユーザーがそれらのアプリケーションにアクセスできるように ACL を調整します。ただし、最後にログインした JSP を除いて、/DSess サブ

ディレクトリーのすべてのリソースにユーザーがアクセスできないようにする必要があります。具体的には、ACL を DSess サブディレクトリーに付加する場合、以下の **pdadmin** コマンドを使用します。

```
pdadmin> acl attach /WebSEAL/server-object/sms/DSess noaccess
```

ログイン・ヒストリーを表示するための JSP のカスタマイズ

Session Management Server (SMS) をインストールすると、ログイン中のすべてのユーザーに関するログイン・ヒストリーを表示できるサンプル JavaServer Page (JSP) もインストールされます。

このタスクについて

このサンプル・ログイン・ヒストリー JSP は、ご使用のインストール済み環境に合わせてカスタマイズできます。

Session Management Server マシン上のこのページのデフォルト・ロケーションは、以下のとおりです。

```
/DSess/lastLogin.jsp
```

デフォルトでは、このサンプル JSP で戻されるページには、単に以下のメッセージが表示されます。

```
You last logged in at <time> on <date>.
There have been <N> unsuccessful login attempts since that time.
The last unsuccessful login attempt was at <time> on <date>.
```

この JSP をカスタマイズして、最初の認証時にユーザーに表示されるウェルカム・ページに統合することができます。例えば、ユーザーが WebSEAL に対する最初の認証を行った場合、常にログイン・ヒストリーの報告ページに転送されるように、自動リダイレクト (**[acct-mgt]** スタンザ、**login-redirect-page** スタンザ・エントリー) を構成できます。

サンプル・ログイン・ヒストリー JSP のカスタマイズ方法について詳しくは、「*IBM Security Access Manager for Web: 共用セッション管理 管理ガイド*」を参照してください。

第 6 部 許可

第 21 章 許可の構成

この章では、許可サービスおよび許可プロセスに影響を与える WebSEAL の機能について説明します。

この章で扱うトピックは以下のとおりです。

- 『WebSEAL 固有の ACL ポリシー』
- 393 ページの『保護品質 POP』
- 393 ページの『許可データベースの更新とポーリングの構成』
- 394 ページの『保護品質レベルの構成』
- 397 ページの『許可決定情報』
- 397 ページの『OAuth 許可決定のサポート』

WebSEAL 固有の ACL ポリシー

保護オブジェクト・スペース内の `/WebSEAL` コンテナには、以下のようなセキュリティの考慮事項が適用されます。

- WebSEAL オブジェクトは、オブジェクト・スペースの WebSEAL 領域に対する ACL 継承のチェーンを開始します。
- 他の ACL を明示的に適用しない場合、このオブジェクトは、Web スペース全体のセキュリティ・ポリシーを (継承によって) 定義します。
- このポイントの下にあるいずれのオブジェクトにアクセスするにも、全探索 (T) 許可を使用する必要があります。

Security Access Manager ACL ポリシーについて詳しくは、「*IBM Security Access Manager for Web: 管理ガイド*」を参照してください。

`/WebSEAL/host-instance_name`

このサブディレクトリー・エントリーは、特定の WebSEAL インスタンス用の Web スペースの始めを表します。次のようなセキュリティの考慮事項がこのオブジェクトに適用されます。

- このポイントの下にあるいずれのオブジェクトにアクセスするにも、全探索 (T) 許可を使用する必要があります。
- 他の ACL を明示的に適用しない場合、このオブジェクトは、このマシンのオブジェクト・スペース全体のセキュリティ・ポリシーを (継承によって) 定義することになります。

`/WebSEAL/host-instance_name/file`

このサブディレクトリー・エントリーは、HTTP アクセスの際に検査されるリソース・オブジェクトを表します。

検査される許可は、要求された操作によって異なります。

WebSEAL ACL 許可

以下の表は、オブジェクト・スペースの WebSEAL 領域に適用される ACL 許可について説明しています。

	操作	説明
r	読み取り	Web オブジェクトを表示します。
x	実行	CGI プログラムを実行します。
d	削除	Web スペースから Web オブジェクトを除去します。
m	変更	HTTP オブジェクトを PUT します。(HTTP オブジェクトを WebSEAL オブジェクト・スペースに入れ、公表する。)
l	リスト	ポリシー・サーバーが Web スペースのディレクトリーのリストを自動的に作成する際に必要になります。 この許可は、デフォルトの「index.html」ページがない場合に、クライアントがディレクトリー内容のリストを見られるかどうかを制御します。

デフォルトの /WebSEAL ACL ポリシー

デフォルトの WebSEAL ACL である **default-webseal** の中核エントリーには、以下のものがあります。

Group iv-admin	Tcmdbsvarx1
Group webseal-servers	Tgmdbsrx1
User sec_master	Tcmdbsvarx1
Any-other	Trx
Unauthenticated	T

このデフォルト ACL は、インストール時に、オブジェクト・スペース内の /WebSEAL コンテナ・オブジェクトに付加されます。

webseal-servers グループには、セキュア・ドメイン内の個々の WebSEAL サーバーについて 1 つずつエントリーが含まれています。デフォルトの許可では、サーバーはブラウザ要求に応答することができます。

全探索のアクセス許可があれば、Web Portal Manager 内で表示される Web スペースを拡張することができます。リスト許可が付与されると、Web portal Manager Web スペースのコンテンツを表示することができます。

ACL 名に使用できる文字

ACL 名を作成するために使用できる有効文字は、以下のとおりです。

- A から Z
- a から z
- 0 から 9
- 下線 (_)
- ハイフン (-)
- 円記号 (¥)

- 2 バイト文字セットの任意の文字

ACL 名の作成について詳しくは、「*IBM Security Access Manager for Web 管理ガイド*」を参照してください。

保護品質 POP

保護品質の保護オブジェクト・ポリシー (POP) 属性によって、オブジェクトについての操作を行う場合に、どのレベルのデータ保護が必要であるかを指定することができます。

保護品質 POP 属性を使用して、要求リソースにアクセスを認可するかどうかを決定します。リソースに対する ACL 検査が成功すると、保護品質 POP が検査されます。保護品質 POP は存在するが、リソース・マネージャー (WebSEAL) が必要レベルの保護を保証できない場合、要求は拒否されます。

保護品質 POP 属性を設定するための構文は、次のとおりです。

```
pdadmin> pop modify pop-name set qop {none|integrity|privacy}
```

保護品質レベルが保全性またはプライバシーに設定されていると、WebSEAL では Secure Socket Layer (SSL) の使用を介したデータ暗号化が必要です。

以下に例を示します。

```
pdadmin> pop modify test set qop privacy
```

許可データベースの更新とポーリングの構成

このセクションは、以下のトピックで構成されています。

- 『データベースの更新とポーリングの概念』
- 394 ページの『更新通知の listen の構成』
- 394 ページの『許可データベースのポーリングの構成』

データベースの更新とポーリングの概念

Security Access Manager ポリシー・サーバー (pdmgrd) は、マスター許可ポリシー・データベースを管理し、セキュア・ドメイン内の他の Security Access Manager サーバーに関するロケーション情報を維持しています。Security Access Manager アドミニストレーターは、セキュア・ドメインにいつでもセキュリティー・ポリシーの変更操作をすることができます。ポリシー・サーバーは、セキュリティー・ポリシー変更がインプリメントされるたびに、マスター許可データベースに対して必要な調整を行います。

ポリシー・サーバーは、マスター許可データベースに変更を加えた後で、セキュア・ドメイン内にあって個々のポリシー・エンフォーサー (WebSEAL など) をサポートしているすべてのレプリカ・データベースに、この変更を示す通知を送ることができます。これを受けたポリシー・エンフォーサーは、マスター許可データベースから実際のデータベース更新を要求する必要があります。

リソース・マネージャーおよびポリシー・エンフォーサーとしての WebSEAL は、次の 3 つのオプションのいずれかを使用して、許可データベースの変更に関する情報を取得できます。

- ポリシー・サーバーからの更新通知を `listen` する (構成可能であり、デフォルトでは使用可能にされている)。
- マスター許可データベースを定期的にチェック (ポーリング) する (構成可能であり、デフォルトでは使用不可にされている)。
- `listen` とポーリングの両方を使用可能にする。

WebSEAL 構成ファイルの **[aznapi-configuration]** スタンザには、更新通知の `listen` およびデータベースのポーリングを構成するためのスタンザ・エントリーが入っています。

更新通知の `listen` の構成

WebSEAL による更新通知 `listen` を使用可能または使用不可にするには、WebSEAL 構成ファイルの **[aznapi-configuration]** スタンザ内にある **listen-flags** スタンザ・エントリーを使用します。デフォルトでは、通知 `listen` は使用可能にされています。通知 `listen` を使用不可にするには、「`disable`」と入力します。

```
[aznapi-configuration]
listen-flags = enable
```

通知リスナーの SSL ポートは、WebSEAL 構成ファイルの **[ssl]** スタンザ内にある **ssl-listening-port** スタンザ・エントリーにより指定されます。

```
[ssl]
ssl-listening-port = 7234
```

注: **ssl-listening-port** エントリーを直接変更しないでください。このオプションは、`listen` ポートが変更されたことがポリシー・サーバーによって検出できるようにするために、**scrsslcfg -chgport** コマンドを発行することによってのみ変更する必要があります。それ以外の方法では、リソース・マネージャーはポリシー更新通知または `pdadmin` サーバー・タスク・コマンドを受け取ることができません。

許可データベースのポーリングの構成

WebSEAL が定期的にマスター許可データベースをポーリングして更新情報の有無を確認するように、構成することができます。**cache-refresh-interval** スタンザ・エントリーは、「`default`」、「`disable`」、または特定の時間間隔 (秒数) に設定できます。「`default`」(デフォルト) の設定は 600 秒です。デフォルトでは、ポーリングは使用不可にされています。

```
[aznapi-configuration]
cache-refresh-interval = disable
```

保護品質レベルの構成

保護品質 (QOP) を構成することにより、SSL (HTTPS) を介して WebSEAL にアクセスするために必要な、デフォルトの暗号化レベルをコントロールすることができます。デフォルトの保護品質管理は、WebSEAL 構成ファイルの「**SSL QUALITY OF PROTECTION MANAGEMENT**」セクション内のスタンザ・エントリーを使用して制御します。

- QOP 管理を使用可能または使用不可にするには、**ssl-qop-mgmt** スタンザ・エントリーを使用します。
- 使用できる暗号化レベルは、**[ssl-qop-mgmt-default]** スタンザで指定します。

1. 保護品質管理を使用可能にします。

```
[ssl-qop]
ssl-qop-mgmt = yes
```

2. HTTPS アクセスに関するデフォルトの暗号化レベルを指定します。構文は次のようになります。

```
default = {ALL|NONE|cipher_level}
```

cipher_level でサポートされている値は次のとおりです。

```
NONE, ALL, NULL, DES-56, FIPS-DES-56, DES-168, FIPS-DES-168,
RC2-40, RC2-128, RC4-40, RC4-56, RC4-128, AES-128, AES-256
```

値「NONE」は、暗号化を使用不可にします。

例:

```
[ssl-qop-mgmt-default]
default = ALL
```

選択した暗号グループも指定できるという点に注意してください。

```
[ssl-qop-mgmt-default]
default = RC4-128
default = RC2-128
default = DES-168
```

注:

- NONE を指定すると、SSL 接続は許可されません。
- NULL を指定すると、暗号化されていない SSL 接続が許可されます。
- ALL を指定すると、すべてのタイプの SSL 接続が許可されます。
- 特定の保護品質暗号選択の接続に複数の暗号/MAC レベルを使用可能にできます。これらの構成は暗号化ビットの強度は同じで、MAC メソッドが異なる (SHA1 または MD5) だけです。
- RC2-128 は、SSLv2 でのみ使用できます。これが唯一の暗号選択である場合、WebSEAL は影響を受ける接続の SSLv3 および TLSv1 を使用不可にします。
- NULL、FIPS-DES-56、FIPS-DES-168、RC4-56、AES-128、および AES-256 は、SSLv3 および TLSv1 でのみ使用できます。特定の接続に対して、これらが唯一の使用可能な暗号である場合、影響を受ける接続の SSLv2 は使用不可にされます。
- AES サポートは、base-crypto-library の設定に基づいて、GSKit によって自動的に決定されます。AES-128 および AES-256 が使用可能になるのは GSKit によって AES サポートが使用可能にされている場合だけで、そうでない場合は無視されます。
- FIPS-DES-56 および FIPS-DES-168 が使用可能になるのは fips-mode-processing が使用可能にされている (yes に設定されている) 場合だけです。そうでない場合は無視されます。

Security Access Manager は GSKit 8 を使用します。インターネット・セキュリティで SSLv2/TLS で使用されているときに GSKIT7 がサポートしている暗号仕様は以下のとおりです。

```
SSL_RSA_WITH_NULL_MD5
SSL_RSA_WITH_NULL_SHA
SSL_RSA_EXPORT_WITH_RC4_40_MD5
SSL_RSA_WITH_RC4_128_MD5
SSL_RSA_WITH_RC4_128_SHA
SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
SSL_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
```

上記の TLS 暗号仕様は、SSLV3 でも使用されます。

個々のホストおよびネットワークの保護品質の構成

ssl-qop-mgmt = yes スタンザ・エントリーは、**[ssl-qop-mgmt-hosts]** スタンザおよび **[ssl-qop-mgmt-networks]** スタンザ内の設定もすべて使用可能にします。これらのスタンザを使用することで、保護品質管理を特定のホスト/ネットワーク/ネットマスク IP アドレス単位で行うことができます。

注: **[ssl-qop-mgmt-hosts]** スタンザおよび **[ssl-qop-mgmt-networks]** スタンザは、前のバージョンの WebSEAL との互換性を維持するためにのみ提供されています。Security Access Manager の構成では、これらのスタンザは使用しないことをお勧めします。また、インターネット・プロトコル (IP) バージョン 6 (IPv6) アドレスは、これらのスタンザでサポートされていません。

[ssl-qop-mgmt-default] スタンザは、**[ssl-qop-mgmt-hosts]** および **[ssl-qop-mgmt-networks]** スタンザ内に一致するものがないすべての IP アドレス用に使用されている暗号をリストします。

ホストの場合の構成構文の例:

```
[ssl-qop-mgmt-hosts]
xxx.xxx.xxx.xxx = ALL
yyy.yyy.yyy.yyy = RC2-128
```

ネットワーク/ネットマスクの場合の構成構文の例:

```
[ssl-qop-mgmt-networks]
xxx.xxx.xxx.xxx/255.255.255.0 = RC4-128
yyy.yyy.yyy.yyy/255.255.0.0 = DES-56
```

[ssl-qop-mgmt-hosts] の下に指定された IP アドレスのエントリーのほうが、**[ssl-qop-mgmt-networks]** の中の同じアドレスのエントリーよりも優先順位が高いことに注意してください。同様に、**[ssl-qop-mgmt-networks]** の中のエントリーのほうが、**[ssl-qop-mgmt-default]** の中の同じアドレスのエントリーよりも優先順位が高くなります。

互換性との兼ね合いで、[ssl-qop-mgmt-hosts]または[ssl-qop-mgmt-networks]を使用しなければならない場合は、すべてのスタンザの中の IP アドレス設定を検討して、特定の IP アドレスが複数のスタンザにリストされていないことを確かめてください。ある IP アドレスが複数のスタンザにリストされている場合は、評価の順序によって望ましい構成が生成されることを確認してください。

許可決定情報

WebSEAL では、HTTP 要求の構成済みエレメントを、許可決定を下す際に使用するための許可フレームワークに渡すことができます。

以下の HTTP 要求エレメントを許可フレームワークに渡すことができます。

- 要求の HTTP メソッド
- 要求の HTTP スキーム
- 要求 URI
- 要求に含まれる特定の HTTP ヘッダー
- 要求に含まれる特定の POST データ・エレメント

WebSEAL 構成ファイル内の [azn-decision-info] スタンザには、許可フレームワークに渡される追加情報が指定されます。詳しくは、「*IBM Security Web Gateway Appliance: Web* リバース・プロキシ・スタンザ・リファレンス」を参照してください。

OAuth 許可決定のサポート

OAuth は、リソース所有者 (別のクライアントやエンド・ユーザーなど) の代わりにクライアントがサーバー・リソースにアクセスする方法を提供します。また、エンド・ユーザーが自分の資格情報 (通常は、ユーザー名とパスワードのペア) を共有することなく、ユーザー・エージェントのリダイレクトを使用して、サード・パーティーにサーバー・リソースへのアクセスを許可するためのプロセスも提供します。

WebSEAL では、EAS プラグインがサポートされます。このプラグインは、Tivoli Federated Identity Manager OAuth バージョン 1.0 および 2.0 の機能を利用します。このプラグインを使用すると、WebSEAL 要求に対する標準許可の一部として OAuth 決定が行われます。この機能を使用するには、ご使用の環境に OAuth トークンを拒否または許可するために構成されている Tivoli Federated Identity Manager サーバーが必要です。詳しくは、Tivoli Federated Identity Manager サーバーの製品資料を参照してください。

OAuth EAS の概要

ハイレベルでは、EAS フレームワークにより、許可決定時にカスタム・モジュールを呼び出して、カスタマイズされた意思決定論理を許可決定に追加することができます。EAS の構成は、特定の POP が検出されたときにモジュールを呼び出すか、特定の許可ビットが許可決定に適用されたときにモジュールを呼び出すかを制御します。

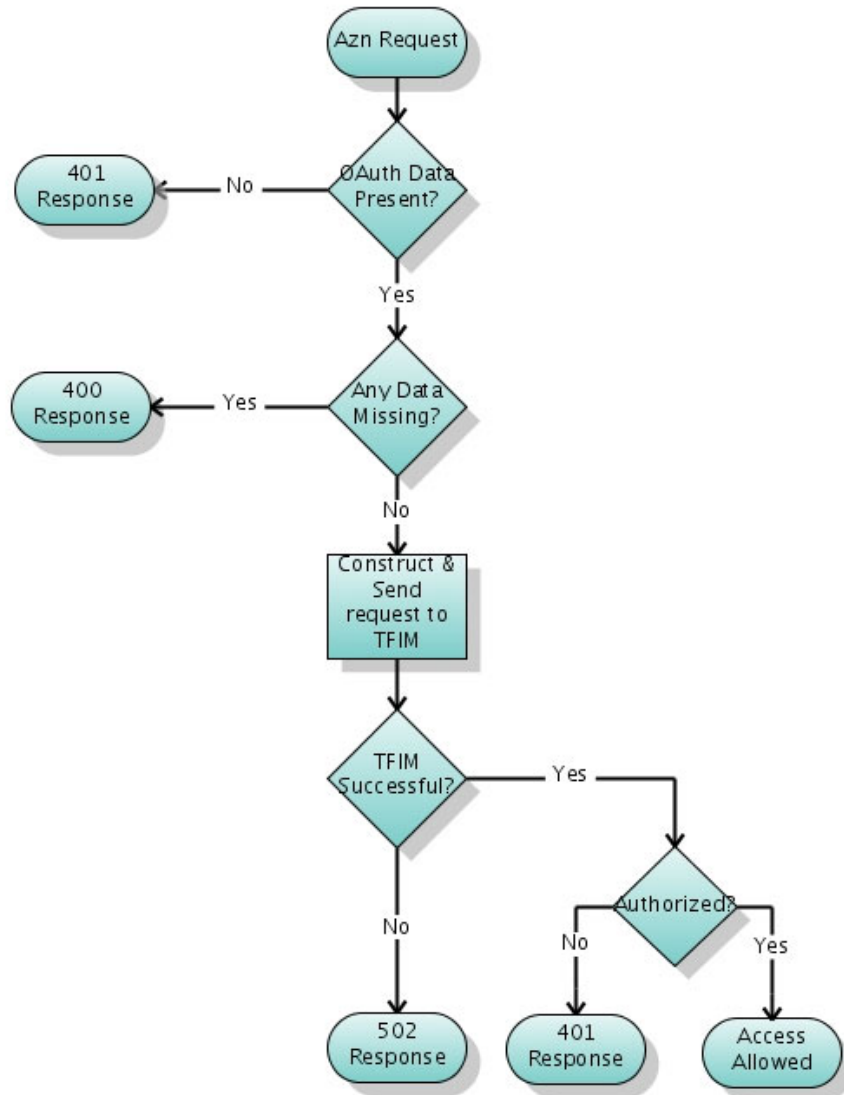


図 27. OAuth EAS の論理フロー

図 27 は、許可決定を行うときの OAuth EAS 全体の論理フローを示しています。以下のステップは、この許可プロセスの概要を示します。

1. OAuth EAS が許可要求を受け取ります。
2. OAuth EAS は、必要な OAuth データが存在するかどうかを判断します。
3. OAuth データが欠落している場合、401 応答が生成され、それ以上の処理は行われません。必要な OAuth データが使用可能な場合は、次のステップに進みます。
4. EAS は、必要なすべてのデータが要求で使用可能であることを確認します。

5. データが欠落している場合、400 応答が生成され、それ以上の処理は行われません。すべてのデータが使用可能な場合、EAS は 要求セキュリティー・トークン (RST) を構成し、Tivoli Federated Identity Manager サーバーに送信します。
6. Tivoli Federated Identity Manager が要求を処理します。Tivoli Federated Identity Manager の処理が失敗した場合、502 応答が生成され、それ以上の処理は行われません。そうでない場合、Tivoli Federated Identity Manager は OAuth EAS にアクセス決定を返します。
7. 要求が許可されると、要求されたリソースへのアクセス権限が付与されます。要求が許可されない場合、401 応答が生成されます。

OAuth 決定を含める WebSEAL の構成

OAuth 許可決定を行うため、Tivoli Federated Identity Manager では、要求に関連した固有の情報を必要とします。必要なデータには以下が含まれます。

- **許可データ**。このデータは、許可ヘッダー、照会ストリング、または POST データのいずれかから取得されます。
- **リソース情報**。このデータは HTTP 要求から取得され、OAuth シグニチャーの検証に使用されます。

WebSEAL は、EAS プラグインを使用してこの必要なデータを提供し、Tivoli Federated Identity Manager で OAuth 機能を使用します。EAS プラグインは、Security Access Manager Web Security Runtime パッケージによってインストールされます。

OAuth 決定を WebSEAL 要求の標準許可の一部として含めるには、以下のタスクを実行する必要があります。

1. 必要な許可決定データを構成します。
2. 追加の EAS 固有のデータを構成します。

この構成では、正しいデータが要求ごとに EAS に確実に渡されるようにします。

許可決定データ:

RST を正しく構成するため、EAS では要求自体からのさまざまな情報を必要とします。WebSEAL は、この情報を EAS に提供するように構成する必要があります。

必要なデータの大半は、**[azn-decision-info]** スタンザでこれらの HTTP 要求エレメントを指定することによって、すべての許可要求で提供されます。 397 ページの『許可決定情報』を参照してください。

注: 特定の状況では、POST データも必要とされます。効率を高めるため、EAS プラグインは許可決定要求ごとに毎回 POST データを提供するとは限りません。代わりに、このプラグインでは、WebSEAL 内で既存の動的アクセス決定情報を使用して、必要に応じて POST データを要求します。WebSEAL は、**[aznapi-configuration]** スタンザの **resource-manager-provided-adi** 構成エントリーに基づいて POST データの要求を認識します。

データが EAS に渡されるようにするために、この構成スタンザが正しいことは重要です。EAS が正常に機能するために、以下の構成エントリーが必要です。


```
[azn-decision-info]

#
# The following information will be provided to the authorization
# framework for every authorization request. This information
# is required by the OAuth EAS when validating an OAuth token.
#

HTTP_REQUEST_METHOD      = method
HTTP_REQUEST_SCHEME      = scheme
HTTP_REQUEST_URI         = uri
HTTP_HOST_HDR            = header:host
HTTP_CONTENT_TYPE_HDR    = header:content-type
HTTP_TRANSFER_ENCODING_HDR = header:transfer-encoding
HTTP_AZN_HDR             = header:authorization

[aznapi-configuration]

resource-manager-provided-adi = AMWS_pb_
```

EAS 固有データ:

EAS には、正常に機能するために固有の構成データが必要です。このデータは、ほとんどの場合、**[oauth-eas]** スタンザに格納されています。

WebSEAL には、これらのスタンザ・エントリーを含んだ `oauth_eas.conf.template` と呼ばれるテンプレート・ファイルが用意されています。

[oauth-eas] スタンザの必須構成エントリーの 1 つは **cluster-name** です。このエントリーには、OAuth サービスをホスティングする Tivoli Federated Identity Manager クラスターの名前を指定します。対応する **[tfim-cluster:<cluster>]** スタンザを構成して、指定したクラスターを定義する必要があります。

以下の抜粋は、必須スタンザの例です。

```
[oauth-eas]

# This stanza contains definitions for OAuth EAS specific information.

# ....

[tfim-cluster:oauth-cluster]

#
# This stanza contains definitions for the cluster of TFIM
# servers that hosts the OAuth service.
#

# ....
```

これらの各スタンザで必要な構成エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

エラー応答

場合によっては、以下を含む HTTP エラー応答をクライアントに返す必要があります。

- 400 Bad Request

- 401 Unauthorized
- 502 Bad Gateway

401 応答の場合、追加の WWW-Authenticate ヘッダーが、次の形式で応答に追加されます。

WWW-Authenticate: OAuth realm = <realm-name>

応答の HTML コンポーネントは、EAS 構成で指定されているファイルからプリロードされます。つまり、**[oauth-eas]** スタンザ内の **[bad-request-rsp-file]**、**[unauthorized-rsp-file]**、および **[bad-gateway-rsp-file]** の各構成エントリーです。**[oauth-eas]** スタンザについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

トラブルシューティング

EAS は、標準の Security Access Manager トレース・メカニズムによりトレース情報を提供します。

このメカニズムは、Security Access Manager サーバー・タスク・コマンド **trace** を使用して制御します。**[oauth-eas]** スタンザ内の **trace-component** 構成エントリーを使用して、EAS に関連するトレース・コンポーネントの名前を指定します。

第 22 章 鍵管理

この章では、WebSEAL サーバーによる証明書の処理を管理するためにユーザーが実行できるタスクについて説明します。

この章で扱うトピックは以下のとおりです。

- 『鍵管理の概要』
- 404 ページの『ローカル管理インターフェースの鍵管理』
- 404 ページの『クライアント・サイドおよびサーバー・サイドの証明書の概念』
- 405 ページの『WebSEAL 鍵データベース・ファイルの構成』
- 409 ページの『CRL 検査の構成』
- 410 ページの『CRL キャッシュの構成』
- 411 ページの『SSL 接続用 WebSEAL テスト証明書の使用』

鍵管理の概要

LMIは、WebSEAL と Security Access Manager ドメイン内の他のコンポーネントとの間の SSL 通信を使用可能にするために必要な鍵を管理します。

鍵データベース・ファイルを作成し、これらの鍵データベース・ファイルに保管されるデジタル証明書を管理するには、LMI を使用できます。

図 28 は、WebSEAL が Security Access Manager 環境の他のコンポーネントとの SSL 通信に使用する鍵管理構成を表したものです。構成スタanzas とスタanzas・エントリーは WebSEAL 構成ファイル内にあります。

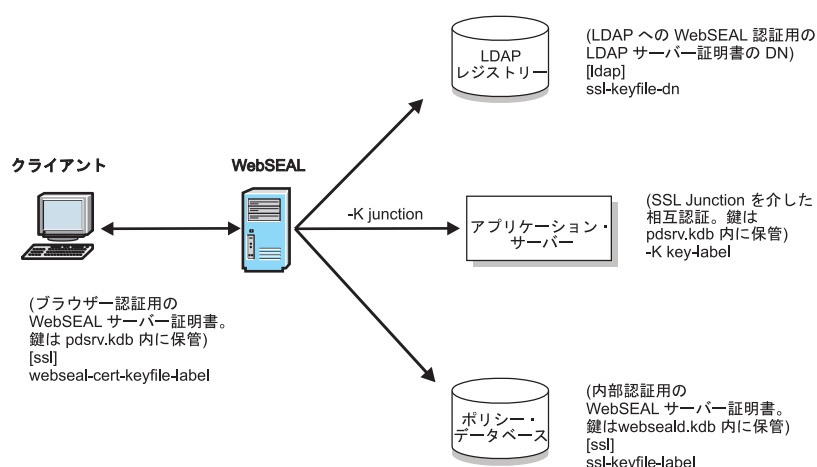


図 28. 鍵格納ファイルの管理構成

ローカル管理インターフェースの鍵管理

LMI を使用して、WebSEAL が使用するデジタル証明書を管理できます。LMI で、「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「グローバル・キー」メニューに移動し、以下の鍵管理ページにアクセスします。

SSL 証明書

「SSL 証明書」管理ページを使用して、以下のタスクを実行します。

- 現行 SSL 証明書データベース名をすべてリストまたは取得する。
- 証明書データベースを作成する。
- 証明書データベースの名前を変更する。
- 証明書データベースについて説明する。
- 証明書データベースを削除する。
- 証明書データベースをインポートする。
- 証明書データベースをエクスポートする。
- 署名者証明書を管理する。
- 個人証明書を管理する。
- 証明書要求を管理する。

SSO キー

「SSO キー」管理ページを使用して、以下のタスクを実行します。

- 現行 SSO 鍵ファイルをすべてリストする。
- 新規 SSO 鍵ファイルを作成する。
- 既存の SSO 鍵ファイルをインポートする。
- SSO 鍵ファイルをエクスポートする。
- SSO 鍵ファイルを削除する。

LTPA 鍵

「LTPA 鍵」管理ページを使用して、以下のタスクを実行します。

- 現行 LTPA 鍵ファイルをすべて取得する。
- LTPA 鍵ファイルの名前を変更する。
- LTPA 鍵ファイルを削除する。
- LTPA 鍵ファイルをエクスポートする。
- LTPA 鍵ファイルをインポートする。

これらの鍵管理タスクを実行するための LMI の使用について詳しくは、「*IBM Security Web Gateway Appliance: 管理ガイド*」を参照してください。

クライアント・サイドおよびサーバー・サイドの証明書の概念

このセクションでは、クライアント・サイドとサーバー・サイドのデジタル証明書を処理するための WebSEAL のセットアップに必要な管理タスクと構成タスクについて説明します。デジタル証明書は、SSL を介して認証に使用されます。

WebSEAL では、以下の状況下で証明書を必要とします。

- WebSEAL がサーバー・サイドの証明書を使用して、SSL クライアントに対してそれ自身を識別する。
- WebSEAL がクライアント・サイド証明書を使用して、junction 先バックエンド・サーバー (相互認証用に構成済み) に対してそれ自身を識別する。
- WebSEAL が認証局 (CA) のルート証明書のデータベースを参照して、クライアント・サイドの証明書を使用してアクセスするクライアントの妥当性検査を行う。
- WebSEAL が認証局 (CA) のルート証明書のデータベースを参照して、junction 先バックエンド・サーバーの妥当性検査を行う。

WebSEAL は、SSL の IBM Global Security Kit (GSKit) インプリメンテーションを使用して、デジタル証明書を構成および管理します。IBM Security Web Gateway Appliance には、証明書鍵データベースのセットアップと管理を行うための LMI が組み込まれています。このデータベースには、1 つ以上の WebSEAL サーバー/クライアント証明書と CA ルート証明書が保管されています。

WebSEAL には、インストール時に、デジタル証明書を使用して SSL 認証をサポートする以下のコンポーネントが入ります。

- デフォルトの鍵データベース (pdsrv.kdb)
- デフォルトの鍵データベース stash ファイル (pdsrv.sth) およびパスワード (「pdsrv」)
- いくつかの共通する CA ルート証明書
- WebSEAL が SSL クライアントに対してそれ自身を識別するために使用できる自己署名テスト証明書

実稼働環境で WebSEAL を使用する前に、既知の認証局発行の、共通して認識される証明書を適用して、このテスト証明書の代わりに使用してください。

WebSEAL 鍵データベース・ファイルの構成

このセクションは、以下のトピックで構成されています。

- 『WebSEAL 鍵データベース・ファイル』
- 406 ページの『鍵データベース・ファイル・パスワード』
- 407 ページの『WebSEAL テスト証明書』
- 407 ページの『サーバー名表示』
- 408 ページの『Security Access Manager でのサーバー間 SSL 通信』

WebSEAL 鍵データベース・ファイル

インストール中に、WebSEAL は、クライアントおよび junction 先サーバー両方の認証に使用される、デフォルト証明書の鍵データベースを提供します。WebSEAL では junction 先サーバーの認証に使用できる個別の証明書鍵データベースもオプションで提供しています。

デフォルトでは、junction 証明書鍵データベース・オプションは WebSEAL 構成ファイルでコメント化されています。このオプションが使用可能でない場合、junction はクライアントおよび junction 先サーバーに共有鍵データベースを使用するという、デフォルト動作を維持します。

注: junction 先サーバーに個別の証明書鍵データベースを使用している場合、ユーザーは、junction 鍵データベースに格納された CA 証明書によって検証されるクライアント証明書を使用できません。同様に、junction 先サーバーでは、デフォルトの証明書データベースに格納された CA 証明書によって検証される証明書を使用できません。

デフォルトの証明書鍵データベースは、WebSEAL 構成ファイルの **[ssl]** スタンザ内にある **webseal-cert-keyfile** スタンザ・エントリーにより識別されます。例えば、次のとおりです。

```
[ssl]
webseal-cert-keyfile = pdsrv.kdb
```

オプションの個別の Junction 証明書鍵データベースは、WebSEAL 構成ファイルの **[junction]** スタンザ内にある **jct-cert-keyfile** スタンザ・エントリーにより識別されます。以下に例を示します。

```
[junction]
jct-cert-keyfile = pdjct.kdb
```

LMI の「SSL 証明書」管理ページを使用して、新規の鍵データベースを作成できます。ただし、**webseal-cert-keyfile** スタンザ・エントリーにこの新規の鍵格納ファイルの名前と場所を入力して、WebSEAL がそのデータベース内にある証明書を検索し、使用できるようにしておかなければなりません。

鍵データベース・ファイル・パスワード

WebSEAL が提供する stash ファイルには、デフォルトの証明書鍵データベース **pdsrv.kdb** のパスワードが格納されています。

以下のスタンザ・エントリーは、stash ファイルの名前を指定します。

```
[ssl]
webseal-cert-keyfile-stash = pdsrv.sth
```

pdsrv.sth stash ファイル内で暗号化されたデフォルト・パスワードは、**pdsrv** です。代わりに、対応するスタンザ・エントリー内で、パスワードをプレーン・テキストで表すこともできます。以下に例を示します。

```
[ssl]
webseal-cert-keyfile-pwd = pdsrv
```

インストール中に、WebSEAL は stash ファイルを使用して鍵格納ファイルのパスワードを取得します。**webseal-cert-keyfile-pwd** スタンザ・エントリーはコメント化されています。stash ファイルを使用すると、パスワードが構成ファイル内のテキストで表示されないようにすることができます。

注: 使用したいパスワード・スタンザ・エントリーだけアンコメントしてください。パスワードと stash ファイル・スタンザ・エントリーの両方を指定すると、パスワード値が使用されます。

WebSEAL テスト証明書

インストール中に、WebSEAL は非セキュアの自己署名テスト証明書を提供します。テスト証明書はサーバー・サイドの証明書としての役割を果たすもので、WebSEAL はこれを使用して、自身を SSL クライアントに識別させます。

このテスト証明書を効果的に管理できるようにするために、この証明書はデフォルトの証明書としてインストールされません。代わりに、**webseal-cert-keyfile-label** スタンザ・エントリーを使用して、この証明書をアクティブなサーバー・サイドの証明書として指定し、鍵格納ファイル・データベース内で「デフォルト」として指定されている他の証明書をオーバーライドすることができます。

```
[ssl]
webseal-cert-keyfile-label = WebSEAL-Test-Only
```

注: WebSEAL は、GSKit 証明書処理機能性を使用します。GSKit は、鍵格納ファイル・データベースの中にある証明書を、デフォルトの証明書と指定できるようにしますが、必須ではありません。

このテスト証明書は、WebSEAL が、SSL 使用可能ブラウザの要求に応答できるようにしますが、これを (適切なルート CA 証明書を含まない) ブラウザーで検査することはできません。このデフォルト証明書の秘密鍵は、あらゆる WebSEAL 配布に含まれているため、この証明書では、真に安全な通信は約束していません。

LMI を使用して、認証局 (CA) に送信可能な証明書要求を生成することができます。LMI を使用して、戻されたサーバー証明書をインストールし、それにラベルを付けます。

他のシナリオ (**-K junction** など) で別の証明書を使用する場合は、LMI を使用して、その証明書を作成、インストール、およびラベル付けすることができます。この鍵格納ファイル・ラベルにはスペースを含めることはできません。

WebSEAL (デフォルトでは **user ivmgr** として実行される) は、これらの鍵データベース・ファイルに対する読み取り (r) 許可を持っている必要があります。

サーバー名表示

WebSEAL では、サーバー名表示を使用して、要求内のホスト名を識別し、一致するホスト名が含まれるサーバー証明書を送信することができます。WebSEAL が各ホストに対して使用する証明書を構成できます。

サーバー名表示は SSL および TLS プロトコルの拡張機能です。サーバー名表示により、ブラウザが接続を要求しているホスト名が識別されます。

デフォルトでは、WebSEAL はすべてのホストに同じ証明書を送信します。ただし、サーバー名表示を使用することにより、WebSEAL は、要求されたホストごとに異なる証明書を送信することができます。

サーバー名表示をサポートするには、要求が以下の要件を満たしている必要があります。

- TLS over SSL を使用して WebSEAL に接続する。SSLv2 および SSLv3 はサポートされません。

- サーバー名表示をサポートするブラウザを使用する。

WebSEAL 構成ファイルの **[ssl]** スタンザ内の **webseal-cert-keyfile-sni** 構成エントリーを使用して、WebSEAL が特定のホストに関して送信する証明書を指定します。例:

```
[ssl]
webseal-cert-keyfile-sni = <host_name>:<label>
```

説明:

<host_name>
WebSEAL が証明書を返すホストの名前。

<label>
WebSEAL が使用する証明書の名前。

注: **cn=<host_name>** の **dn** 値を含む証明書を指定します。

この構成エントリーは、複数回指定することができます。サーバー証明書ごとに個別のエントリーを指定します。

ブラウザ要求内のホスト名に対応するエントリーを WebSEAL が検出しなかった場合、WebSEAL は **webseal-cert-keyfile-label** エントリーに指定されているデフォルト証明書を送信します。要求がサーバー名の表示の要件を満たしていない場合も、WebSEAL はデフォルト証明書を使用します。例えば、ブラウザがサーバー名の表示をサポートしていない場合などです。

webseal-cert-keyfile-sni エントリーを構成しない場合、WebSEAL は単一の証明書しか送信できません。つまり、WebSEAL は別々のホストを区別することができません。ユーザーが SSL を使用してデフォルト証明書に一致しないホストに接続すると、ブラウザで証明書不一致エラーが発生します。

サーバー名表示はこの問題を解決します。**webseal-cert-keyfile-sni** を使用して、WebSEAL が各ホスト名に対して一致する証明書を提供するように構成します。

Security Access Manager でのサーバー間 SSL 通信

WebSEAL 構成ファイルの **[ssl]** スタンザには、WebSEAL が他の Security Access Manager サーバーとの内部 SSL 通信に使用する鍵格納ファイルを構成するための、4 つの追加スタンザ・エントリーが含まれています。これらのスタンザ・エントリーを変更するには、必ず **pdconfig** 構成スクリプトを使用してください。

```
[ssl]
ssl-keyfile =
ssl-keyfile-pwd =
ssl-keyfile-stash =
ssl-keyfile-label =
```

WebSEAL での証明書の失効

証明書は、さまざまな理由で失効する可能性があります。証明書を使用する前に、WebSEAL では、最新のソースを調べてその証明書が有効であることを確認する必要があります。

証明書取り消しリスト (CRL)

証明書取り消しリスト (CRL) は、不必要な証明書の妥当性検査を省くために使用できる方法です。CRL には、信頼性がないと見なされる証明書の識別が入っています。WebSEAL は、CRL 検査をサポートする SSL の GSKit インプリメンテーションを使用します。WebSEAL は、GSKit を使用して、クライアント・サイドの証明書と SSL junction からの証明書で CRL 検査を実行することができます。

認証局 (CA) は、制限された時間の間だけ有効な CRL を提供します。CA は、CRL の有効存続時間を指定します。CA には、この情報を保持する責任があります。CRL を更新するためのポリシーを調べるには、CA に問い合わせてください。

WebSEAL は、証明書の管理で OCSP、CRL、またはその両方を使用するように構成できます。デフォルトで WebSEAL (GSKit を使用) は、最初に OCSP を試し、それから CRL を試します。これら 2 つの方式に失敗した場合、WebSEAL は LDAP を試すことができます (構成されている場合)。この検索順序は RFC によって定義されており、変更できません。

WebSEAL は、CA によって証明書で指定されている証明書の分散ポイント (CDP) に接続できる状態になっている必要があります。WebSEAL がファイアウォールの背後のサーバーにインストールされている場合は、CDP を経由する通信を許可する必要があります。そうしないと、パフォーマンスに影響し、期限切れの CRL に照らして証明書の検証を行うというリスクを冒すことになります。

期限切れの CRL を使用する場合は時間制限はありません。ただし、期限切れの CRL の使用を許可すると、機密漏れが生じます。GSKit は、CRL の期限が切れていると判断すると、UNDETERMINED 状況メッセージを返します。その後このアプリケーションは、取るべき最善のアクションを決定することができます。WebSEAL で実行するアクションは、`[ssl]` スタンザの構成オプション `undetermined-revocation-cert-action` を、`ignore`、`log`、または `reject` のいずれかに設定することによって構成できます。

CRL 検査の構成

WebSEAL は、CRL 検査を行うために CRL リストの場所を認識していなければなりません。クライアント・サイドの証明書の認証時に CRL 検査のために参照できる LDAP サーバーの場所を示すスタンザ・エントリは、WebSEAL 構成ファイルの `[ssl]` スタンザの中にあります。

```
[ssl]
#crl-ldap-server = server-name
#crl-ldap-server-port = port-id
#crl-ldap-user = webseal-admin-name
#crl-ldap-user-password = admin-password
```

SSL junction 間の認証時に CRL 検査のために参照できる LDAP サーバーの場所を示すスタンザ・エントリは、WebSEAL 構成ファイルの `[junction]` スタンザ内にあります。

```
[junction]
#cr1-ldap-server = server-name
#cr1-ldap-server-port = port-id
#cr1-ldap-user = webseal-admin-name
#cr1-ldap-user-password = admin-password
```

デフォルトでは、CRL 検査は使用不可になっています (スタンザ・エントリーはコメント化されています)。証明書の認証時に CRL 検査を使用可能にするには、各スタンザ・エントリーをアンコメントして、適切な値を入力してください。

cr1-ldap-user スタンザ・エントリーの値がヌルのときは、SSL 認証メカニズムが匿名ユーザーとして LDAP サーバーにバインドされなければならないことを示しています。

証明書の分散ポイント

CA は、証明書において、失効情報の入手先を指定します。これらの詳細な情報は、WebSEAL や GSKit ライブラリーによっては提供されません。

まれにですが、証明書で複数の CDP が指定されていることがあります。CDP が複数存在することの主な理由は、LDAP や HTTP などの異なるプロトコルに対応するためです。証明書が複数の CDP を指定して構成されている場合、WebSEAL は有効な結果が返されるまで各 CDP に問い合わせます。

さまざまな CA の証明書を使用することができます。各 CRL には各 CA が署名し、混同されることのないようになっています。各証明書には、独自の CDP が含まれています。

CRL キャッシュの構成

GSKit を使用すると、クライアント・サイドの証明書と SSL junction からの証明書の CRL 検査を WebSEAL で実行できるようになります。CRL 検査のパフォーマンスを高めるには、特定の認証局 (CA) からの CRL をキャッシュに入れます。以後の CRL 検査は、このリストのキャッシュ・バージョンに照らして行われます。

このセクションで説明する 2 つの構成ファイル・スタンザ・エントリーの設定は、GSKit ユーティリティーに直接渡されます。GSKit の機能についての詳細は、GSKit の資料を参照してください。

キャッシュ・エントリーの最大数の設定

gsk-cr1-cache-size スタンザ・エントリーは、GSKit CRL キャッシュ内のエントリーの最大数を指定します。各エントリー は、特定の認証局に関連した 1 つの CRL 全体を表します。デフォルトの設定は「0」です。キャッシュを活動化するには、この値を「0」より大きくする必要があります。

注: CRL エントリーは大量のメモリーを使用することがあります。したがって、**gsk-cr1-cache-size** には最小値を指定するようにしてください。

```
[ssl]
gsk-cr1-cache-size = 0
```

GSKit キャッシュ存続時間タイムアウト値の設定

gsk-crl-cache-entry-lifetime スタンザ・エントリーは、GSKit CRL キャッシュ内のすべてのエントリーの存続時間タイムアウト値を指定します。この値は秒数で表され、範囲は 0 から 86400 秒です。デフォルト値は 0 です。

注: WebSEAL と GSKit のどちらにも最大値の制限はありませんが、値は 64 ビットの整数である必要があります。

```
[ss1]
gsk-crl-cache-entry-lifetime = 0
```

CRL キャッシュの使用可能化

gsk-crl-cache-size と **gsk-crl-cache-entry-lifetime** スタンザ・エントリーが両方とも「0」(デフォルト) に設定されている場合は、CRL キャッシングは使用不可になります。

キャッシュを使用可能にするには、**gsk-crl-cache-size** と **gsk-crl-cache-entry-lifetime** のいずれか、または両方の設定を 0 以外の値に変更します。両方の値が 0 の場合、キャッシュは使用不可になります。キャッシュは、これらのスタンザ・エントリーのいずれか、または両方が 0 以外の値に構成されている場合に使用可能になります。

どちらかの構成エントリーの値が 0 になっているときに、他方のエントリーが 0 以外の値になっている場合、GSKit は値が 0 のエントリーに自動的にデフォルト値を割り当てます。GSKit は以下のプロセスを使用します。

- **gsk-crl-cache-entry-lifetime** が 0 以外の値で構成されていても、**gsk-crl-cache-size** が 0 として構成されている場合は、CRL キャッシュが使用可能です。この場合、GSKit は **gsk-crl-cache-size** に次のデフォルト値を使用します。
 - **gsk-crl-cache-size** = 50
- **gsk-crl-cache-size** が 0 以外の値で構成されていても、**gsk-crl-cache-entry-lifetime** が 0 として構成されている場合は、CRL キャッシュが使用可能です。この場合、GSKit は **gsk-crl-cache-entry-lifetime** に次のデフォルト値を使用します。
 - **gsk-crl-cache-entry-lifetime** = 43200

証明書の CDP により CRL の HTTP ソースが指定されている場合、WebSEAL は **gsk-crl-cache-size** および **gsk-crl-cache-entry-lifetime** 構成設定を使用しません。HTTP ソースからの CRL がキャッシュされることはありません。OCSP がオプションではなく、サイズの大きな CRL を HTTP を使用して読み込む必要がある場合は、GSKit 環境変数 **GSK_HTTP_CDP_MAX_RESPONSE_SIZE** を使用できます。

SSL 接続用 WebSEAL テスト証明書の使用

クライアント・サイドの証明書認証は、Secure Socket Layer (SSL) 接続を介して行われなければなりません。SSL 接続は、証明書認証プロセスより前に確立されます。SSL 接続は、クライアントが HTTPS を介してリソースにアクセスしようとすると、確立されます。リソースが認証済みアクセスを必要としないときは、クライアントは、WebSEAL サーバーと、SSL セッションについて折衝します。SSL セッ

セッションは、クライアントとサーバー (WebSEAL) が相互に証明書を検査し、署名権限の妥当性を受け入れると確立されます。

新しい WebSEAL サーバーに対する SSL セッションの確立を使用可能にするために、WebSEAL には自己署名テスト・サーバー証明書が用意されています。WebSEAL は、この自己署名証明書をクライアントに提示することができます。クライアントがこの証明書を受け入れる場合、SSL セッションが確立されます。

このテスト証明書は、WebSEAL サーバーが永続使用するには適切ではありません。このテスト証明書は、WebSEAL が、SSL 使用可能ブラウザの要求に応答できるようにしますが、これをブラウザで検査することはできません。検査できない理由は、ブラウザに適切なルート認証局 (CA) 証明書がないからです。これは、ブラウザが、ルート CA 証明書がない自己署名証明書を受け取る場合も同じです。このデフォルト証明書の秘密鍵は、あらゆる WebSEAL 配布に含まれているため、この証明書では、真に安全な通信は約束していません。

SSL におけるセキュア通信を確実にするためには、WebSEAL アドミニストレータは、トラステッド認証局 (CA) から固有のサイト・サーバー証明書を取得しておく必要があります。LMI を使用して、CA に送信される証明書要求を生成することができます。LMI を使用して、新規のサイト証明書をインストールし、それにラベルを付けることもできます。

この証明書をアクティブな WebSEAL サーバー・サイド証明書として指定するには、WebSEAL 構成ファイルの **[ssl]** スタンザ内の **webseal-cert-keyfile-label** スタンザ・エントリーを使用します (これを設定すると、鍵格納ファイル・データベース内で「デフォルト」として指定されている証明書はオーバーライドされます)。

他のシナリオ (例えば相互認証 junction) のために別の証明書が必要な場合は、LMI を使用して、必要な追加の証明書の作成、インストール、およびラベル付けができます。405 ページの『WebSEAL 鍵データベース・ファイルの構成』を参照してください。

また、証明書の妥当性検査には、証明書取り消しリスト (CRL) の検査が含まれていることを確認することが重要です。該当する CRL にアクセスできる十分な許可を持つ LDAP ユーザーとして、該当 LDAP サーバーにアクセスできるように WebSEAL を構成します。以下の構成ファイル・エントリーに値を指定してください。

```
[ssl]
cr1-ldap-server
cr1-ldap-server-port
cr1-ldap-user
cr1-ldap-user-password
```

WebSEAL は、CRL をキャッシュに入れるように構成できます。キャッシュを構成するには、以下の構成ファイル・エントリーに値を指定してください。

```
[ssl]
gsk-cr1-cache-size
gsk-cr1-cache-entry-lifetime
```

CRL のアクセスおよび処理に影響する値の設定手順 (キャッシュ設定値の有効範囲を含む) については、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」に説明があります。

410 ページの『CRL キャッシュの構成』も参照してください。

第 7 部 標準 WebSEAL junction

第 23 章 標準 WebSEAL junction

この章では、標準 WebSEAL junction の構成について説明します。

ほとんどの標準 Junction オプションは、仮想ホスト Junction でもサポートされています。仮想ホスト junction については、547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』で説明しています。

この章で扱うトピックは以下のとおりです。

- 『WebSEAL junction の概要』
- 419 ページの『Web Portal Manager を使用した Junction の管理』
- 420 ページの『ローカル管理インターフェースでの junction 管理』
- 421 ページの『pdadmin ユーティリティーを使用した junction の管理』
- 422 ページの『標準 WebSEAL junction の構成』
- 427 ページの『透過パス Junction』
- 430 ページの『WebSEAL junction を使用するときの技術上の注意点』
- 434 ページの『バックエンドのサーバー Web スペースの生成方法 (query_contents)』

WebSEAL junction の概要

WebSEAL junction とは、フロントエンド WebSEAL サーバーとバックエンド Web アプリケーション・サーバーの間の HTTP または HTTPS 接続のことです。

Junction は、バックエンド・サーバーの Web スペースを WebSEAL サーバーの Web スペースと論理的に結合するためのものであり、これにより Web オブジェクト・スペース全体の統一化されたビューが得られます。

Junction により、WebSEAL がバックエンド・サーバーに代わって、保護サービスを提供できます。WebSEAL は、すべてのリソース要求について認証検査と許可検査を行ってから、junction を介してバックエンド・サーバーに要求を渡します。また、junction を利用することで、クライアントと junction 先バックエンド・アプリケーションの間の多様なシングル・サインオン・ソリューションが可能になります。

WebSEAL Junction の作成には、**pdadmin** コマンド行ユーティリティー、Web ポータル・マネージャー、または LMI を使用できます。

このセクションは、以下のトピックで構成されています。

- 418 ページの『junction タイプ』
- 418 ページの『大まかなアクセス・コントロールの適用: 要約』
- 418 ページの『きめの細かいアクセス・コントロールの適用: 要約』
- 419 ページの『WebSEAL junction に関する他の参照情報』

junction タイプ

以下の WebSEAL junction タイプを作成することができます。

- TCP 接続を介した WebSEAL からバックエンド・サーバーへ
- HTTP プロキシ・サーバーを使用して TCP 接続を介した WebSEAL からバックエンド・サーバーへ
- SSL 接続を介した WebSEAL からバックエンド・サーバーへ
- HTTPS プロキシ・サーバーを使用して SSL 接続を介した WebSEAL からバックエンド・サーバーへ
- SSL 接続を介した WebSEAL から WebSEAL へ
- mutual junction を介した WebSEAL からバックエンド・サーバーへ

いずれの junction の作成時にも、以下の 2 つの事項について注意しなければなりません。

1. WebSEAL オブジェクト・スペース内の Web アプリケーション・サーバーの junction (マウント) の場所を決めます。
2. Junction のタイプを選択します。

大まかなアクセス・コントロールの適用: 要約

このタスクについて

junction オブジェクトに配置された保護 ACL は、バックエンド・リソース全体を大まかにコントロールします。ACL は、junction を介してアクセスされるすべての個々のリソースに全般的な一連のアクセス許可を与えます。

手順

1. **pdadmin** ユーティリティーおよび Web Portal Manager を使用して、WebSEAL とバックエンド・サーバーの間の junction を作成します。
2. 適切な ACL ポリシーを junction ポイントに配置して、バックエンド・サーバーに対する大まかなコントロールを実施します。

きめの細かいアクセス・コントロールの適用: 要約

junction オブジェクトに配置された保護 ACL は、バックエンド・リソース全体を大まかにコントロールします。ACL は、junction を介してアクセスされるすべての個々のリソースに全般的な一連のアクセス許可を与えます。

このタスクについて

また、個々のリソース・オブジェクトまたはオブジェクト・グループに明示的に ACL を配置して、junction を介してアクセスするリソースにきめの細かい保護を提供することもできます。WebSEAL は、バックエンドのファイル・システムを自動的に認識して、理解することはできません。**query_contents** と呼ばれる特殊アプリケーションを使用して、WebSEAL にバックエンドのオブジェクト・スペースに関する情報を通知しなければなりません。このアプリケーションは、バックエンドの Web スペースを調べ、WebSEAL に対して構造とコンテンツを報告するものです。

手順

1. **pdadmin** ユーティリティーおよび Web Portal Manager を使用して、WebSEAL とバックエンド・サーバーの間の **junction** を作成します。
2. **query_contents** プログラムをバックエンド・サーバーにコピーします。
3. **query_contents** プログラムで示されたオブジェクト・スペースの適切なオブジェクトに ACL ポリシーを適用します。

WebSEAL junction に関する他の参照情報

WebSEAL junction の概念についての概要は、16 ページの『標準 WebSEAL Junction』を参照してください。

Junction の拡張オプションについての詳細は、443 ページの『第 24 章 拡張 junction の構成』を参照してください。

機能カテゴリー別の junction コマンド・オプションの要約は、531 ページの『第 28 章 コマンド・オプションの要約: 標準 junction』を参照してください。

Web Portal Manager を使用した Junction の管理

Security Access Manager Web Portal Manager のグラフィカル・ユーザー・インターフェースを使用して、junction を作成、リスト、および削除できます。

このセクションは、以下のトピックで構成されています。

- 『Web Portal Manager を使用した junction の作成』
- 420 ページの『Web Portal Manager を使用した junction のリスト』
- 420 ページの『Web Portal Manager を使用した junction の削除』

Web Portal Manager を使用した junction の作成

このタスクについて

Web Portal Manager を使用して junction を作成します。

手順

1. ドメインにログインします。
2. 「WebSEAL」→「Junction の作成」をクリックします。
3. 「WebSEAL サーバー名」インスタンスを選択します。
4. 「Junction ポイント」の名前を入力します。
5. junction の「タイプ」を選択します。
サポートされているタイプのオンライン・ヘルプを表示するには、右上の隅の ? アイコンをクリックします。
6. 選択した junction タイプに必要な構成情報を指定します。Junction タイプに基づいて Web Portal Manager ウィンドウのフィールドが変わることに注意してください。以下のセクションで、該当するチェック・ボックスを選択し、要求された値を入力してください。

- サーバー情報
- クライアント識別ヘッダー
- 一般オプション
- 基本認証

それぞれの構成セクションのオンライン・ヘルプについては、右上の隅の ? アイコンをクリックしてください。

7. WebSphere への LTPA を使用してシングル・サインオンを構成するときは、「**WebSphere シングル・サインオン**」セクションに値を指定してください。
8. 複数の junction を構成するときは、「**Junction フェアネス**」セクションに値を指定することによって、ワーカー・スレッドの割り振りをコントロールできます。注: 「ソフト制限」のデフォルト値は 90% です。Junction フェアネスについて詳しくは、58 ページの『ワーカー・スレッドの割り振り』を参照してください。

Web Portal Manager を使用した junction のリスト

junction をリストするには Web Portal Manager を使用します。

手順

1. ドメインにログインします。
2. 「WebSEAL」→「**Junction のリスト**」をクリックします。
3. 「WebSEAL サーバー名」インスタンスを選択します。
4. 「**Junction の表示**」をクリックします。

Web Portal Manager を使用した junction の削除

このタスクについて

Web Portal Manager を使用して、構成済みの 1 つ以上の junction を削除するには、次のようにします。

手順

1. ドメインにログインします。
2. 「WebSEAL」→「**Junction のリスト**」をクリックします。
3. 「WebSEAL サーバー名」インスタンスを選択します。
4. 「**Junction の表示**」をクリックします。
5. Junction 名の横のチェック・ボックスを選択し、「**削除**」をクリックします。
一時に複数の junction を削除することができます。

ローカル管理インターフェースでの junction 管理

LMI を使用して、標準 junction と仮想 junction を管理できます。LMI の junction 管理ページにアクセスするには、「リバース・プロキシ」管理ページに移動して、適切な WebSEAL インスタンスを選択します。「管理」>「**Junction 管理**」をクリックして、選択したインスタンスの「Junction 管理」ページを開きます。

「Junction の管理」ページでは、次のタスクを実行できます。

- 標準および仮想の junction のリストを取得する。
- 1 つの標準 junction または仮想 junction のパラメーターを取得する。
- 標準または仮想の junction を削除する。
- 標準または仮想の junction を作成する。
- バックエンド・サーバーを既存の標準 junction または仮想 junction に追加する。

LMI を使用してこれらの junction 管理タスクを実行する方法について詳しくは、「*IBM Security Web Gateway Appliance 管理ガイド*」を参照してください。

pdadmin ユーティリティーを使用した junction の管理

このタスクについて

注: また、Security Access Manager Web Portal Manager のグラフィカル・ユーザー・インターフェースを使用して、junction を作成することもできます。詳細については、419 ページの『Web Portal Manager を使用した junction の作成』を参照してください。

pdadmin ユーティリティーを使用する前に、**sec_master** などの管理許可を持つユーザーとしてセキュア・ドメインにログインする必要があります。

例:

UNIX または Linux:

```
# pdadmin
pdadmin> login
Enter User ID: sec_master
Enter Password:
pdadmin>
```

WebSEAL junction を作成するには、**pdadmin server task create** コマンドを使用します。

```
pdadmin> server task instance_name-webseald-host_name create options
```

例えば、単一の WebSEAL インスタンスの構成名が **web1** で、**www.example.com** という名前のホストにインストールされている場合、完全なサーバー名は次のようになります。

```
web1-webseald-www.example.com
```

完全なサーバー名を正しい形式で表示するには、**pdadmin server list** コマンドを使用します。

```
pdadmin> server list
web1-webseald-www.example.com
```

詳しくは、707 ページの『付録 B. コマンド・リファレンス』または「*IBM Security Access Manager for Web: Command Reference*」の **pdadmin server task create** の参照ページを参照してください。

junction データベースのインポートとエクスポート

junction データベースを WebSEAL サーバー間でコピーすることによって、WebSEAL サーバー自体を停止することなく、junction データベースのレプリカを生成できます。この操作を実行するには、junction データベースをサーバー上の指定されたファイルにエクスポートした後、同じファイルから別のサーバーにインポートします。

次のサーバー・タスク・コマンドを実行すると、既存の junction データベースが指定されたファイルにエクスポートされます。

```
pdadmin> server task server jdb export file path=file
```

次のサーバー・タスク・コマンドを実行すると、junction データベースが指定されたファイルからインポートされます。

```
pdadmin> server task server jdb import file path=file
```

これらのコマンドは **pdadmin** または Web Portal Manager を使用して実行できます。2 つのコマンドで指定されたファイルは、コマンドの実行元のファイル・システムでなく、ターゲットの WebSEAL サーバーのファイル・システム上に存在します。

エクスポートされたテキストまたは XML ファイルを編集して、サーバーの IP アドレスなどを変更することができます。

[jdb-cmd:replace] スタンザに、**import** コマンドの実行中に junction 属性値の再マップを制御するルールを追加できます。

junction 定義をインポートする前に、junction に事前のセットアップが必要になることもあります。例えば、junction を作成する前に、必須の SSL 証明書を SSL junction の WebSEAL 鍵データベースにインポートしなければならないことがあります。

標準 WebSEAL junction の構成

このセクションは、以下のトピックで構成されています。

- 『pdadmin server task create コマンド』
- 423 ページの『TCP タイプの標準 junction の作成』
- 423 ページの『SSL タイプの標準 junction の作成』
- 426 ページの『標準 junction への複数のバックエンド・サーバーの追加』

pdadmin server task create コマンド

WebSEAL は、WebSEAL とバックエンド Web アプリケーション・サーバーとの間で、標準の非セキュア TCP (HTTP) junction およびセキュア SSL (HTTPS) junction の両方をサポートします。

WebSEAL とバックエンド・サーバーの間の junction は、クライアント (ブラウザ) と WebSEAL サーバーとの間の接続タイプ (およびそのセキュリティ・レベル) とは別個のものです。

pdadmin server task create を使用して基本 WebSEAL junction を作成するために必要な必須コマンド・オプションには、以下のものがあります。

- バックエンド・アプリケーション・サーバーのホスト名 (**-h** オプション)
- Junction タイプ: **tcp**、**ssl**、**tcpproxy**、**sslproxy**、**local** (**-t** オプション)
- Junction ポイント (マウント・ポイント)

コマンド構文 (1 行で入力します)

```
pdadmin> server task instance_name-webseald-host-name  
create -t type -h host_name jct_point
```

例:

```
pdadmin> server task web1-webseald-cruz create -t tcp -h doc.ibm.com /pubs
```

注: **-h** オプションに引数を指定するときは、常に、バックエンド・サーバーの完全修飾ドメイン名を使用します。

TCP タイプの標準 junction の作成

このタスクについて

TCP 接続による WebSEAL junction は、junction の基本プロパティは用意していますが、junction を介したセキュア接続は用意していません。

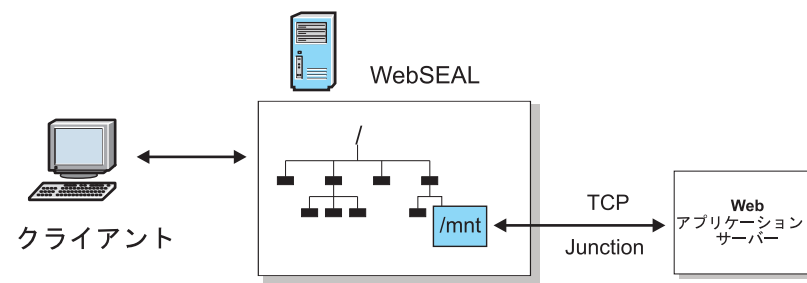


図 29. 非セキュアの TCP (HTTP) junction

手順

セキュア TCP junction を作成し、初期サーバーを追加するには、以下のように **-t tcp** オプションを指定して **create** コマンドを使用 (1 行で入力します) します。

```
pdadmin> server task instance_name-webseald-host_name create -t tcp  
-h host-name [-p port] jct-point
```

TCP junction のデフォルト・ポート値 (ポート値が指定されていない場合) は **80** です。

SSL タイプの標準 junction の作成

このタスクについて

SSL junction は、TCP junction とまったく同様に機能するだけでなく、さらに、WebSEAL とバックエンド・サーバーの間の通信がすべて暗号化されるという付加

価値が加わります。

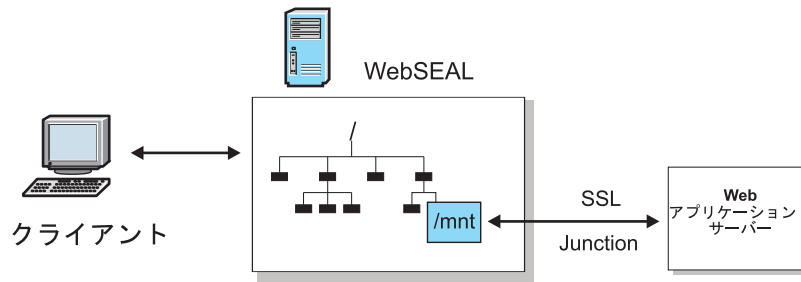


図 30. セキュア SSL (HTTPS) junction

SSL junction を使用することにより、エンドツーエンドのブラウザー対アプリケーション・トランザクションのセキュリティを達成できます。SSL を使用することにより、クライアントから WebSEAL への通信と、WebSEAL からバックエンド・サーバーへの通信を保護することができます。SSL junction を使用する場合は、バックエンド・サーバーが HTTPS 使用可能でなければなりません。

手順

セキュア SSL junction を作成し、初期サーバーを追加するには、以下のように **-t ssl** オプションを指定して **create** コマンドを使用 (1 行で入力します) します。

```
pdadmin> server task instance_name-webseald-host-name create -t ssl
-h host_name [-p port] jct_point
```

SSL junction のデフォルト・ポート値 (ポート値が指定されていない場合) は **443** です。

SSL ベース標準 junction の構成について詳しくは、425 ページの『SSL ベース標準 junction』を参照してください。

mutual junction の作成

このタスクについて

mutual junction には、HTTP または HTTPS (junction 要求の受信に使用された通信プロトコルによって決まる) を介して junction 要求を送信する機能があります。

HTTP を介して mutual junction に着信した要求は、HTTP を介して junction 先サーバーに送信されます。HTTPS を介して着信した要求は、HTTPS を介して junction 先サーバーに送信されます。

手順

mutual junction を作成し、初期サーバーを追加するには、**-t mutual** オプションを指定して **create** コマンドを使用します。HTTP ポートの場合は **-p** オプションを、HTTPS ポートの場合は **-P** オプションを使用します。同様に、HTTP 要求の仮想ホスト名を指定するには **-v** オプションを、HTTPS 要求の仮想ホスト名を指定するには **-V** オプションを使用します。例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task instance_name-webseald-host-name create -t mutual  
-h host_name [-p HTTP_port] [-P HTTPS_port] [-v HTTP_virtual_host_name]  
[-V HTTPS_virtual_host_name]jct_point
```

mutual junction のデフォルト HTTP ポート値 (ポート値が指定されていない場合) は **80** です。

mutual junction のデフォルト HTTPS ポート値 (ポート値が指定されていない場合) は **443** です。

SSL ベース標準 junction

バックエンド・サーバー証明書の検証

クライアントが、バックエンド・サーバー上のリソースに対して要求を出すと、セキュリティ・サーバーとして機能する WebSEAL が、クライアントに代わってその要求を実行します。SSL プロトコルでは、バックエンド・サーバーに要求が出されたときに、そのサーバーがそのものであることを、サーバー・サイド証明書によって証明しなければならないと指定しています。

WebSEAL は、この証明書をバックエンド・サーバーから受け取ると、その証明書データベースに保管されているルート CA 証明書のリストとその証明書を突き合わせて、認証性を検証しなければなりません。

Security Access Manager は、SSL の IBM Global Security Kit (GSKit) インプリメンテーションを使用します。LMI を使用して、バックエンド・サーバー証明書に署名した CA のルート証明書を WebSEAL 証明書鍵ファイル (pdsrv.kdb) に追加することができます。

SSL junction の例

SSL を介した junction ポイント /sales の junction ホスト **sales.ibm.com** の場合は、次のようになります (1 行で入力します)。

```
pdadmin> server task web1-webseald-cruz create -t ssl -h  
sales.ibm.com /sales
```

注: この販売の例では、**-t ssl** オプションでデフォルト・ポート 443 が指示されています。

SSL を介した junction ポイント /travel のポート **4443** の junction ホスト **travel.ibm.com** の場合は、次のようになります (1 行で入力します)。

```
pdadmin> server task web1-webseald-cruz create -t ssl -p 4443  
-h travel.ibm.com /travel
```

Junction の SSL プロトコル・バージョンの使用不可

このタスクについて

必要に応じて、junction 接続の 1 つ以上の SSL プロトコル・バージョンを使用不可にすることができます。デフォルトでは、SSL v2 は使用不可です。その他のサポートされているすべての SSL バージョンは使用可能になっています。WebSEAL 構成ファイルには、デフォルトで、以下のエントリが用意されています。

```
[junction]
disable-ssl-v2 = yes
disable-ssl-v3 = no
disable-tls-v1 = no
disable-tls-v11 = no
disable-tls-v12 = no
```

手順

Junction の SSL プロトコル・バージョンを使用不可にするには、対応するエントリを `yes` に設定します。

標準 junction への複数のバックエンド・サーバーの追加

このタスクについて

431 ページの『同一 junction への複数のバックエンド・サーバーの追加』を参照してください。

ローカル・タイプの標準 Junction

ローカル・タイプの junction (`-t local`) は、WebSEAL サーバーのローカルに配置されている特定のコンテンツのマウント・ポイントです。Junction されたリモート・サーバーのコンテンツと同様に、ローカル junction のコンテンツは、WebSEAL の統一された保護オブジェクト・スペース・ビューに組み込まれます。

ローカル・タイプの junction に用いる junction オプションは、以下のとおりです。

表 34. ローカル・タイプの junction オプション

オプション	説明
<code>-t type</code>	junction のタイプ (local)。
<code>-f</code>	既存の junction を強制的に置き換えます。 453 ページの『新規 junction の強制適用』を参照してください。
<code>-l percent-value</code>	ワーカー・スレッドの消費量のソフト限界を定義します。
<code>-L percent-value</code>	ワーカー・スレッドの消費量のハード限界を定義します。

ローカル junction の無効化

ローカル junction 機能を無効にして、WebSEAL インスタンスがローカルに保管されたページを提供できないようにします。

ローカル junction 機能を無効にするには、WebSEAL 構成ファイルの `[junction]` スタンザの `disable-local-junctions` エントリを `yes` に設定します。

```
[junction]
disable-local-junctions = yes
```

disable-local-junctions 構成項目を有効にすると、新しいローカル junction は作成されません。既存のローカル junction が WebSEAL インスタンス内にある場合、それらの junction はインスタンスの始動時にロードされません。

透過パス Junction

このセクションは、以下のトピックで構成されています。

- 『標準 WebSEAL junction のフィルター操作の概念』
- 428 ページの『透過パス Junction の概念』
- 429 ページの『透過パス Junction の構成』
- 429 ページの『透過パス Junction の例』

標準 WebSEAL junction のフィルター操作の概念

標準的な junction の構成では、構成済み junction は特定のバックエンドのホスト・マシンを表します。junction とその名前は、WebSEAL 保護オブジェクト・スペースのサブディレクトリーとして表されます。

pdadmin server task コマンドを使用して、バックエンド・サーバー pubs.ibm.com に junction (/jct) を作成する例を次に示します。

```
pdadmin> server task web1-webseald-www.cruz.com create -t tcp -h pubs.ibm.com /jct
```

/jct という名前のサブディレクトリー が、WebSEAL オブジェクト・スペースに作成されます。この junction のマウント・ポイントは、バックエンド・サーバー pubs.ibm.com です。

pubs.ibm.com から戻される応答ページには、次のように、このページの HTML のリンク (URL) が含まれています。

```
http://pubs.ibm.com/docs/readme.html
```

標準的な junction に対する WebSEAL の標準フィルター・メカニズムでは、応答ページの HTML が構文解析され、このバックエンド・サーバー用に構成されている junction 名がこのリンクに追加されます。また、URL の元の絶対式はサーバー相対式に変更されます。この時点で、ユーザーには次のようなリンクが表示されます。

```
/jct/docs/readme.html
```

注: rewrite-absolute-with-absolute が「yes」に設定されている場合、リンクは次のように表示されます。

```
http://www.cruz.com/jct/docs/readme.html
```

488 ページの『rewrite-absolute-with-absolute オプションの構成』を参照してください。

ここで、リンクをクリックしてバックエンド・リソース (readme.html) にアクセスします。

Junction 名を表す URL の部分 (/jct) は、WebSEAL によってバックエンド・サーバー pubs.ibm.com にマップされます。設計により、junction は、バックエンド・サ

サーバーの文書スペースの **root** を指します。URL で Junction 名に続くすべてのパス式は、サーバーの **root** ロケーションからリソースまでのパスを表します。

この例の結論として、WebSEAL が正常に次の場所にリソースを見つけます。

<http://pubs.ibm.com/docs/readme.html>

最終的には junction 名はパスから除去され、URL は応答ページに最初に表示されていたとおりになります。

透過パス Junction の概念

標準 WebSEAL junction では、WebSEAL が受信した要求の URL にこの junction の ID が含まれる場合のみ、junction バックエンド・サーバー上のリソースへのリンクが成功します。Junction は、デフォルトおよびオプションのフィルター操作ソリューションを使用して、HTML 応答ページにある URL が WebSEAL の単一のホスト文書スペースの一部として表示される場合、正しく表示されるようにします。475 ページの『第 25 章 Junction リソースへの URL の変更』を参照してください。

フィルター操作ソリューションでは、以下の 3 つの URL の部分について考慮しなければなりません。

- プロトコル
- ホスト名: ポート
- パス

URL の 3 つの部分のうち、パスは、フィルター操作にとって最も問題が多く、また最も制約される部分です。透過パス Junction オプション (**-x**) を指定すると、標準 junction のメカニズムにおいて、URL のパスの部分をフィルターする必要がないようインプリメントされます。

透過パス Junction には、重要な要件があります。その要件とは、構成される junction 名は、バックエンド・サーバーの文書スペースの **root** の下のサブディレクトリー名と一致していなければならないというものです。この junction を介してアクセスされるすべてのリソースは、このサブディレクトリーの下に配置する必要があります。透過パス Junction 名は、バックエンド・サーバー上の実際のサブディレクトリーの名前を表しています。

透過パス Junction は、junction 名が、URL パスに追加されるのではなく、バックエンド・アプリケーション上に既に存在するパスに基づいている以外は標準 junction とまったく同じです。透過パス Junction によって、WebSEAL は、パスに追加された junction 名ではなく、バックエンド・サーバー・リソースの URL パスに基づいた junction に、要求をルーティングすることができます。

例えば、構成される junction 名が /docs の場合、この junction でコントロールされるすべてのリソースは、バックエンド・サーバー上の /docs というサブディレクトリーの下になければなりません。

透過パス Junction メカニズムを使用すると、この junction によって保護されているリソースへのリンクのパス部分が WebSEAL によってフィルター処理されるのを防ぎます。これで junction 名はリソースの位置を示す実際のパス式の一部になり、フ

フィルター操作を必要としなくなります。透過パスのオプションなしで作成された junction の場合、junction 名が URL のパス部分に追加されたり、そこから除去されたりすることはありません。

WebSEAL はネストされたパスをサポートしています。例えば、次の 3 つの junction はすべて有効であり、同じ WebSEAL システム上に作成できます。

```
/financing/tools  
/financing/tools/gars  
/financing/tools/gars/custom
```

WebSEAL 内のパターン・マッチングは十分な精度があるため、最も「具体的な」junction (この例では /financing/tools/gars/custom) へのマッピングが最初に行われます。

透過パス Junction の構成

このタスクについて

透過パス Junction を構成するには、次のようにします。

手順

1. **pdadmin server task** コマンド (または Web Portal Manager) を使用して透過パス Junction を作成します。**-x** オプションを含める以外は、標準の WebSEAL junction の場合と同様に junction を作成してください。次の例では、junction 名は /files/docs です (1 行で入力します)。`pdadmin> server task web1-webseald-www.cruz.com create -t tcp -x-h pubs.ibm.com /files/docs`
2. この junction で保護されているすべてのリソースがバックエンド・サーバー (この例では pubs.ibm.com) の /files/docs/ というサブディレクトリの下に含まれていることを確認してください。

タスクの結果

Junction 名 (およびバックエンド・サーバー上の関連サブディレクトリ) は、WebSEAL 環境にある他のすべての保護サーバー間で固有でなければなりません。

透過パス Junction の例

pdadmin コマンドを使用して (1 行で入力します)、バックエンド・サーバー pubs.ibm.com への透過パス Junction (/docs) を作成する例を次に示します。

```
pdadmin> server task web1-webseald-www.cruz.com create -t tcp -x  
-h pubs.ibm.com /docs
```

クライアントから (WebSEAL プロキシ・サーバーを介して) バックエンド・リソースへの要求が出されると、pubs.ibm.com から応答ページが返されます。この応答ページには、以下のような、このページの HTML のリンク (URL) が含まれています。

```
http://pubs.ibm.com/docs/readme.html
```

WebSEAL の標準 junction フィルター・メカニズムによって、応答ページの HTML が構文解析され、URL の元の絶対式をサーバー相対式に変更する修正がこのリンク

に加えられます。ただし、透過パス Junction (-x) であるため、パスはフィルター操作されません。この時点で、ユーザーには次のようなリンクが表示されます。

`/docs/readme.html`

注: **rewrite-absolute-with-absolute** が「yes」に設定されている場合、リンクは次のように表示されます。

`http://www.cruz.com/docs/readme.html`

488 ページの『rewrite-absolute-with-absolute オプションの構成』を参照してください。

ここで、リンクをクリックしてバックエンド・リソース (`readme.html`) にアクセスします。

Junction 名を表す URL の部分 (`/docs`) が、バックエンド・サーバー `pubs.ibm.com` のサブディレクトリー `/docs` に関連付けられていることが WebSEAL によって認識されます。

この例の結論として、WebSEAL が正常に次の場所にリソースを見つけます。

`http://pubs.ibm.com/docs/readme.html`

透過パス Junction の利点には、次のものがあります。

- 同じバックエンド・サーバーへの複数の異なる透過パス Junction を作成して、そのサーバーのさまざまな領域 (サブディレクトリー) をポイントできます。
- 個々の透過パス Junction が、それぞれ異なる認証要件および ACL コントロールを処理できます。

WebSEAL junction を使用するときの技術上の注意点

このセクションは、以下のトピックで構成されています。

- 『WebSEAL junction を作成するためのガイドライン』
- 431 ページの『同一 junction への複数のバックエンド・サーバーの追加』
- 432 ページの『Junction を介して許可を実施する場合の例外条件』
- 432 ページの『junction を介した証明書認証』
- 433 ページの『ドメイン Cookie の取り扱い』
- 433 ページの『要求および応答でサポートされる HTTP バージョン』
- 434 ページの『Web Portal Manager を持った junction 先アプリケーション』

WebSEAL junction を作成するためのガイドライン

以下のガイドラインには、junction に関する「ルール」を要約してあります。

- Junction は、1 次 WebSEAL オブジェクト・スペース内のどこにでも追加できます。
- 同じマウント・ポイントに複数のレプリカ・バックエンド・サーバーを junction することができます。

同じ junction ポイントにマウントされる複数のレプリカ・バックエンド・サーバーは、同じタイプである必要があります。

- ACL ポリシーは、junction 経由でバックエンド Web サーバーに継承されます。
- バックエンド・サーバーにある HTML ページに、そのサーバーの Web スペース内にあるディレクトリへのサーバー相対 URL を持つプログラム (JavaScript やアプレットなど) が含まれている場合は、junction 名はそのディレクトリー名に一致してはなりません。例えば、バックエンド・サーバーにあるページに、/path/... という形式の URL を持つプログラムが含まれている場合は、/path を使用した junction 名は作成しないでください。
- 同じバックエンド・アプリケーション・サーバー/ポートをポイントする複数の WebSEAL junction を作成することは、セキュアな junction 構成ではありません。各 junction は固有の ACL でコントロールできます。セキュリティの低い ACL で保護されている junction により、セキュリティの高い ACL で保護されている別の junction のセキュリティまで下げることがあります。このような構成は、リソースへのアクセス・コントロールに不測の事態を招くおそれがあるので、Security Access Manager 構成ストラテジーとしてはサポートされていません。
- WebSEAL は、junction 間にまたがる HTTP/1.1 をサポートしています。

同一 junction への複数のバックエンド・サーバーの追加

このタスクについて

複数のレプリカ・バックエンド・サーバーを同じ junction ポイントに junction することにより、Security Access Manager で保護されているリソースのハイ・アベイラビリティをさらに高めることができます。同一ポイントにマウントできるレプリカ・サーバーの数に制限はありません。

- 同じ junction ポイントに追加される複数のバックエンド・サーバーは、同一の (ミラーリングされた) Web 文書スペースを持つレプリカ・サーバーである必要があります。
- 同じ junction ポイントに追加される複数のバックエンド・サーバーは、同じプロトコルを使用する必要があります。
- 同一 junction ポイントに異なるサーバーを追加することはできません。
- WebSEAL は、「一番すいている」アルゴリズムを使用して、最も要求接続数の少ないバックエンド・レプリカ・サーバーを判別し、新規の要求をそのサーバーに転送します。

手順

1. まず初期 junction を作成します。例:

```
pdadmin> server task web1-webseald-cruz create -t tcp -h server1 /sales
```

2. 次にバックエンド・サーバー・レプリカを追加します。例:

```
pdadmin> server task web1-webseald-cruz add -h server2 /sales
```

3. 1 次 Security Access Manager サーバーの Web スペースから、junction 先サーバーに属するページにアクセスしてみてください。これらのページに (許可に基づいて) 正常にアクセスできること、およびこれらのページが常に一貫した形で

表示されることが必要です。ページが検出できなかったり、時々変化したりすることがある場合は、そのページが正しく複製されなかったということを意味します。

4. 両方の複製サーバーの文書ツリー内に同一文書が存在し、それぞれの内容が同じであることを確認してください。

Junction を介して許可を実施する場合の例外条件

Security Access Manager 許可によっては、junction を介して実施できないものもあります。例えば、**x** 許可を持つ CGI スクリプト、または **l** 許可を持つディレクトリー・リストの実行はコントロールできません。WebSEAL には、例えば、バックエンド・サーバー上の要求されたオブジェクトが、CGI プログラム・ファイルなのか、動的ディレクトリー・リストなのか、通常の HTTP オブジェクトなのかを正確に判別する手段はありません。

CGI プログラムやディレクトリー・リストなど、複数の junction にわたるオブジェクトへのアクセスは、**r** 許可によってのみコントロールできます。

junction を介した証明書認証

インストール時には、WebSEAL は非デフォルトのテスト証明書を使用して構成されます。テスト証明書は、WebSEAL 構成ファイルの **[ssl]** スタンザ内の **webseal-cert-keyfile-label** スタンザ・エントリーにより、アクティブなサーバー・サイド証明書として指定されます。

junction バックエンド・アプリケーション・サーバーに対して、WebSEAL がクライアント・サイド証明書を使用して自身を識別する必要がある場合は、まず、ローカル管理インターフェース (LMI) を使用して、この証明書を作成、インストール、およびラベル付けする必要があります。次に、**-K key-label** オプションを使用して、junction を構成します。443 ページの『相互認証 SSL junction』を参照してください。

-K を指定して junction を構成しなかった場合は、GSKit は、鍵格納ファイル・データベースに含まれている「デフォルト」の証明書を自動的に送信することにより、相互認証を要する要求を取り扱います。必要としているのがこの応答ではない場合は、鍵ファイル・データベース (pdsrv.kdb、または別の junction 鍵ファイルが構成されている場合は junction 鍵ファイル) 内に「デフォルト」としてマークされた (アスタリスクが付いた) 証明書が存在しないようにする必要があります。

上記を要約すると以下ようになります。

- 必要なすべての証明書をラベル名で識別する。
- 鍵格納ファイル・データベース内の証明書には「デフォルト」のマークが付いていてはならない。
- WebSEAL サーバー・サイド証明書応答は、**webseal-cert-keyfile-label** スタンザ・エントリーを使用してコントロールする。
- WebSEAL クライアント・サイド証明書応答は、**-K junction** オプションを使用してコントロールする。

ドメイン Cookie の取り扱い

このタスクについて

WebSEAL が Cookie ヘッダー内のドメイン属性をどのように扱うかを制御するには、WebSEAL 構成ファイルの **[junction]** スタンザ内の **allow-backend-domain-cookies** スタンザ・エントリーを使用します。

このスタンザ・エントリーの値が「no」(デフォルト) に設定されている場合は、WebSEAL は、「テール・マッチング」を行って、ドメイン (Cookie ヘッダー内に属性として含まれている) が有効かどうかを確認します。Cookie ヘッダー内のドメインが有効である場合は、Cookie ヘッダーからドメイン属性を削除してから、Cookie がブラウザに送信されます。ドメイン属性のない Cookie を受け取った場合、ブラウザは、発信元サーバーに対してのみその Cookie を戻すことができます。「テール・マッチング」により Cookie ヘッダー内のドメインが無効であると判断された場合は、Cookie はブラウザに送られません。したがって、ブラウザから送り返す Cookie はないことになります。

```
[junction]
allow-backend-domain-cookies = no
```

このスタンザ・エントリーの値を「yes」に設定した場合は、WebSEAL は「テール・マッチング」を行わず、ドメイン属性値に関係なく、すべての Cookie をブラウザに送信することを許可します。ブラウザは、該当する 1 つまたは複数のサーバーに Cookie を戻すことができます。

```
[junction]
allow-backend-domain-cookies = yes
```

手順

調整した構成項目を **[junction:{junction_name}]** スタンザに追加することによって、特定の junction 用に **allow-backend-domain-cookies** 構成項目をカスタマイズできます。

{junction_name} は、標準 junction の junction ポイント (先頭の / 文字を含む)、または仮想ホスト junction の仮想ホスト・ラベルを表します。

要求および応答でサポートされる HTTP バージョン

HTTP/1.0 要求が junction 先バックエンド・サーバーに送られるのは、そのサーバーが状況として 400 (無効な要求) または 504 (サポートされない HTTP バージョン) を戻す場合か、クライアント・ブラウザが要求の中で HTTP/1.0 を指定している場合のみです。

その他の場合は、バックエンド・サーバーが HTTP/1.1 を受け入れるのであれば、WebSEAL は HTTP/1.1 要求を送ります。

しかし、WebSEAL が junction 先バックエンド・サーバーに HTTP/1.0 要求を送る (そしてそのバックエンド・サーバーが HTTP/1.0 応答を戻す) 場合でも、WebSEAL は常に HTTP/1.1 応答をクライアント・ブラウザに戻します。

Web Portal Manager を持った junction 先アプリケーション

問題: Web Portal Manager が絶対 URL またはサーバー相対 URL を Javascript に入れて送ります。これらのアドレスはブラウザによって正常に解決されず、パス名を完全にするために junction Cookie 情報が必要です。

ソリューション: Web Portal Manager を持ったアプリケーション・サーバーが WebSEAL に junction されている場合は、この junction を作成するときに、**-j** オプションを使用する必要があります。**-j** オプションによって提供されている junction Cookie を使用すると、ブラウザ (クライアント) は、Web Portal Manager に正常にコマンドを出すことができます。

-j オプションの使用に加えて、**-c iv_user, iv_creds** オプションも使用する必要があります。

バックエンドのサーバー Web スペースの生成方法 (query_contents)

このセクションは、以下のトピックで構成されています。

- 『query_contents の概要』
- 436 ページの 『query_contents コンポーネント』
- 437 ページの 『UNIX ベースの Web サーバーでの query_contents のインストールおよび構成』
- 438 ページの 『Windows ベースの Web サーバーでの query_contents のインストールおよび構成』
- 440 ページの 『query_contents の一般プロセス・フロー』
- 440 ページの 『query_contents プログラムの保護』

query_contents の概要

Security Access Manager セキュリティー・サービスを使用して、バックエンド・アプリケーションの Web サーバーのリソースを保護したい場合は、バックエンド Web スペースの内容についての情報を WebSEAL に提示する必要があります。

query_contents と呼ばれる CGI プログラムによって、この情報が提供されます。**query_contents** プログラムは、バックエンド Web スペースの内容を検索し、このインベントリ情報を WebSEAL に提供します。デフォルト・バージョンのプログラムは、WebSEAL のインストールに付属していますが、バックエンド Web サーバーに手動でインストールしなければなりません。利用できるプログラム・ファイル・タイプは、バックエンド・サーバーで UNIX を使用しているか、Windows を使用しているかによって異なります。

保護オブジェクト・スペースの junction に属している部分が、オブジェクト・スペース管理パネルで展開される場合は、常に、Web Portal Manager のオブジェクト・スペース・マネージャーによって、**query_contents** が自動的に実行されます。Web Portal Manager がバックエンド Web スペースの内容を認識すると、ユーザーはこの情報を表示し、該当するオブジェクトにポリシー・テンプレートを適用することができます。

pdadmin **object show** コマンドは **query_contents** プログラムを使用して Web スペースを参照し、管理者がポリシーを付加できるリソースを識別できるようにします。これにより、pdadmin および **query_contents** は、Security Access Manager オブジェクト・スペース内のオブジェクトへのアクセスを管理および制御するために使用されます。

WebSEAL には **query_contents** のソース・ファイル、サンプル構成ファイル、および HTML ヘルプ・ファイルが含まれています。管理者はこれらのファイルを使用して、**query_contents** を構成したり、必要に応じてその動作を変更したりできます。

カスタム **query_contents** プログラム

query_contents プログラムを実行すると、WebSEAL は単純な HTTP GET 要求をプログラムに送信して、単純な HTTP 応答と、要求された情報を含む応答本体を待ち受けます。**query_contents** プログラムは任意の言語で記述することができ、任意の HTTP 準拠アプリケーション・サーバーでホストすることができます。

プログラムの入力:

query_contents プログラムを呼び出すには、以下の構文を持つ HTTP GET 要求を使用します。

```
<URL>?dirlist=<resource root>
```

ここで、<URL> は **query_contents** プログラムの場所、<resource root> は検索対象のオブジェクト・スペースのルートです。

GET 要求では、X_QUERY_CONTENTS_URIENCODED という名前の HTTP Cookie が使用されます。この Cookie にはパスが URI エンコードされているかどうかを示す値 (yes または no) が含まれています。

以下の例は、WebSEAL が **query_contents** プログラムに送信する可能性のある GET 要求を示しています。

```
GET /appserver/cgi-bin/custom_query_contents.exe?dirlist=/directory
Cookie: X_QUERY_CONTENTS_URIENCODED=no
```

プログラムの出力:

query_contents プログラムは、UTF-8 エンコードされた応答を返す必要があります。この応答には以下のヘッダーが含まれていなければなりません。

```
content-type: text/plain;charset=utf8
```

応答の本体には、戻りコードを含む 1 行と、その後続くディレクトリーのリストが含まれている必要があります。リストでは、オブジェクトまたはコンテナ名を、1 行に 1 つずつ記述します。

```
<return-code>
<file or directory>
<file or directory>
<file or directory>
```

ここで、<return-code> は、以下の表の定義に従って 100、102、または 103 になります。

表 35. 戻りコード

照会状況	説明
QUERY_STATUS_OK = 100	ディレクトリー・リスト作成操作に成功しました。
QUERY_STATUS_NO_EXISTS = 102	照会されたディレクトリーが存在しません。
QUERY_STATUS_NOT_DIR = 103	照会されたディレクトリーがディレクトリーではありません。

応答内のオブジェクト名およびコンテナは、以下のルールに従う必要があります。

- コンテナの末尾は 2 つのスラッシュにします (/path/directory// など)。
- ディレクトリー・エントリーに句読点「.」および「..」を含めないでください。
- オブジェクトまたはコンテナ・データは UTF-8 エンコードされた後、URI エンコードされている必要があります。

query_contents コンポーネント

query_contents コンポーネントは、LMI を使用して入手できます。「セキュア・リバース・プロキシー設定 (Secure Reverse Proxy Settings)」>「ツール」>「照会サイトのコンテンツ」と進みます。

query_contents コンポーネントには、以下のものがあります。

ファイル	説明
Windows:	
query_contents.exe	Windows システム用の実行可能なメイン・プログラム。バックエンド Web サーバーの cgi-bin ディレクトリーにこのファイルをインストールします。
query_contents.cfg	Web サーバーの文書ルートを識別する Windows 構成ファイル。
query_contents.c	Windows システム用のソース・コード。ソースが提供されるのは、 query_contents の振る舞いを変更する必要がある場合です。ほとんどの場合、これは必要ありません。
UNIX:	
query_contents.sh	UNIX システム用の実行可能なメイン・プログラム。バックエンド Web サーバーの cgi-bin ディレクトリーにこのファイルをインストールします。

UNIX ベースの Web サーバーでの query_contents のインストールおよび構成

このタスクについて

以下の手順は、UNIX ベースのバックエンド Web サーバーでの **query_contents** のインストールを解説しています。

手順

1. LMI で「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「ツール」>「照会サイトのコンテンツ」にアクセスし、シェル・スクリプト **query_contents.sh** を探します。
2. バックエンド Web サーバーに CGI ディレクトリーが正しく構成されているか確認します。
3. テストのために、バックエンド Web サーバーの文書ルートに、有効な文書が存在しているか確認します。
4. バックエンド Web サーバー上で機能している **/cgi-bin** ディレクトリーに **query_contents.sh** をコピーします。Web サーバーの該当する文書を参照して、このディレクトリーの場所を確認してください。
5. バックエンド Web サーバーに **junction** を作成するときに、**-q location** オプションを使用することで、WebSEAL が使用する正しいファイル名として **query_contents.sh** を指定する必要があります。

-q location オプションと引数は、WebSEAL にファイルの正しい名前とそのファイルの検索場所を提供します。例えば、以下のようにします (1 行として入力し、他のオプションは追加で選択すると想定)。

```
pdadmin> server task default-webseald-www.example.com create -t tcp -h
host-name <...> -q /cgi-bin/query_contents.sh /junction-name
```

location 引数値は、**query_contents** プログラムを呼び出す実際の URL ストリングで使用されます。例:

```
http://unix-machine-name/cgi-bin/query_contents.sh?dirlist=/
```

6. 手でスクリプト・ファイルを編集し、正しい文書ルート・ディレクトリーを指定します。

DOCROOTDIR 変数の値 (デフォルトは **/usr/local/html**) をバックエンド Web サーバーの文書ルートに変更します。例 (IBM HTTP Server バージョン 6.0):

```
ADD_TO_ROOT=
DOCROOTDIR=/opt/IBMIHS/htdocs
#ADD_TO_ROOT="cgi-bin/"
```

注: ブランクの **ADD_TO_ROOT** 行は、変数の初期化で要求されます。

また、バックエンドの Web サーバーの **cgi-bin** ディレクトリーが Web サーバーのファイル・システム上の文書ルートにない場合は、**ADD_TO_ROOT** 変数の行をアンコメントします。例:

```
ADD_TO_ROOT=
DOCROOTDIR=/opt/IBMIHS/htdocs
ADD_TO_ROOT="cgi-bin/"
```

7. バックエンド Web サーバーの管理アカウントとして UNIX 実行ビットを設定します。

タスクの結果

構成のテスト (UNIX)

手順

1. バックエンド UNIX ベースのマシン上のコマンド・プロンプトで、以下のよう
に CGI ディレクトリーから **query_contents** プログラムを実行します。

```
# query_contents dirlist=/
```

次のような出力が表示されるはずです。

```
100  
index.html  
cgi-bin//  
pics//
```

番号 100 は、正常に実行されたことを示す戻り状況です。少なくとも番号 100 が最初 (かつ場合によっては唯一) の値であることを確認するのは非常に大切です。

代わりにエラー・コードが表示された場合は、文書ルート・エントリーが正しくない可能性があります。query_contents.sh ファイルの構成を確認し、文書ルートが正しく指定されていることを確認してください。

2. ブラウザーから、次の URL を入力します。

```
http://unix-machine-name/cgi-bin/query_contents.sh?dirlist=/
```

この URL によって、前のステップと同じ結果が戻されなければなりません。この結果が戻されない場合は、Web サーバーの CGI 構成に誤りがあります。サーバーの資料を参照して、問題を訂正します。

Windows ベースの Web サーバーでの query_contents のインストールおよび構成

このタスクについて

以下の手順は、Windows ベースのバックエンド Web サーバーでの **query_contents** のインストールを解説しています。

手順

1. LMI で、query_contents.exe という名前の実行可能プログラムと query_contents.cfg という名前の構成ファイルを見つけます。「**セキュア・リバース・プロキシー設定 (Secure Reverse Proxy Settings)**」>「**ツール**」>「**照会サイトのコンテンツ**」にアクセスします。
2. バックエンド Web サーバーに CGI ディレクトリーが正しく構成されているか確認します。
3. テストのために、バックエンド Web サーバーの文書ルートに、有効な文書が存在しているか確認します。

4. バックエンド Web サーバーの `cgi-bin` ディレクトリーに `query_contents.exe` をコピーします。Web サーバーの該当する文書を参照して、このディレクトリーの場所を確認してください。
5. 「Windows」ディレクトリーに `query_contents.cfg` をコピーします。

次の表に、このディレクトリーのデフォルト値を示します。

オペレーティング・システム	Windows ディレクトリー
Windows	c:%windows

6. バックエンド Web サーバーの文書ルート・ディレクトリーを正しく指定するように、`query_contents.cfg` ファイルを編集します。ファイルには、IBM HTTP Server (バージョン 6.0) のデフォルト値を含このファイルには、IBM HTTP Server (バージョン 6.0) のデフォルト値が指定されています。

```
[server]
docroot="C:/Program Files/IBM HTTP Server/htdocs/en_US"
```

7. バックエンド Web サーバーに `junction` を作成するときに、**-q location** オプションを使用することで、WebSEAL が使用する正しいファイル名として `query_contents.exe` を指定する必要があります。

-q location オプションと引数は、WebSEAL にファイルの正しい名前とそのファイルの検索場所を提供します。例えば、以下のようにします (1 行として入力し、他のオプションは追加で選択すると想定)。

```
pdadmin> server task default-webseald-www.example.com create -t tcp -h
host-name <...> -q /cgi-bin/query_contents.exe %junction-name
```

location 引数値は、**query_contents** プログラムを呼び出す実際の URL ストリングで使用されます。例:

```
http://windows-machine-name/cgi-bin/query_contents.exe?dirlist=/
```

構成のテスト (Windows)

手順

1. Windows マシン上の MS-DOS プロンプトに従って、以下のように CGI ディレクトリーから **query_contents** プログラムを実行します。

```
MSDOS> query_contents dirlist=/
```

次のような出力が表示されるはずです。

```
100
index.html
cgi-bin//
pics//
```

番号 100 は、正常に実行されたことを示す戻り状況です。少なくとも番号 100 が最初 (かつ場合によっては唯一) の値であることを確認するのは非常に大切です。

代わりにエラー・コードが表示された場合は、構成ファイルが正しい場所にないか、有効な文書ルート・エントリーが含まれていません。`query_contents.cfg` ファイルの構成を検査し、文書ルートが存在することを確認してください。

2. ブラウザーから、次の URL を入力します。

`http://windows-machine-name/cgi-bin/query_contents.exe?dirlist=/`

この URL によって、前のステップと同じ結果が戻されなければなりません。この結果が戻されない場合は、Web サーバーの CGI 構成に誤りがあります。サーバーの資料を参照して、問題を訂正します。

query_contents の一般プロセス・フロー

query_contents のジョブは、URL 要求に組み込まれているディレクトリーの内容を戻すためのものです。

例えば、サーバーの Web スペースのルート・ディレクトリーの内容を入手するには、ブラウザーで、次のような URL (UNIX 例) に対する **query_contents** を実行します。

`http://back-end-server/cgi-bin/query_contents.sh?dirlist=/`

query_contents スクリプトは、次のアクションを実行します。

1. **DOCROOTDIR** 変数用に構成された値を読み取ります。
2. 要求された URL から **QUERY_STRING** 変数の値を読み取って、要求された操作を取得し、オブジェクト・パスを入手します。

操作の値は、**OPERATION** 変数に保管されます。オブジェクト・パスの値は、**\$OBJPATH** 変数に保管されます。例では、**OPERATION** 変数の値は、`dirlist` です。**OBJPATH** 変数の値は、`/` です。

3. オブジェクト・パスに対してディレクトリー・リスト作成 (**DO**) を実行し、Security Access Manager サーバーによる使用に備えて、結果を標準出力に送信します。サブディレクトリーを示すエントリーには、ダブルスラッシュ (`//`) が付加されています。

通常の実出力は、以下のとおりです。

```
100
index.html
cgi-bin//
pics//
```

番号 100 は、正常に実行されたことを示す戻り状況です。

query_contents プログラムの保護

このタスクについて

Security Access Manager は、**query_contents** CGI プログラムを使用して、Web Portal Manager で、junction 先 Web サーバー・オブジェクト・スペースを表示します。このファイルを、無許可のユーザーが実行することができないように保護することは、きわめて重要な要件です。

手順

ポリシー・サーバー (**pdmgrd**) 識別以外は **query_contents** プログラムにアクセスできないようにするためのセキュリティ・ポリシーを設定する必要があります。次に示す ACL の例 (**query_contents_acl**) は、この基準を満たしています。

```
group ivmgrd-servers Tl
user sec_master dbxTrlcam
```

Junction 先サーバー上の **query_contents.sh** オブジェクト (UNIX) または **query_contents.exe** オブジェクト (Windows) にこの ACL を付加するには、**pdadmin** ユーティリティを使用します。例えば次のように指定します (UNIX の場合)。

```
pdadmin> acl attach /WebSEAL/host/junction-name/cgi-bin/query_contents.sh
query_contents_acl
```

第 24 章 拡張 junction の構成

ほとんどの標準 junction オプションは、仮想-host junction でもサポートされています。

仮想-host junction については、547 ページの『第 29 章 仮想-host junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想-host junction』で説明しています。

この章で扱うトピックは以下のとおりです。

- 『相互認証 SSL junction』
- 446 ページの『TCP および SSL のプロキシ junction の作成』
- 447 ページの『SSL を介した WebSEAL から WebSEAL への junction』
- 448 ページの『ステートフル junction』
- 453 ページの『新規 junction の強制適用』
- 455 ページの『Junction スロットル』
- 464 ページの『Cookie の管理』
- 466 ページの『Junction ポータル・サーバーへのセッション Cookie の引き渡し』
- 467 ページの『大文字小文字を区別しない URL のサポート』
- 468 ページの『Windows ファイル・システムへの junction』
- 470 ページの『仮想-host への標準 Junction』
- 471 ページの『HTTP ヘッダー・データ用の UTF-8 エンコード』
- 473 ページの『junction を介したシングル・サインオン・ソリューション』

相互認証 SSL junction

このセクションは、以下のトピックで構成されています。

- 『相互認証 SSL junction プロセスの要約』
- 444 ページの『バックエンド・サーバー証明書の妥当性検査』
- 445 ページの『識別名 (DN) の突き合わせ』
- 445 ページの『クライアント証明書による認証』
- 446 ページの『BA ヘッダーによる認証』

相互認証 SSL junction プロセスの要約

WebSEAL は、SSL junction (**-t ssl**、**-t sslproxy**、または **-t mutual**) を介して WebSEAL サーバーとバックエンド・サーバーとの間の相互認証をサポートします。

以下の概要説明は、SSL を介した相互認証についてサポートされる機能を要約したものです (必要に応じてコマンド・オプションもリストしてあります)。

1. WebSEAL は、バックエンド・サーバーの認証を行います (通常の SSL プロセス)。
 - WebSEAL は、バックエンド・サーバーにあるサーバー証明書の妥当性検査を行います。

『バックエンド・サーバー証明書の妥当性検査』を参照してください。

- WebSEAL は、証明書に含まれる識別名 (DN) を検査します (**-D**) (オプションですが、高水準のセキュリティーを提供します)。

445 ページの『識別名 (DN) の突き合わせ』を参照してください。

2. バックエンド・サーバーは、WebSEAL の認証を行います (2 つの方式があります)。
 - バックエンド・サーバーは、WebSEAL にあるクライアント証明書の妥当性検査を行います (**-K**)。

445 ページの『クライアント証明書による認証』を参照してください。

- バックエンド・サーバーは、基本認証 (BA) ヘッダーの WebSEAL の識別情報の妥当性検査を行います (**-B**、**-U**、**-W**)。

446 ページの『BA ヘッダーによる認証』を参照してください。

SSL を介して相互認証をコントロールするコマンド・オプションには、以下の機能があります。

- クライアント証明書または BA 認証方式を指定できます。
- Junction ごとに認証方式を適用できます。

SSL において **-b** オプション (BA 情報を処理する) を相互認証と組み合わせる場合に、特に考慮する点については、595 ページの『junction を介したクライアント識別情報』で説明してあります。

SSL 仮想ホスト junction を介した相互認証もサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

バックエンド・サーバー証明書の妥当性検査

WebSEAL は、標準 SSL プロトコルに従って、バックエンド・サーバー証明書を検査します。バックエンド・サーバーは、自分のサーバー証明書を WebSEAL に送信します。WebSEAL は、ルート認証局 (CA) 証明書の定義済みリストと比較して、サーバー証明書の妥当性検査を行います。

アプリケーション・サーバー証明書の trust チェーン (署名している CA からルート証明書まで) を形成する認証局 (CA) の証明書は、WebSEAL が使用する鍵データベースに組み込まれていなければなりません。

LMIを使用して、ルート CA 証明書のデータベースを作成して管理します。

識別名 (DN) の突き合わせ

このタスクについて

識別名 (DN) の突き合わせを行うことにより、サーバー・サイド証明書の検査を強化することができます。サーバー DN の突き合わせを使用可能化するには、当該サーバーに対して SSL junction を作成する際に、バックエンド・サーバー DN を指定しなければなりません。DN の突き合わせはオプションの構成ですが、SSL junction を介した相互認証において高度なセキュリティを提供します。

サーバー・サイド証明書の検査の際には、証明書に含まれている DN と、junction により定義されている DN とが比較されます。この 2 つの DN が一致しないと、バックエンド・サーバーへの接続は失敗します。

手順

サーバー DN の突き合わせを使用可能にするには、SSL ベースの junction の作成時に、**-D "DN"** オプションを使用してバックエンド・サーバー DN を指定します。ストリング内にブランク・スペースを入れたい場合は、DN ストリングを二重引用符で囲みます。以下に例を示します。

```
-D "CN=Access Manager,OU=SecureWay,O=Tivoli,C=US"
```

-D オプションは、**-K** または **-B** オプションと一緒に使用した場合のみ有効です。

クライアント証明書による認証

WebSEAL が、クライアント証明書を使用して、junction 先バックエンド・サーバーに認証できるようにするには、**-K** オプションを使用します。

```
-K "key_label"
```

このシナリオの条件は、以下のとおりです。

- バックエンド・サーバーは、クライアント証明書による WebSEAL の識別の検査を必要とするようにセットアップします。
- LMIを使用して、特殊鍵を作成し、ラベル付けし、そして保管します。この鍵は、junction 先バックエンド・サーバーに認証を行うときに、WebSEAL のクライアント証明書として使用されます。
- セキュリティの水準をさらに高めるには、さらに DN の突き合わせを考慮して junction を構成するようにしてください (**-D**)。

-K オプションは、GSKit 鍵データベースに保管されている、必須の証明書の鍵ラベルを指定する引数を使用します。LMIを使用して、鍵データベースに新規証明書を追加します。

鍵ラベル引数は、引用符で囲まなければなりません。例えば、次のとおりです。

```
-K "cert1_Tiv"
```

鍵が暗号ハードウェアにある場合は、鍵ラベルと共に WebSEAL トークン・デバイスを指定する必要があります。

```
-K "token_name:key-label"
```

例:

```
-K "websealtoken:junctionkey"
```

405 ページの『WebSEAL 鍵データベース・ファイルの構成』を参照してください。

BA ヘッダーによる認証

WebSEAL が基本認証を使用して認証を行えるようにするには、**-B-U "username"-W "password"** オプションを使用します。

```
-B -U "username" -W "password"
```

このシナリオの条件は、以下のとおりです。

- バックエンド・サーバーは、BA ヘッダーによる WebSEAL の識別の検査を必要とするようにセットアップします。
- Junction の構成では、**-b** オプションも使用しないでください。(ただし、内部的には、**-B** オプションは **-b filter** を使用します。)
- WebSEAL は、自分の識別情報を BA ヘッダーに渡して、バックエンド・サーバーに認証を行うように構成します。
- セキュリティーの水準をさらに高めるには、さらに DN の突き合わせを考慮して junction を構成するようにしてください (**-D**)。

ユーザー名引数とパスワード引数は、二重引用符で囲まなければなりません。例えば、次のとおりです。

```
-U "WS1" -W "abCde"
```

TCP および SSL のプロキシ junction の作成

通信が HTTP または HTTPS プロキシ・サーバーを使用するネットワーク・トポロジーを全探索できる WebSEAL junction を作成できます。標準 TCP 通信または保護 SSL 通信として要求を処理するように junction を構成できます。

プロキシ・サーバーを介して、TCP ベースまたは SSL ベースの junction を確立するには、**create** コマンドの **type** オプションに以下のいずれかの引数を指定する必要があります。

- **-t tcp** proxy
- **-t ssl** proxy

プロキシ・サーバーおよびターゲット Web サーバーを識別するには、**create** および **add** コマンドはいずれも、以下のオプションおよび引数を必要とします。

-H <i>host-name</i>	プロキシ・サーバーの DNS ホスト名または IP アドレス。
-P <i>port</i>	プロキシ・サーバーの TCP ポート。
-h <i>host-name</i>	ターゲット Web サーバーの DNS ホスト名または IP アドレス。

-p port	ターゲット Web サーバーの TCP ポート。デフォルトは、TCP junction の場合は 80 で、SSL junction の場合は 443 です。

TCP プロキシ junction の例 (1 行で入力します):

```
pdadmin> server task web1-webseald-cruz create -t tcpproxy
-H clipper -P 8081 -h www.ibm.com -p 80 /ibm
```

SSL プロキシ junction の例 (1 行で入力します):

```
pdadmin> server task web1-webseald-cruz create -t sslproxy
-H clipper -P 8081 -h www.ibm.com -p 443 /ibm
```

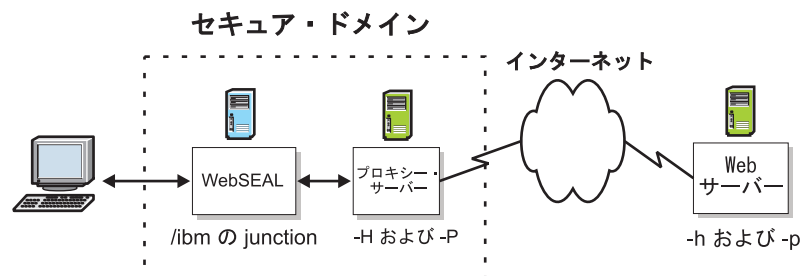


図 31. プロキシ junction の例

TCP および SSL のプロキシ仮想ホスト junction もサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

SSL を介した WebSEAL から WebSEAL への junction

Security Access Manager は、フロントエンド WebSEAL サーバーとバックエンド WebSEAL サーバーの間の SSL junction をサポートしています。SSL を介して 2 つの WebSEAL サーバーを junction して相互認証ができるようにするには、**create** コマンドの **-c** オプションを使用します。

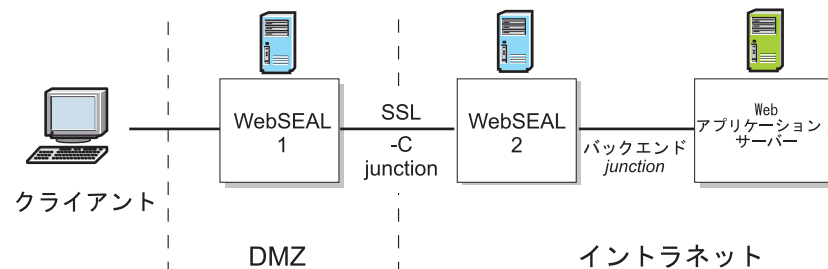


図 32. WebSEAL-to-WebSEAL junction シナリオ

例:

```
pdadmin> server task web1-webseald-cruz create -t ssl -C -h serverA /jctA
```

相互認証は、以下の 2 つの段階で発生します。

- SSL プロトコルを用いれば、バックエンド WebSEAL サーバーが、フロントエンド WebSEAL サーバーに対し、そのサーバー証明書を使って認証することができます。
- **-C** オプションを指定すると、フロントエンド WebSEAL サーバーが、その識別情報を基本認証 (BA) ヘッダーに入れて、バックエンド WebSEAL サーバーに渡すことができます。

さらに、**-C** オプションを使用すると、**-c** オプションによって提供されるシングル・サインオン機能性を使用可能にできます。**-c** オプションを使用すると、行き先がバックエンド WebSEAL サーバーである要求の HTTP ヘッダーの中に、Security Access Manager 固有のクライアント識別情報およびグループ・メンバーシップ情報を入れることができます。ヘッダー名には、iv-user、iv-groups、および iv-creds があります。596 ページの『HTTP ヘッダー内のクライアント識別 (-c)』を参照してください。

WebSEAL から WebSEAL への junction には以下の条件が適用されます。

- この junction は、**-t ssl** または **-t sslproxy** タイプの場合のみ有効です。
- どちらの WebSEAL サーバーも共通のユーザー・レジストリーを共有しなければなりません。この構成により、バックエンド WebSEAL サーバーは、フロントエンド WebSEAL サーバー識別情報の認証を行うことができます。
- WebSEAL から WebSEAL への junction およびバックエンド・アプリケーション・サーバーの junction の両方が **-j junction** オプション (junction Cookie 用) を使用する場合、2 つの WebSEAL サーバーのそれぞれによって作成された 2 つの junction Cookie の間でネーミングの競合が起こります。(このセクションの先頭にある図を参照してください。) この競合を避けるには、中間 WebSEAL サーバー (図の WebSEAL 2) を構成して junction Cookie を識別しなければなりません。中間 WebSEAL サーバーでのみ、WebSEAL 構成ファイルの **[script-filtering]** スタンザの **hostname-junction-cookie** スタンザ・エントリーの値を「yes」に設定します (デフォルトは「no」です)。

```
[script-filtering]
hostname-junction-cookie = yes
```

Junction Cookie を使用することにより、WebSEAL は、クライアント・サイドで生成されたサーバー相対 URL を処理できます。これらの URL は、宛先アプリケーションの Junction ポイントの情報を持っていません。Junction Cookie がこの情報を提供します。Junction Cookie について詳しくは、493 ページの『junction Cookie を使用したサーバー相対 URL の変更』を参照してください。

ステートフル junction

このセクションでは、以下のトピックについて説明します。

- 449 ページの『ステートフル junction の概念』
- 449 ページの『ステートフル junction の構成』
- 450 ページの『ステートフル junction のためのバックエンド・サーバー UUID の指定』
- 453 ページの『使用不可なステートフル・サーバーの処理』

ステートフル junction の概念

ほとんどの Web 対応アプリケーションは、クライアントからの一連の HTTP 要求に関する「状態」を保持しています。この状態を使用して、例えば、以下のことを行います。

- CGI プログラムによって生成されるデータ・エントリー形式内のフィールドにより、ユーザーの進行状態を追跡する。
- 一連のデータベース照会の実行時に、ユーザーのコンテキストを維持する。
- ユーザーが購入する品目をランダムにブラウズし選択するような、オンライン・ショッピング・カート・アプリケーション内に品目のリストを維持する。

ロード配分によるパフォーマンスの向上を図るために、Web 対応アプリケーションを実行するバックエンド・サーバーを複製できます。デフォルトでは、Security Access Manager は、使用可能な複製サーバーすべてにわたって要求を分散することによって、バックエンド・サーバーのロード・バランシングを行います。Security Access Manager は、「一番すいている」アルゴリズムを使用します。このアルゴリズムによって、各新規要求は、既に進行中の接続が最も少ないサーバーに送信されます。

ただし、WebSEAL がステートフル junction を介して要求を処理するとき、このセッション中にクライアントからのすべての後続の要求が同じサーバーに転送され、ロード・バランシング・ルールに従って、他の複製されたバックエンド・サーバー間では配分されないことが保証されなければなりません。

ステートフル junction の構成

ロード・バランシング・ルールを無効にしてステートフル Junction を作成するために、**-s** オプションを指定して **pdadmin server task create** コマンドを使用します。ステートフル junction は、クライアントの要求が、1 つのセッションの間中、同一のサーバーに転送されることを保証します。ステートフル junction を介して最初のクライアント要求が発生すると、WebSEAL は、指定されたバックエンド・サーバーの UUID を含むクライアント・システムに Cookie を配置します。クライアントが、同じセッション中に、同じリソースに対してその後要求を行うときは、Cookie の UUID 情報により、要求は、確実に同じバックエンド・サーバーにルーティングされクライアントが同じセッション中に同じリソースに対して後続の要求を行うと、Cookie の UUID 情報により、要求は確実に同じバックエンド・サーバーにルーティングされます。

-s オプションは、同じ junction ポイントで junction される複数のバックエンド・サーバーを備えた 1 つのフロントエンド WebSEAL サーバーに適しています。最初の junction がステートフルとして作成されると同時に、残りの複製バックエンド・サーバーを同じ junction ポイントに junction するために、**-s** オプションを指定しない **pdadmin server task add** コマンドが使用されるという点に注意してください。

ステートフル仮想ホスト junction もサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

すべて同じバックエンド・サーバーに junction された複数のフロントエンド WebSEAL サーバーがシナリオに含まれている場合は、**-u** オプションを使用して、各バックエンド・サーバー UUID を各フロントエンド WebSEAL サーバーに正しく指定しなければなりません。『ステートフル junction のためのバックエンド・サーバー UUID の指定』を参照してください。

また、使用不可になるステートフル・サーバーを WebSEAL がどのように処理するかを制御することもできます。453 ページの『使用不可なステートフル・サーバーの処理』を参照してください。

ステートフル junction のためのバックエンド・サーバー UUID の指定

このタスクについて

バックエンド Web アプリケーション・サーバーへの新規 junction を作成する際、WebSEAL は通常、汎用固有 ID (UUID) を生成して、そのバックエンド・サーバーを識別します。この UUID は内部的に使用されるほか、ステートフル junction を維持するためにも使用されます (**create -s**)。

最初のクライアント要求が発生すると、WebSEAL は、指定されたバックエンド・サーバーの UUID を含むクライアント・システムに Cookie を配置します。クライアントが、同じリソースに対してその後要求を行うときは、Cookie の UUID 情報により、要求は、確実に同じバックエンド・サーバーにルーティングされます。

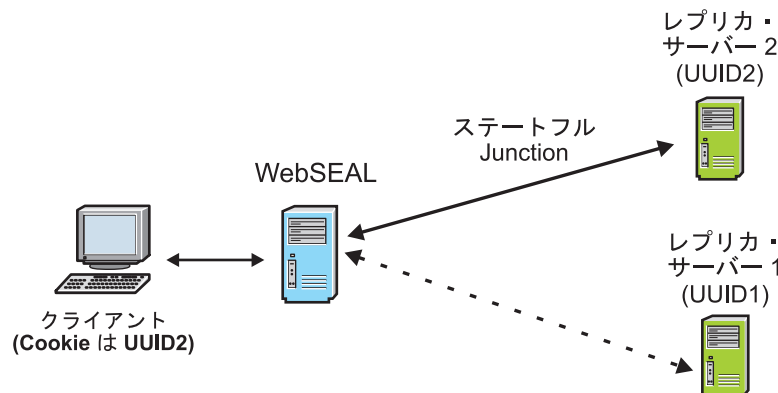


図 33. ステートフル junction でのバックエンド・サーバー UUID の使用

複数のバックエンド・サーバーに junction される複数のフロントエンド WebSEAL サーバーが存在するときは、ステートフル junction の処理は、さらに複雑になります。通常、1 つのフロントエンド WebSEAL サーバーと 1 つのバックエンド・サーバーの間の junction ごとに、バックエンド・サーバーは固有の UUID を生成します。すなわち、単一のバックエンド・サーバーが、フロントエンド WebSEAL サーバーごとに異なる UUID を持つことになります。

複数のフロントエンド・サーバーは、2 つのサーバー間のロードを分散するために、ロード・バランシング・メカニズムを必要とします。例えば、特定の UUID を使用し、WebSEAL サーバー 1 を介して、バックエンド・サーバーに対して初期「状態」を確立することができます。

しかし、同じクライアントからの今後の要求が、ロード・バランシング・メカニズムにより WebSEAL サーバー 2 を介してルーティングされる場合、WebSEAL サーバー 2 で同じ UUID を使用して、同じバックエンド・サーバーを識別しない限り、「状態」は存在しなくなります。通常、こういう問題は起きません。

-u オプションを用いれば、各フロントエンド WebSEAL サーバーに対して、特定のバックエンド・サーバーの同じ UUID を指定することができます。

また、**-u** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

例として、2 つの複製フロントエンド WebSEAL サーバーで、それぞれ 2 つのバックエンド・サーバーへのステートフル junction を備えているものを考えてみてください。WebSEAL サーバー 1 とバックエンド・サーバー 2 の間にステートフル junction を作成すると、バックエンド・サーバー 2 を識別する固有の UUID (UUID A) が生成されます。しかし、WebSEAL サーバー 2 とバックエンド・サーバー 2 の間にステートフル junction が作成されると、バックエンド・サーバー 2 を識別するための新規の異なる UUID (UUID B) が生成されます。

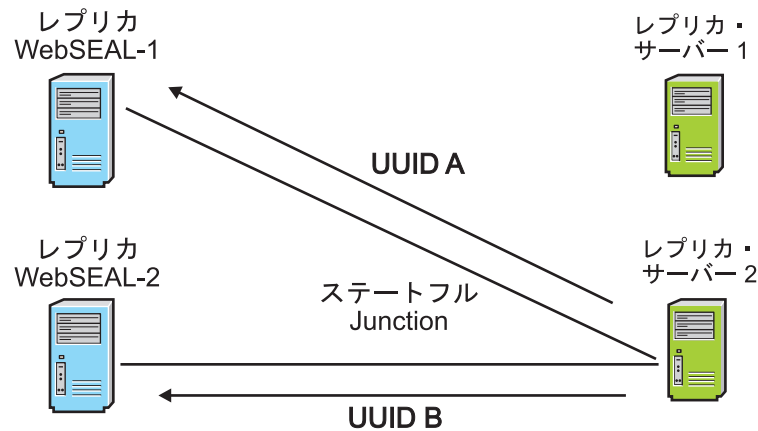


図 34. 異なる UUID

WebSEAL サーバー 1 を経て、クライアントとバックエンド・サーバー 2 の間に確立された「状態」は、クライアントからの次の要求が、WebSEAL サーバー 2 を介してルーティングされた場合は失敗します。

次の図では、バックエンド・サーバー 1 は、UUID 1 として、WebSEAL-1 と WebSEAL-2 の両方で認識されています。バックエンド・サーバー 2 は、UUID 2 として、WebSEAL-1 と WebSEAL-2 の両方で認識されています。

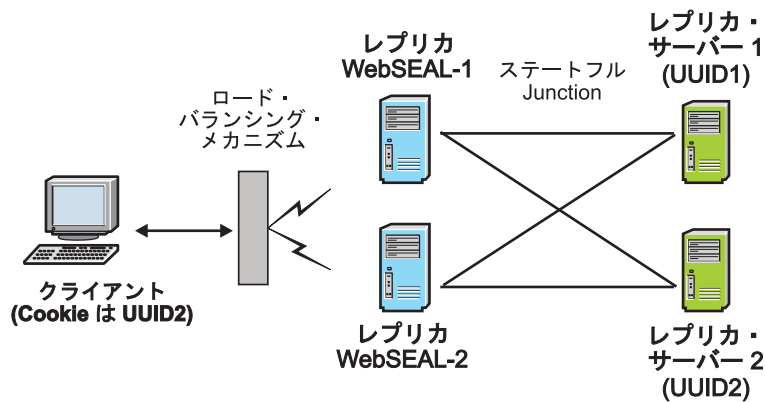


図 35. ステートフル *junction* のためのバックエンド・サーバー *UUID* の指定

手順

Junction の作成時に *UUID* を指定するため、以下の処理を適用してください。

1. WebSEAL サーバー 1 から各バックエンド・サーバーへの *junction* を作成します。 **create -s** および **add** を使用します。
2. ステップ 1 で、バックエンド・サーバーごとに生成される *UUID* をリストします。 **show** を使用します。
3. WebSEAL サーバー 2 から各バックエンド・サーバーへの *junction* を作成して、ステップ 2 で識別される *UUID* を指定します。 **create -s -u** および **add -u** を使用します。

ステートフル *junction* の例

以下の例には次の条件を適用します。

- WebSEAL-1 インスタンスを WS1 と呼ぶ。ホスト名は **host1**。
- WebSEAL-2 インスタンスを WS2 と呼ぶ。ホスト名は **host2**。
- バックエンド・サーバー 1 を APP1 と呼ぶ。
- バックエンド・サーバー 2 を APP2 と呼ぶ。

```
pdadmin> server task WS1-webseald-host1 create -t tcp -h APP1 -s /mnt
pdadmin> server task WS1-webseald-host1 add -h APP2 /mnt
pdadmin> server task WS1-webseald-host1 show /mnt
```

(output of this command reveals *UUID1* and *UUID2*)

```
pdadmin> server task WS2-webseald-host2 create -t tcp -h APP1 -u UUID1 -s /mnt
pdadmin> server task WS2-webseald-host2 add -h APP2 -u UUID2 /mnt
```

クライアントは、バックエンド・サーバー 2 とのステートフル接続を確立する際に、*UUID2* を含む Cookie を受け取ります。この例では、今後の要求が WebSEAL-1 または WebSEAL-2 を介してルーティングされるかどうかに関係なく、クライアントは、常に、バックエンド・サーバー 2 に接続されることが保証されます。

使用不可なステートフル・サーバーの処理

このタスクについて

WebSEAL 構成ファイルの `[junction]` スタンザの `use-new-stateful-on-error` スタンザ・エントリーを使用して、使用不可になるステートフル・サーバーに WebSEAL がどのように応答するかをコントロールすることができます。

`use-new-stateful-on-error` が「yes」に設定されている場合で、元のサーバーがセッション中に使用不可になった場合、WebSEAL は、ユーザーの次の要求を同じステートフル junction 上の新規レプリカ・サーバーに送信します。新規レプリカ・サーバーが、そのステートフル Junction で検出され、その要求に応答している場合、WebSEAL は、新しいステートフル Cookie をユーザーのブラウザー上に設定します。このセッション中の後続の要求は、同じ新規サーバーに送信されます。

`use-new-stateful-on-error` が「no」に設定されている場合で、元のサーバーがセッション中に使用不可になった場合、WebSEAL は、ユーザーの後続の要求を同じステートフル junction 上の新規レプリカ・サーバーに送信しません。代わりに、WebSEAL はエラーを戻し、このセッション中のユーザーによる後続の要求のために同一のサーバーにアクセスを試みます。デフォルト設定の「no」値は、バージョン 6.0 より前の WebSEAL との互換性を保つためのものです。例:

```
[junction]
use-new-stateful-on-error = no
```

応答しないサーバーでのアクセス試行を終えるには、ユーザーがブラウザーを再始動するか、管理者がその応答しないサーバーを除去する必要があります。

調整した構成項目を `[junction:{junction_name}]` スタンザに追加することによって、特定の junction 用にこの構成項目をカスタマイズできます。

ここで、`{junction_name}` は、標準 junction の junction ポイント (先頭の / 文字を含む)、または仮想ホスト junction の仮想ホスト・ラベルを表します。例:

```
[junction:/WebApp]
```

新規 junction の強制適用

このタスクについて

既存の junction を新規の junction で上書きするよう強制するときは、`-f junction` オプションを使用する必要があります。

以下の例 (WebSEAL インスタンス名 = **cruz**) は、そのための手順を示しています。

手順

1. **pdadmin** にログインします。

```
# pdadmin
pdadmin> login
Enter User ID: sec_master
Enter Password:
pdadmin>
```

2. **server task list** コマンドを使用して、現行 junction ポイントをすべて表示します。

```
pdadmin> server task web1-webseald-cruz list
/
```

3. **server task show** コマンドを使用して、junction の詳細を表示します。

```
pdadmin> server task web1-webseald-cruz show /
Junction point: /
Type: Local
Junction hard limit: 0 - using global value
Junction soft limit: 0 - using global value
Active worker threads: 0
Root Directory: /opt/pdweb/www/docs
...
```

4. 新規のローカル junction を作成して、現行 junction ポイントと置き換えます (既存の junction を新規 junction で強制的に上書きするには、**-f** オプションを指定する必要があります)。

```
pdadmin> server task web1-webseald-cruz create -t local -f -d /tmp/docs /
Created junction at /
```

5. 新規 junction ポイントをリストします。

```
pdadmin> server task web1-webseald-cruz list
/
```

6. この junction の詳細を表示します。

```
pdadmin> server task web1-webseald-cruz show /
Junction point: /
Type: Local
Junction hard limit: 0 - using global value
Junction soft limit: 0 - using global value
Active worker threads: 0
Root Directory: /tmp/docs
...
```

また、**-f** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

仮想ホスト Junction での /pkmslogout の使用

ポリシーを pkmslogout に付加することはできますが、WebSEAL が常にポリシーを適用するとは限りません。

例えば、ユーザーが WebSEAL に認証された状態で pkmslogout へのアクセスを試行すると、pkmslogout ページは許可検査を実行せずにユーザーのセッションを終了します。ACL ポリシーは、このような要求には適用されません。ただし、ユーザーが WebSEAL に認証されていない状態で pkmslogout へのアクセスを試行すると、要求は通常の要求として処理されます。WebSEAL は許可検査を実行します。

許可検査に失敗した場合、要求は通常の許可の失敗として続行されます。WebSEAL のデフォルト構成では、ユーザーにログインを求めるプロンプトが出されます。

許可検査にパスすると、WebSEAL はルート junction から /pkmslogout というオブジェクトを取得しようとし、通常はこれによって、404「見つかりません」の応答が WebSEAL から戻されます。

[**acnt-mgmt**] スタンザの **allow-unauthenticated-logout** オプションは、非認証ユーザーが、最初に認証されることなく **pkmslogout** リソースを要求できるかどうかを判別します。**yes** に設定された場合、ログアウトするユーザーが認証済みでも認証済みでなくても、WebSEAL は同じ方法で動作します。

Security Access Manager を使用してシングル・ログアウトを実現するには、いくつかの方法があります。1 つの方法は、**** または **<IFRAME>** HTML タグをログアウト・ページに埋め込むことです。これによって、そのページが表示されたとき、ブラウザーはユーザーを複数のサーバーから同時にログアウトします。例えば、以下の HTML タグは、3 つの異なる仮想ホストの **/pkmslogout** に要求を送信します。

```



```

この技法をシングル・ログアウトに使用する場合、ACL を **/pkmslogout** に付加するか、[**acnt-mgmt**] **allow-unauthenticated-logout** オプションを使用して WebSEAL の動作を制御することが有利になる場合があります。**allow-unauthenticated-logout** オプションについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

Junction スロットル

このセクションは、以下のトピックで構成されています。

- 『junction スロットルの概念』
- 456 ページの『スロットル状態への junction サーバーの遷移』
- 458 ページの『オフライン状態の Junction サーバー』
- 460 ページの『オンライン状態の Junction サーバー』
- 462 ページの『junction スロットル・メッセージ』
- 463 ページの『junction スロットルと既存の WebSEAL 機能の併用』

junction スロットルの概念

コンピューター・ネットワーク環境における設備の定期保守は、重要かつ必須のタスクです。一般に、WebSEAL 環境では、データ・ストレージおよびアプリケーション・プログラムは、WebSEAL によって保護された junction バックエンド・ホスト・マシンに常駐します。通常、頻繁に使用される WebSEAL 環境は、複製されたコンテンツおよびアプリケーションのホストとして機能する複数のマシンで構成されるサーバー・クラスターに依存しています。

レプリカ・サーバー環境によって、個々のサーバーをオフラインにし、定期的な保守を実行することが可能になります。ネットワークの負荷は残りのレプリカ全体に再配分されるため、スムーズなユーザー・エクスペリエンスが可能となります。

junction スロットルによって、既存のセッションとのユーザーのトランザクションを妨げることなく、junction バックエンド Web サーバーを徐々にオフラインに切り替

えることができます。junction のスロットル・アクションは、ショッピング・カート・トランザクションなどの完了するまで継続するステートフル・セッションを有効にする場合に特に便利です。

junction スロットルによって、次のアクションが可能になります。

- スロットル・サーバーは、スロットル・アクションが実行される前に作成されたセッションを実行しているユーザーからの現在および後続の要求を継続して処理します。
- スロットル・サーバーは、非認証ユーザーおよび新規認証ユーザーからのすべての要求をブロックするとともに、これらの要求を同じ junction 上のその他の使用可能なレプリカ・サーバーに誘導します。
- 現行ユーザーがセッションを終了すると、スロットル・サーバーは最終的にアイドル状態になり、オフラインに切り替え可能になります。
- junction スロットルでは、WebSEAL を停止する必要はなく、その他の junction Web サーバーへのユーザーのアクセスが妨げられません。

pdadmin ユーティリティでは、junction サーバーを以下の 3 つの操作状態のいずれかにするコマンドが使用可能です。

- スロットル
- オフライン
- オンライン

注: この 3 つの操作状態は、junction サーバーの実行状態 (実行中、非実行中、不明、および非 HTTP サーバー) とは異なります。サーバーの実行状態は、**pdadmin server task show** および **pdadmin server task virtualhost show** コマンドの「サーバーの状態」フィールドで報告されます。

このコマンドによってユーザーは、junction 上のサーバーを個々にまたは集散的に管理できます。集散的な管理は、例えば、セキュリティ・ブリーチ (抜け穴) などが必要となる場合があります。

junction スロットル機能は、Web ポータル・マネージャーからは制御できません。

junction スロットル機能は、標準 WebSEAL junction および仮想ホスト junction でサポートされています。junction スロットルは、標準ローカル junction または仮想ホスト・ローカル junction では使用できません。

スロットル状態への junction サーバーの遷移

このタスクについて

サーバーをオフライン状況に管理しながら徐々に遷移する場合は、junction サーバーをスロットル操作状態にします。スロットル操作状態によって、junction の状態は以下ようになります。

- スロットル・サーバーは、スロットル・アクションが実行される前に作成されたセッションを実行しているユーザーからの現在および後続の要求を継続して処理します。

- スロットル・サーバーは、非認証ユーザーおよび新規認証ユーザーからのすべての要求をブロックするとともに、これらの要求を同じ `junction` 上のその他の使用可能なレプリカ・サーバーに誘導します。
- 要求がスロットル・サーバーによってブロックされると、別のレプリカ・サーバーで再試行されます。レプリカ・サーバーが構成されていないか、または使用不可の場合は、アプリケーション・サーバーがオフラインであることを示すエラー・ページがユーザーに戻されます。
- `junction` のスロットル操作状態は `junction` データベースに保管され、WebSEAL サーバーの再始動時にはその状態が常に維持されます。

標準 WebSEAL junction のスロットル・コマンドの使用法

以下の **pdadmin** コマンド構文は、標準 WebSEAL junction での使用に適しています。

```
server task instance_name-webseald-host_name throttle [-i server_uuid] jct_point
```

-i オプションを使用して、スロットル操作状態に遷移中の `junction` サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、`junction` にあるすべてのサーバーがスロットル操作状態になります。

例:

以下の例では、`/pubs` Junction ポイントにある `backappl` サーバーがスロットル操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう
に、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Online
Hostname: backappl.diamond.example.com
...
Current requests: 0
...
```

次に、このサーバーをスロットル操作状態にします (1 行で入力します):

```
pdadmin> server task default-webseald-cruz throttle
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

仮想ホスト junction のスロットル・コマンドの使用法

以下の **pdadmin** コマンド構文は、仮想ホスト `junction` での使用に適しています。

```
server task instance_name-webseald-host_name virtualhost throttle [-i server_uuid]
vhost_label
```

-i オプションを使用して、スロットル操作状態に遷移中の `junction` サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、`junction` にあるすべてのサーバーがスロットル操作状態になります。

以下の例では、WebSEAL サーバー `abc.ibm.com` 上に構成されている `support-vhost-https` というラベルの仮想ホスト Junction により、バックエンド Junction サーバー `int3.ibm.com` にある仮想ホスト `support.ibm.com` がサポートされます。

`int3.ibm.com` サーバーをスロットル操作状態にする要求があります。この junction サーバーの UUID を判別するには、**server task virtualhost show** コマンドを実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Online
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1
```

次のコマンドを使用して、このサーバーをスロットル操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost throttle
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

オフライン状態の Junction サーバー

すべてのユーザーからのあらゆる要求をブロックする場合、junction サーバーをオフライン操作状態に遷移します。オフライン操作状態によって、junction の状態は以下のようになります。

- オフライン・サーバーは既存の要求の処理を継続します。
- オフライン・サーバーはすべてのユーザーからのあらゆる後続の要求をブロックします。
- 要求がオフライン・サーバーによってブロックされると、別のレプリカ・サーバーで再試行されます。レプリカ・サーバーが構成されていないか、または使用不可の場合は、アプリケーション・サーバーがオフラインであることを示すエラー・ページがユーザーに戻されます。
- junction のオフライン操作状態は junction データベースに保管され、WebSEAL サーバーの再始動時にはその状態が常に維持されます。

標準 WebSEAL junction のオフライン・コマンドの使用法

以下の **pdadmin** コマンド構文は、標準 WebSEAL junction での使用に適しています。

```
server task instance_name-webseald-host_name offline [-i server_uuid] jct_point
```

-i オプションを使用して、オフライン操作状態に遷移中の junction サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、junction にあるすべてのサーバーがオフライン操作状態になります。

例:

以下の例では、/pubs junction ポイントにある backappl サーバーがオフライン操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう
に、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Throttled
Throttled at: 2005-03-01-17:07:24
Hostname: backappl.diamond.example.com
...
Current requests: 0
...
```

次に、このサーバーをオフライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz offline
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

仮想ホスト junction のオフライン・コマンドの使用法

以下の **pdadmin** コマンド構文は、仮想ホスト junction での使用に適しています。

```
server task instance_name-webseald-host_name virtualhost offline [-i server_uuid]
vhost_label
```

-i オプションを使用して、オフライン操作状態に遷移中の junction サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、junction にあるすべてのサーバーがオフライン操作状態になります。

以下の例では、WebSEAL サーバー abc.ibm.com 上に構成されている support-vhost-https というラベルの仮想ホスト Junction により、バックエンド Junction サーバー int3.ibm.com にある仮想ホスト support.ibm.com がサポートされます。

int3.ibm.com サーバーをオフライン操作状態にする要求があります。この junction サーバーの UUID を判別するには、**server task virtualhost show** コマンドを実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Throttled
Throttled at: 2005-03-01-17:07:24
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1
```

次のコマンドを使用して、このサーバーをオフライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost offline
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

オンライン状態の Junction サーバー

サーバーを通常の操作に戻す場合、junction サーバーをオンライン操作状態に遷移します。オンライン操作状態によって、Junction の状態は以下のようになります。

- オンライン・サーバーでは通常の操作が再開されます。
- junction のオンライン操作状態は junction データベースに保管され、WebSEAL サーバーの再始動時にはその状態が常に維持されます。

標準 WebSEAL junction のオンライン・コマンドの使用法

以下の **pdadmin** コマンド構文は、標準 WebSEAL junction での使用に適しています。

```
server task instance_name-webseald-host_name online [-i server_uuid] jct_point
```

-i オプションを使用して、オンライン操作状態に遷移中の junction サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、junction にあるすべてのサーバーがオンライン操作状態になります。

例:

以下の例では、/pubs junction ポイントにある backapp1 サーバーがオンライン操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう
に、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```

Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Offline
Hostname: backappl.diamond.example.com
...
Current requests: 0
...

```

次に、このサーバーをオンライン操作状態にします (1 行で入力します)。

```

pdadmin> server task default-webseald-cruz online
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs

```

仮想ホスト junction のオンライン・コマンドの使用法

以下の **pdadmin** コマンド構文は、仮想ホスト junction での使用に適しています。

```

server task instance_name-webseald-host_name virtualhost online [-i server_uuid]
vhost_label

```

-i オプションを使用して、オンライン操作状態に遷移中の junction サーバーの UUID を指定します。サーバーがこのオプションを使用して指定されない場合、junction にあるすべてのサーバーがオンライン操作状態になります。

以下の例では、WebSEAL サーバー `abc.ibm.com` 上に構成されている `support-vhost-https` というラベルの仮想ホスト Junction により、バックエンド Junction サーバー `int3.ibm.com` にある仮想ホスト `support.ibm.com` がサポートされます。

`int3.ibm.com` サーバーをオンライン操作状態にする要求があります。この junction サーバーの UUID を判別するには、**server task virtualhost show** コマンドを実行します。

```

pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https

```

出力は以下のようになります。

```

Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Offline
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1

```

次のコマンドを使用して、このサーバーをオンライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost online  
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

junction スロットル・メッセージ

このセクションは、以下のトピックで構成されています。

- 『junction スロットル・エラー・ページ』
- 『スロットル・サーバーの状況およびアクティビティのモニター』

junction スロットル・エラー・ページ

junction サーバーの操作状態がスロットルまたはオフラインであるため要求がブロックされると、エラー・ページ (38b9a4b0.html) がユーザーに戻されます。このエラー・ページは、junction 上のすべてのサーバーがスロットルまたはオフライン操作状態の場合にのみ送信されます。

ユーザーは、この HTML エラー・ページの内容をカスタマイズできます。86 ページの『HTML サーバー応答ページの変更』を参照してください。

または、エラー・レポートを外部のエラー・ページ・サービスへリダイレクトするように WebSEAL を構成することもできます。103 ページの『ローカル応答リダイレクト』を参照してください。

スロットル・サーバーの状況およびアクティビティのモニター

スロットル・サーバーの状況およびアクティビティは、**pdadmin server task show** および **pdadmin server task virtualhost show** コマンドに対して、以下の 3 つの新規フィールドを使用することによって、モニターできます。

- 操作状態: {スロットル|オフライン|オンライン}
- 現行要求: 数値

このフィールドには、現在 junction サーバーを使用しているワーカー・スレッド数が表示され、サーバーがアイドル状態になる時期を特定できます。

- スロットル日時: 日時

このフィールドには、サーバーがスロットル操作状態になる時期のみが表示されます。

例:

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs  
...  
Server 1:  
ID: 6fc3187a-ealc-11d7-8f4e-09267e38aa77  
Server State: running  
Operational State: Throttled
```



```
Throttled at: 2005-03-01-17:07:24
Hostname: backappl.diamond.example.com
...
Current requests: 3
...
```

「現行要求」フィールドは、サーバーがアイドル状態かどうかを判断する際に役に立ちます。ただし、さまざまな状態によって、サーバー・アクティビティの状況を完全に正しく判別することが困難な場合もあります。例:

- 「現行要求」フィールドは、junction サーバーからの応答本文を読み取り、クライアントへ応答を戻す間のラグ・タイム中は更新されません。
- セッションの存続時間のリセットおよび延長が構成されている場合、サーバーでの処理が終了し今後操作が行われないということを保証することはほぼ不可能になります。

junction スロットルと既存の WebSEAL 機能の併用

junction スロットルは、以下の WebSEAL 機能に影響を与えます。

- フェイルオーバー認証

フェイルオーバー認証は、junction のスロットル前に元のセッションが作成されている場合、スロットル junction を継続して使用するフェイルオーバー・セッションを透過的にサポートします。セッション作成時間は、属性としてフェイルオーバー Cookie に追加され、フェイルオーバー Cookie が認証のために使用される際に復元されます。フェイルオーバー Cookie が認証に使用されると、Cookie からのセッション作成時間が新規に作成されたフェイルオーバー・セッションに対して設定されます。

- Session Management Server (SMS)

SMS によって、セッションを共用するすべてのプロセスでセッション作成時間が有効になります。セッション作成時間は、junction サーバーのスロットル前に作成されたセッションのみがスロットル junction サーバーへのアクセスを継続できるため、重要です。

- 再認証

再認証セッションは、このセッションが junction のスロットル前に最初に作成された場合、スロットル junction サーバーへのアクセスを継続できます。セッション存続時間の延長またはリセットというその他の影響があるため、スロットル junction が実際にアイドル状態である時期を特定することが困難になる場合があります。

- ユーザー切り替え

ユーザー切り替えイベントが発生すると、新規セッション作成時間が生成されます。この新規作成時間は、スロットル junction サーバーへのアクセス可能性を判別するために使用されます。切り替えたユーザーがログアウトし、元の ID に戻ると、元のセッション作成時間が再度有効になり、スロットル junction サーバーへのアクセス可能性を判別するために使用されます。

- ステートフル junction

ステートフル junction によって、特定のセッションからの要求が junction 上の同一のサーバーに必ず送信されるようになります。使用中の junction サーバーの操作状態がスロットルである場合、ステートフル・セッションはそのサーバーへのアクセスを継続できます。ただし、新規のステートフル・セッションでそのサーバーを使用することはできません。

junction サーバーがオフラインになると、ステートフル・セッションはサーバーにアクセスできなくなります。これらのセッションは新規 junction サーバーを選択する必要があり、元の状態情報が失われる可能性があります。

- ステップアップ認証

ステップアップ認証では、新規セッションは作成されません。したがって、セッション作成時間は影響を受けず、スロットル junction にアクセスできるセッションも変わりません。

- Web ポータル・マネージャー (WPM) を使用した junction の変更

スロットル junction を Web ポータル・マネージャーを使用して変更すると、必ず「スロットル日時」情報が失われます。スロットル junction は WPM によって変更されると、オンライン状態に戻されます。WPM には junction スロットル操作を実行する機能がないため、junction をスロットル状態に再度戻すには、**pdadmin** ユーティリティを使用する必要があります。

Cookie の管理

WebSEAL はブラウザの代わりに Cookie をホストし、転送される要求でバックエンド・アプリケーションに提供することができます。保存されるこれらの Cookie はブラウザには送信されず、セッション・キャッシュ (つまり Cookie jar) 内に保持されます。

WebSEAL の Cookie jar は、ユーザー・セッション単位でインスタンス化されます。Cookie jar に保存されない Cookie は、クライアントに渡されて保存されません。

Cookie jar は、[junction] スタンザの以下の構成エントリーによる定義に従って、Cookie を保存および処理します。

managed-cookies-list

パターン・マッチングされる Cookie 名のコンマ区切りリストが含まれています。このリストのパターンに一致した Cookie は、Cookie jar に保存されます。このリストが空の場合、Cookie jar は実質的に使用不可になります。

reset-cookies-list

ユーザー・セッションがログアウトされたときにリセットする Cookie を決定します。クライアントから受信した要求およびクライアントに返信される応答の両方について、一致する Cookie があるかどうか検査されます。リセットによって、空の Cookie および有効期限が切れた Cookie をログアウト応答で戻すことによって、クライアントのキャッシュの Cookie をクリアします。これは実質的に、junction アプリケーションに対する基本的なログアウトをインプリメントしています。

share-cookies

Cookie jar に保存されている Cookie を、異なる junction 間で共有するかどうかを決定します。

validate-backend-domain-cookie

このエントリーが *true* に設定されている場合のみ、Cookie のドメイン・チェックが実行されます。

allow-backend-domain-Cookies

ドメインの検証中、このエントリーが *false* に設定された場合、ドメインは Cookie から削除されます。

注: 上記のすべての構成項目 (**share-cookies** を除く) は、調整した構成項目を **[junction:{function_name}]** スタンザに追加することによって、特定の junction 用にカスタマイズできます。

ここで、*{function_name}* は、標準 junction の junction ポイント (先頭の / 文字を含む)、または仮想ホスト junction の仮想ホスト・ラベルを表します。

すべての応答 Cookie は WebSEAL の Cookie jar を通過します。

managed-cookies-list に定義されたパターンに一致する Cookie は Cookie jar に保存され、ブラウザへの応答ストリームから削除されます。Cookie jar に保存されないものはクライアントに渡されます。

junction 先サーバーへの要求がブラウザから WebSEAL に送信されると、Cookie jar を検査して、その要求で Cookie を junction 先サーバーに送信する必要があるかどうかを調べます。その要求で Cookie jar の Cookie を必要とする場合、Cookie は要求に追加されます。Cookie の有効期限が切れている場合、Cookie は Cookie jar から削除されて送信されません。

永続的 Cookie は WebSEAL マシンのディスクに永続しません。

ユーザーがログアウトを実行すると、Cookie jar に保存されていない選択された Cookie のリセットが応答で送信されます。WebSEAL では、**reset-cookies-list** スタンザ・エントリーのパターンのリストに一致する名前を持つすべての Cookie をリセットします。このリセットは実質的に、junction アプリケーションに対する基本的なログアウトをインプリメントしています。

注: 複数の複製 WebSEAL サーバーによって Cookie jar が使用される状況では、Session Management Server をデプロイする必要があります。Session Management Server は、Cookie jar を複数の複製 WebSEAL サーバーに分散できるメカニズムを提供します。このタイプの環境では、Cookie jar に配置する Cookie に注意してください。定期的に更新される Cookie を含めないでください。含めた場合、Session Management Server の負荷が増加し、パフォーマンスへの影響が環境内に発生します。

Junction ポータル・サーバーへのセッション Cookie の引き渡し

Web ポータルは、広範囲にわたるパーソナライズされたリソースおよびサービスを提供するサーバーです。**-k junction** オプションを使用すると、Security Access Manager セッション Cookie (最初にクライアントと WebSEAL の間に確立されたもの) を、バックエンド・ポータル・サーバーに送信することができます。

クライアントがポータル・サーバーに個人用リソース・リストを要求すると、ポータル・サーバーは、同じく WebSEAL で保護されているその他のサポート・アプリケーション・サーバーにあるリソースにアクセスして、このリストを作成します。ポータル・サーバーは、セッション Cookie を使用することにより、クライアントに代わって、これらのアプリケーション・サーバーへのシームレスなシングル・サインオンを行います。

-k オプションは、WebSEAL とバックエンド・ポータル・サーバーの間の junction を作成するときに、引数なしで組み込みます。

また、**-k** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

WebSEAL 構成ファイルには、ステップアップ認証中にセッション Cookie を処理する方法を一部制御するオプションが含まれています。**[step-up]** スタンザの **verify-step-up-user** オプションは、ステップアップ操作を実行するユーザーの ID が、以前認証を実行したユーザーの ID と一致する必要があるかどうかを判断します。このオプションが **yes** に設定された場合、**retain-stepup-session** オプションを使用して、ステップアップ操作中に発行されたセッション Cookie が再使用できるか、または新規の Cookie を発行する必要があるかどうかを判断することができます。**verify-step-up-user** が **no** に設定されると、ステップアップ後に新規の Cookie を常に発行する必要があります。

[session] スタンザの **send-constant-sess** オプションは、認証済みセッションの追跡能力を高めます。このオプションを **yes** に設定すると、WebSEAL はセッション Cookie に加えて別の Cookie を junction 先サーバーに送信することができます。この Cookie の値は、セッション鍵が変更されるかどうかにかかわらず、単一セッション全体で一定に保たれます。この Cookie の名前は構成可能です。

send-constant-sess オプションについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

ポータル・サーバーの構成に際して考慮すべき条件は、次のとおりです。

- ユーザー名とパスワードを使用したアクセスの場合は、フォーム認証が必要です。基本認証 (BA) は使用しないでください。
- WebSEAL 構成ファイルの **[session]** スタンザ内の **ssl-id-sessions** スタンザ・エントリーの値は **no** に設定する必要があります。HTTPS 通信の場合は、この設定を使用すると、SSL セッション ID の代わりにセッション Cookie を使用してセッション状態が維持されます。
- フロントエンド WebSEAL クラスターがポータル・サーバーとしての役割を果たす場合は、フェイルオーバー・タイプの Cookie を使用可能にしてください。フ

エイルオーバー Cookie には暗号化された資格情報が含まれています。この情報を使用することで、要求を処理するすべての複製 WebSEAL サーバーでも認証が成功します。

- **[step-up]** スタンザの **retain-stepup-session** オプションは、**verify-step-up-user** オプションが **yes** に設定されている場合のみ有効です。
- WebSEAL がセッションの保存に Session Management Server (SMS) を使用するように構成されている場合、ステップアップ操作を使用可能にするために **verify-step-up-user** オプションを **yes** に設定する必要があります。このオプションが **no** に設定された場合、ステップアップ認証中にユーザー ID が変更されたときに WebSEAL は SMS を更新しません。

ステップアップ認証について詳しくは、201 ページの『認証強度の概念』を参照してください。

大文字小文字を区別しない URL のサポート

デフォルトでは、Security Access Manager は、アクセス・コントロールで検査を行うときに、URL を大文字小文字の区別があるものとして取り扱います。**-i junction** オプションを指定すると、WebSEAL は、junction 先バックエンド・サーバーに対する要求の許可検査を行うときに、URL を、大文字小文字を区別しないものとして取り扱います。

また、**-i** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

Junction でこのオプションを設定すると、WebSEAL は、URL の構文解析を行う際に、大文字と小文字を区別しません。デフォルトでは、Web サーバーが大文字小文字を区別することを想定しています。

ほとんどの HTTP サーバーでは、URL を大文字小文字を区別するように定義する HTTP 仕様をサポートしていますが、HTTP サーバーによっては、大文字小文字を区別しないように URL を処理するものもあります。

例えば、大文字小文字を区別しないサーバーでは、

`http://server/sales/index.htm`

`http://server/SALES/index.HTM`

この 2 つの URL は同じものと見なされます。この振る舞いにより、アドミニストレータは、両方の URL において同じアクセス・コントロール (ACL) を適用する必要があります。

-i オプションを指定してサード・パーティー・サーバーを junction すると、WebSEAL は、そのサーバーに送信される URL を、大文字小文字を区別しないで処理します。

大文字小文字を区別しない junction の要求を適切に許可するため、WebSEAL では URL の小文字バージョンで許可検査を実行します。例えば、Windows で実行中の Web サーバーは、INDEX.HTM および index.htm の要求を同じファイルの要求として扱います。

Web サーバーなどとの junction は **-i** [または **-w**] フラグによって作成される必要があります。Junction ポイント下のオブジェクトに付加する ACL または POP は、小文字のオブジェクト名を使用する必要があります。/junction/index.htm に付加された ACL は、**-i** または **-w** フラグが使用されると、以下の要求のすべてに適用されます。

```
/junction/INDEX.HTM  
/junction/index.htm  
/junction/InDeX.HtM
```

このオプションは、local タイプを除くすべての Junction に対して有効です。ローカル junction は Win32 プラットフォームでのみ大文字小文字の区別がありません。他のすべてのプラットフォームでは、大文字小文字の区別があります。

重要: **-i** オプションを使用するときは、オブジェクト名は小文字にして、WebSEAL が、これらのオブジェクトに付加されている ACL または POP があれば検出できるようにしなければなりません。詳細については、470 ページの『小文字オブジェクト名への ACL および POP の付加』を参照してください。

Windows ファイル・システムへの junction

WebSEAL では、URL に指定されているファイル・パスに基づいて、junction 先バックエンド・サーバーに対するクライアント要求のセキュリティー検査が実行されます。Win32 ファイル・システムでは、長いファイル名へのアクセス用に 2 種類の方式が用意されているため、このセキュリティー検査の能力が減殺されることがあります。

第 1 の方式では、ファイル名全体を確認します。例:

```
abcdefghijkl.txt
```

第 2 の方式では、後方互換性を維持するために旧 8.3 ファイル名形式を認識します。例:

```
abcdef~1.txt
```

Windows 環境で junction を作成するときは、1 つのオブジェクト表記にのみにアクセス・コントロールを制限してください。セキュリティー・メカニズムをバイパスする「裏口」ができる可能性を防止することが重要です。

Junction について **-w** オプションを使用することにより、以下の保護手段が得られます。

- 8.3 ファイル名形式の使用を防止する。

-w オプションを指定して `junction` を構成すると、ユーザーは、短い形式 (8.3) のファイル名を使用して、長いファイル名に対する明示 ACL を回避することはできなくなります。短い形式のファイル名を入力すると、サーバーは 403「禁止」エラーを戻します。

- ディレクトリー名およびファイル名の中で後書きドットを使用できないようにする。

ファイルまたはディレクトリーの名前に後書きドットが含まれていると、403「禁止」エラーが戻されます。

- **-i** オプションの設定により、大文字小文字を区別しないことを強制適用する。

-w オプションを指定すると、自動的に **-i** オプションが起動します。したがって、WebSEAL は、`junction` 先バックエンド・サーバーに対する要求について許可検査を行うときに、URL を、大文字小文字を区別しないものとして取り扱います。ACL 検査が成功すると、URL の元のケース (大文字小文字の別) が復元されてから、要求がバックエンド・サーバーに送られます。

注: ファイル名についてのみ大文字小文字を区別しないことを指定する必要がある場合は、`junction` に対して **-w** オプションの代わりに **-i** オプションのみを使用してください。

また、**-w** オプションは、仮想ホスト `junction` でもサポートされています。547 ページの『第 29 章 仮想ホスト `junction`』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト `junction`』を参照してください。

例

Windows 環境内に次のファイルがあるとします。

```
¥Program Files¥Company Inc¥Release.Notes
```

このファイルは、以下のパスを使用してもアクセスできます。

1. ¥progra~1¥compan~2¥releas~3.not
2. ¥Program Files¥Company Inc.¥Release.Notes
3. ¥program files¥company inc¥release.notes

例 1 は、Windows が、ファイル名にスペースを含まず、8.3 形式に準拠する別名 (DOS 互換性確保のため) を作成する方法を示しています。**-w** オプションを指定すると、WebSEAL は ACL 検査でこの形式を拒否します。

例 2 は、Windows がどのように後書き拡張子ドットを含めるかを示しています。**-w** オプションを指定すると、WebSEAL は ACL 検査でこの形式を拒否します。

例 3 は、Windows がファイル名に関する大文字小文字の区別を無視する方法を示しています。**-w** オプションを使用すると、同時に **-i** オプションも起動されるので、大文字小文字を区別しない ACL 検査が行われます。

小文字オブジェクト名への ACL および POP の付加

-w オプションまたは **-i** オプションを使用して **junction** が作成されると、WebSEAL は ACL および POP の比較を大文字小文字を区別しないものとして実行します。これは、ACL 用に評価されるすべてのオブジェクトの名前が小文字に変換されてから、WebSEAL が、これらのオブジェクトを、ACL が付加されるオブジェクト・リストに照らして検査することを意味します。

その結果、大文字が入っている名前を持つ保護オブジェクトは、ACL または POP 検査で検出されません。これらのオブジェクトが検出されない場合、保護オブジェクトに ACL または POP は適用されず、代わりに親ポリシーが適用されます。

この構成で、ポリシーが誤って適用されるのを防ぐには、明示的に ACL または POP を付加する実際の保護オブジェクトの名前の小文字バージョンを作成する必要があります。

仮想ホストへの標準 Junction

仮想ホスティングでは、同一の物理マシン上での複数の WebSEAL インスタンスの保守実施内容が参照されます。仮想ホスティングは、ホスト名および URL がそれぞれ異なり、完全に別のサイトに存在する可能性がある複数の Web サービスを実行するために使用されます。仮想ホスティングを使用するのは、ISP、ホスティング・サイト、および複数のサイトを管理する必要があるものの、それぞれのサイトに対して新規マシンの購入予定がないコンテンツ・プロバイダーです。複数の会社で 1 つの Web サーバーを共用できますが、**www.company1.com** および **www.company2.com** などの固有のドメインを所有しています。

仮想ホスト・ドメインのユーザーは、追加のパス情報を知る必要はありません。仮想ホスト・ドメインに対する要求では、ブラウザーは単純にその **Host:** ヘッダーを介して、要求されたホスト名を提供します。

標準的な **junction** 作成コマンドでは、**-h host** オプションおよび引数によって、リモート **junction** サーバー・マシンが識別されます。通常、WebSEAL によって、このサーバーに対する要求の **Host:** ヘッダーにこのサーバー・アドレスが配置されます。

追加の **-v host** オプションおよび引数によって、このマシンの仮想ホスト・インスタンスが識別されます。**-v** によって提供される仮想ホスト情報は、**-h** 情報の代わりに **Host:** ヘッダーで使用されます。これで、要求は特定の仮想ホスト・インスタンスに送信されます。**-h** 情報は、物理ホスト・マシンを検索するために、WebSEAL によって使用されます。

mutual の **Junction** の場合、**Junction** サーバーとの通信に HTTP プロトコルが使用されるとき、**-v host** オプションは仮想ホスト名に対応します。HTTPS プロトコルが使用されるとき、**-V host** オプションは仮想ホスト名として使用されます。

次の例に、2 つの仮想ホストの構成を示します。**junction** アプリケーション・サーバーは **server.xyz.com** です。このアプリケーション・サーバーはポート 80 上で実行され、**site1.xyz.com** および **site2.xyz.com** の 2 つの仮想サーバーのホストとして機能します。

基本 junction は、以下のように作成されます (各コマンドは 1 行として入力します)。

```
pdadmin> server task default-webseald-surf.xyz.com create -t tcp  
-h server.xyz.com -v site1.xyz.com /jct1  
pdadmin> server task default-webseald-surf.xyz.com create -t tcp  
-h server.xyz.com -v site2.xyz.com /jct2
```

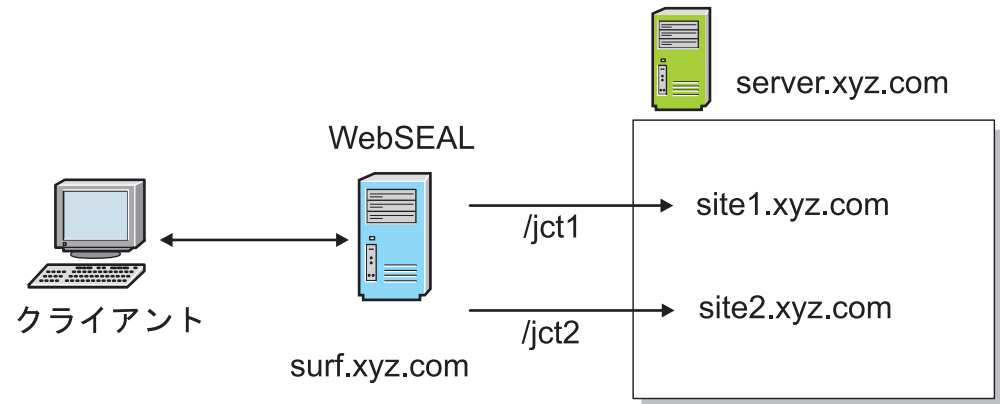


図 36. 仮想ホストの構成

672 ページの『-v を使用した完全な Host ヘッダー情報』も参照してください。

HTTP ヘッダー・データ用の UTF-8 エンコード

WebSEAL は、バックエンド・サーバーへの要求の HTTP ヘッダーに情報を挿入します。この情報には、拡張属性またはユーザー・データを組み込むことができます。バージョン 5.1 より前の WebSEAL バージョンでは、ロー・ローカル・コード・ページを使用して、ヘッダーが要求に追加されていました。WebSEAL バージョン 5.1 以降では、ヘッダー・データは、構成可能な形式で伝送されます。

デフォルトでは、WebSEAL は、このリリースから、UTF-8 エンコードを使用して情報を HTTP ヘッダーに追加するようになりました。このエンコードにより、非 UTF-8 コード・ページに変換する際に発生するデータ損失を防げるようになりました。また、このデータは、デフォルトで、URI でエンコードされて送られます。後方互換性を保つために、ヘッダー・データの形式は、ロー・ローカル・コード・ページに合わせて構成できます。さらに、ロー UTF-8 および URI エンコード・ローカル・コード・ページという 2 つの形式がサポートされます。

-e junction オプションは、HTTP ヘッダーに格納されてバックエンド・サーバーへ送信されるユーザー名、グループ、および他の拡張属性のエンコードを指定します。

また、**-e** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

-e エンコード・オプションは、以下の引数のいずれかを使用します。

引数	説明
utf8_uri	URI でエンコードされた UTF-8 データ。 すべての空白文字および非 ASCII バイトは %XY とエンコードされます。ここで、X および Y は 16 進値 (0 から F) です。
utf8_bin	エンコードされていない UTF-8 データ。 この設定により、データは、データ損失なしに伝送され、ユーザーはデータを URI でデコードする必要がありません。 この設定はよく注意して使用する必要があります。それは、この設定が HTTP 仕様の一部ではないからです。
lcp_uri	URI でエンコードされたローカル・コード・ページ・データ。 ローカル・コード・ページに変換できない UTF-8 文字はすべて疑問符 (?) に変換されます。このオプションはよく注意して、ローカル・コード・ページが望ましいストリングを生成する環境でのみ使用してください。
lcp_bin	エンコードされていないローカル・コード・ページ・データ。 このモードは、5.1 より前のバージョンの WebSEAL で使用されていたものです。このモードを使用すると前のバージョンからのマイグレーションを可能にするので、アップグレード環境で使用されます。 このモードではデータ損失が起こる可能性があるので、注意して使用してください。

WebSEAL による UTF-8 エンコードのサポートについて詳しくは、65 ページの『UTF-8 でのマルチロケール・サポート』を参照してください。

リソース単位でのバッファリングのバイパス

WebSEAL では、WebSEAL への要求と Junction アプリケーションからの応答で送信されるデータを処理する際に内部バッファを使用します。

このタスクについて

通常は、このバッファリングによりパフォーマンスが向上します。少量のデータを送信または返す特定のアプリケーションでは、バッファにデータを取り込む間に、バッファリングによってデータが WebSEAL に一時的に保留される場合があります。一部のアプリケーションでは、バッファリングをバイパスして、データを Junction サーバーまたはクライアントに直接ストリームの方が好ましい場合があります。このスキームは一般的な Web トラフィックには効率的ではありません。ストリーム・データが必要な特定のリソースにのみ適用してください。例えば、RPC over HTTP 通信用に構成された Junction に適用します。525 ページの『第 27 章 HTTP を介した Microsoft RPC』を参照してください。

保護オブジェクト・ポリシー (POP) を個々のリソースに適用することができます。これにより、それらのリソースのバッファリングをバイパスするように WebSEAL に指示されます。特定のリソース応答のバッファリングをバイパスするには、

response-buffer-control という名前の属性の値を **bypass** に設定して、POP をそのリソースに付加します。特定のリソース要求のバッファリングをバイパスするには、**request-buffer-control** という名前の属性の値を **bypass** に設定して、POP をそのリソースに付加します。

ここでは、次の作業例を示します。

- *bypassPOP* という名前の POP を作成する。
- **response-buffer-control** および **request-buffer-control** 属性を **bypass** に設定する。
- *smallCGI* という名前のリソースに POP を付加する。

手順

1. *bypassPOP* という名前の POP を作成し、適切な属性を設定します。

```
pdadmin> pop create bypassPOP
pdadmin> pop modify bypassPOP set attribute response-buffer-control bypass
pdadmin> pop modify bypassPOP set attribute request-buffer-control bypass
```

2. 選択したリソースに POP を付加します。

```
pdadmin> pop attach /WebSEAL/myinstance/myjunction/cgi-bin/smallCGI bypassPOP
```

この POP は、クライアントまたは Junction から受信される要求または応答の本体内のデータにのみ影響します。WebSEAL は引き続き要求および応答ヘッダーのバッファリングを行います。

この POP 技法を使用して HTTP 要求をバッファリングする場合、いくつかの制限があります。特定の WebSEAL 機能には要求本体の全体が必要であり、この本体は Junction サーバーへの要求のストリーミング時には使用できません。

要求 のストリーミング時に使用できない WebSEAL 機能は以下のとおりです。

注: WebSEAL の応答 ストリーミングは、この WebSEAL 機能を使用するリソースに引き続き適用可能です。

- 認証プロセス中の POST データのキャッシング。
- POST データが決定評価の一部である場合の動的許可決定情報 (dynADI)。
- POST データが決定評価の一部である場合の動的 URL (dynURL)。

junction を介したシングル・サインオン・ソリューション

587 ページの『第 31 章 junction を介したシングル・サインオン・ソリューション』を参照してください。

第 25 章 Junction リソースへの URL の変更

本章では、標準 WebSEAL junction を介してバックエンド・アプリケーション・サーバーから送信された URL リンクが、クライアントでの使用向けに適切に再構成されたことを確認するために使用可能な構成オプションについて説明します。

URL のフィルター操作の実行は標準 WebSEAL junction に限定されます。また、WebSEAL では仮想ホスティングがサポートされ、仮想ホスト junction を介することによって、URL フィルター操作の制限がなくなります。WebSEAL では、クライアント要求に HTTP **Host** ヘッダーを使用して、それらの要求を junction サーバーまたはローカル・マシン上にある適切な文書スペースに送信します。

仮想ホスト junction の使用について詳しくは、547 ページの『第 29 章 仮想ホスト junction』を参照してください。

この章で扱うトピックは以下のとおりです。

- 『URL の変更の概念』
- 476 ページの『URL で使用されるパスのタイプ』
- 478 ページの『応答の中の URL の変更』
- 490 ページの『要求の中の URL の変更』
- 501 ページの『複数の -j junction にわたるサーバーからの Cookie の処理』

URL の変更の概念

ほとんどの場合、バックエンド・アプリケーション・サーバーからクライアントに戻されるページには、そのアプリケーション・サーバーが持っているリソースへのリンクが含まれています。このような URL は、すべてのクライアントの要求を適切なリソースに導くように作成することが重要です。

例えば、非 WebSEAL 環境でクライアントがアプリケーション・サーバーのリソースを要求するときに、次のように URL を入力します。

```
http://www.example.com/file.html
```

フロントエンド逆プロキシとしての WebSEAL は、WebSEAL junction 機能を使用して、バックエンド・アプリケーション・サーバーにセキュリティー・サービスを提供します。この機能では、これらのリソースへの元の URL が junction 情報を含むように変更される必要があります。

WebSEAL の標準 junction 機能は、junction 先バックエンド・システム上のリソースにアクセスするために必要なサーバーおよびパスの情報を変更します。URL に junction の識別が含まれていないと、junction バックエンド・サーバー上のリソースへのリンクは失敗します。

このバックエンド・サーバーが WebSEAL 環境の junction サーバーの場合、junction バックエンド・アプリケーション・サーバー上の同じリソースにアクセスする際に使用する URL は、次のようである必要があります。

`http://webseal.example.com/jct/file.html`

標準 junction 機能をサポートし、URL の安全性を維持するために、WebSEAL は可能な限り以下のことを行う必要があります。

1. クライアントに送信される**応答**の中の URL (リンク) を変更する。
2. WebSEAL が変更できなかった URL (リンク) に基づいて発生するリソース要求を変更する。

注: WebSEAL が URL を変更するときのルールとメカニズムは、Security Access Manager の junction 環境の外部にあるリソースを指すリンクには適用されません。

次の図は、WebSEAL が junction 先バックエンド・リソースへの URL を変更するときに行えるソリューションを要約したものです。

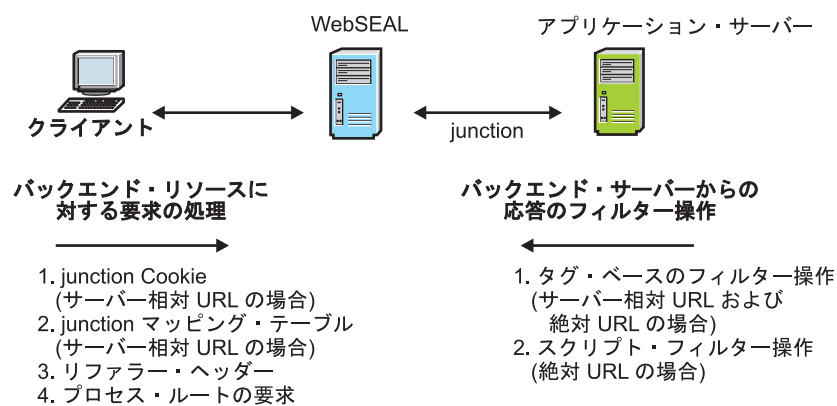


図 37. 要約: バックエンド・リソースへの URL の変更

URL で使用されるパスのタイプ

HTML ページには、そのページを提供するバックエンド・サーバーまたはその他の場所にある他のリソースへの URL (リンク) が含まれていることがよくあります。URL 表記は、以下の形式で表されます。

- 相対
- サーバー相対
- 絶対

相対形式で表現された URL を含むリンクは、WebSEAL による変更を何も必要としません。デフォルトでは、ブラウザは、リンクの中の相対 URL を扱うときに、その URL の前に、正しいスキーム (プロトコル)、サーバー名、およびディレクトリー情報 (junction を含む) を付加します。ブラウザによって、リンクが置かれているページのロケーション情報から、先頭に付加する情報が導き出されます。

相対 URL 表記の例:

```
abc.html
../abc.html
./abc.html
sales/abc.html
```


しかし、**サーバー相対**および**絶対**パス形式の場合は、上記とは取り扱いが異なります。

サーバー相対 URL には、絶対パス形式 (/ で開始します) が含まれますが、スキーム (プロトコル) またはサーバー名に対して付加されます。**サーバー相対** URL 表記の例:

```
/abc.html  
/accounts/abc.html
```

絶対 URL には、スキーム (プロトコル)、サーバー名、および絶対パスが含まれます。**絶対** URL 表記の例:

```
http://www.example.com/doc/abc.html
```

絶対形式またはサーバー相対形式で表されたバックエンド・リソースへのリンクは、WebSEAL が、junction 情報を組み込むように URL パス形式を変更できない場合は失敗します。WebSEAL URL の変更手法は、絶対 URL およびサーバー相対 URL に適用されます。

注: リソースの正常なロケーションを確実にするには、Web スクリプトのあらゆるプログラマーには、動的に生成される URL に対して、**相対**リンク (絶対リンクでもサーバー相対リンクでもない) を使用することをお勧めします。

URL の特殊文字

HTML ページには ASCII 特殊文字を含めることができます。Web コンテンツには、特殊文字または記号を表す HTML コードが含まれます。

特殊文字の HTML コードは「&#」で始まり、「;」で終わります。

HTML コードでは以下の形式を使用します。

```
&#<numeric_id>;
```

ここで、

```
<numeric_id>
```

文字の数値または 16 進表記。数値表記の最大値は 255 (16 進: 0xFF) です。

例えば、パーセント記号 (%) の場合、数値コードは %、16 進コードは % となります。

注: HTTP 応答に存在する特殊文字エンコードの WebSEAL でのフィルター方法について詳しくは、481 ページの『エンコードまたはエスケープされた URL の変更』を参照してください。

応答の中の URL の変更

このセクションでは、junction バックエンド・アプリケーション・サーバーからの応答の中の URL を変更するためのオプションについて説明します。

- 『タグ・ベースの静的 URL のフィルター操作』
- 487 ページの『スクリプトのフィルター操作による絶対 URL の変更』
- 488 ページの『rewrite-absolute-with-absolute オプションの構成』
- 489 ページの『フィルター操作による Content-Length ヘッダーの変更』
- 489 ページの『フィルター掛けされていないサーバー相対リンクでの制限』

タグ・ベースの静的 URL のフィルター操作

WebSEAL は、一組のデフォルト・ルールに従って、クライアント要求への応答として戻されるページに含まれているタグ・ベースの静的 URL をスキャン (またはフィルター操作) に掛けます。このデフォルトのフィルター操作メカニズムによって、タグ・ベース・コンテンツ (HTML または XML など) 内の静的 URL が調べられます。

用語「フィルター操作」は、Web 文書内に絶対リンクおよびサーバー相対リンクがあるかどうかスキャンし、このリンクに junction 情報を組み込むように変更する WebSEAL のプロセスを指します。

このメカニズムでは、URL が WebSEAL に対して可視 URL である必要があります。例えば、タグ・ベースのコンテンツのフィルター操作では、クライアント・サイドで動的に生成される URL を処理することはできません。

preserve-base-href2 オプションも「yes」に設定されている場合、WebSEAL は WebSEAL ホストおよび junction の名前を挿入するために必要な BASE HREF タグについて、最小限のフィルター操作のみを実行します。**preserve-base-href2** オプションの値は、後続スラッシュがないすべての BASE HREF タグの処理に影響します。

備考: **preserve-base-href** が「no」に設定されている場合、このオプションは無効です。

例 5 (後続スラッシュを含まない BASE HREF と junction 先サーバーとの一致)

- WebSEAL サーバー: `www.webseal.com`
- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML

注: BASE HREF 値に後続スラッシュは含まれていません。

```
<HTML>
<HEAD>
<BASE HREF="http://www.example.com">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

preserve-base-href2 が「no」に設定されている場合の、フィルター操作後の HTML

注: 後続スラッシュがないため、WebSEAL は HREF を /jct にマップしてから /jct を除去します。

```
<HTML>
<HEAD>
<BASE HREF="http://www.webseal.com/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

preserve-base-href2 が「yes」に設定されている場合の、フィルター操作後の HTML

注: WebSEAL は WebSEAL ホストおよび junction の名前を挿入するために必要な BASE HREF タグについて、最小限のフィルター操作を実行します。後続スラッシュがない場合、基本 HREF URL 内の最終の構成要素は除去されません。

```
<HTML>
<HEAD>
<BASE HREF="http://www.webseal.com/jct/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

このセクションは、以下のトピックで構成されています。

- ・ 『タグ・ベースの静的 URL のフィルター操作ルール』
- ・ 480 ページの 『タグ・ベースの静的 URL のデフォルト・フィルター操作』
- ・ 482 ページの 『新規コンテンツ (MIME) タイプのフィルター操作の構成』
- ・ 482 ページの 『タグ・ベースのフィルター操作に対するタグおよび属性』
- ・ 483 ページの 『HTML META タグ』
- ・ 483 ページの 『HTML BASE HREF タグ』
- ・ 486 ページの 『BASE タグを使用するページで無視するスキーム』

タグ・ベースの静的 URL のフィルター操作ルール

相対 URL のフィルター操作のルール:

相対 URL は、常にブラウザにより適切に取り扱われます。したがって、WebSEAL は相対 URL に対してフィルター操作を実行しません。

デフォルトでは、ブラウザは、正しいスキーム、サーバー、およびディレクトリ情報 (junction を含む) を相対 URL の前に付加します。前に付加する情報は、リンクが置かれているページを要求する URL から導き出されます。

サーバー相対 URL のフィルター操作のルール:

WebSEAL は、junction 先サーバー上のリソースを指すサーバー相対 URL のパスには、junction 名を付加する必要があります。

サーバー相対 URL は、次の例のように、junction 先サーバーの文書ルートに関連する URL 位置を示します。

```
/dir/file.html
```

サーバー相対 URL は、パス名に junction 先サーバーの junction ポイントが追加されて変更されます。以下に例を示します。

```
/jct/dir/file.html
```

絶対 URL のフィルター操作のルール:

WebSEAL は、junction 先サーバー上のリソースを指す絶対 URL のパスには、junction 名を付加する必要があります。

絶対 URL は、ホスト名または IP アドレス (およびオプションでネットワーク・ポート) に関連する URL 位置を示します。例:

```
http://host-name[:port]/file.html  
https://host-name[:port]/file.html
```

絶対 URL は、以下の一連のルールに従って変更されます。

- URL が HTTP であって、ホストかポートが junction 先 TCP サーバーに一致する場合は、URL は変更されて WebSEAL に対してサーバー相対になり、junction ポイントを反映します。以下に例を示します。

```
http://host-name[:port]/file.html
```

これは次のように変更されます。

```
/tcpjct/file.html
```

- URL が HTTPS であって、ホストまたはポートが junction 先 SSL サーバーに一致する場合は、URL は変更されて WebSEAL に対してサーバー相対になり、junction ポイントを反映します。以下に例を示します。

```
https://host-name[:port]/file.html
```

これは次のように変更されます。

```
/ssljct/file.html
```

488 ページの『rewrite-absolute-with-absolute オプションの構成』も参照してください。

代替の絶対 URL フィルター操作メカニズムについては、487 ページの『スクリプトのフィルター操作による絶対 URL の変更』を参照してください。

タグ・ベースの静的 URL のデフォルト・フィルター操作

WebSEAL では、WebSEAL 構成ファイルの **[filter-content-types]** スタンザでコンテンツ (MIME) タイプが指定されたページにあるタグ・ベースの静的 URL に対してフィルター操作を実行します。

デフォルトでは、このスタンザによって MIME タイプ `text/html` および `text/vnd.wap.wml` が指定されます。

```
[filter-content-types]  
type = text/html  
type = text/vnd.wap.wml
```

WebSEAL 構成ファイルの `[filter-url]` スタンザによって、コンテンツでフィルターが掛けられるタグおよびタグ属性が指定されます。最も一般的に使用される HTML タグおよびタグ属性がデフォルトでこのスタンザにリストされるため、通常、デフォルトの `text/html` および `text/vnd.wap.wml` MIME タイプのタグ・ベースのフィルター操作は追加の構成を実行しなくても動作します。

エンコードまたはエスケープされた URL の変更

WebSEAL は、出力応答内の URL 参照を変更して書き換えることにより、後続の要求が WebSEAL 経由で送信されるようにします。

WebSEAL は、URI エンコードやスラッシュによるエスケープを使用するオリジナルの形式で、各 URL を書き換えます。WebSEAL は特殊文字エンコードを検出すると、HTML コードが表す特殊文字でそのコードを置き換えます。

以下の表は、WebSEAL がこの処理においてフィルタリングを行う URI エンコード・タイプを示しています。

表 36. フィルター対象のエンコード・タイプ: WebSEAL は以下のエンコード・タイプをフィルタリングします。

エンコード・タイプ	エンコードされた URL の例
& 記号エンコード	HTTP://host:port/path?V1=D1 &V2=D2
& 記号 - 16 進エンコード	HTTP://host:port /
& 記号 - 10 進エンコード	HTTP:99host:port9
円記号 (¥) エンコード	HTTP:¥/¥/host:port¥/
% 16 進エンコード	HTTP%3A%2F%2Fhost%3Aport%2F

例: URL のフィルタリング

- WebSEAL サーバー: `www.webseal.com`
- 以下の `junction` が作成されています。
`server task webseal-server create -tcp -h www.example.com /jct`

WebSEAL フィルタリング処理前の HTML:

```
<HTML>
<BODY>
<A HREF="http%3A%2F%2Fwww.example.com%2Findex.html">index.html</A>
<A HREF="http:¥/¥/www.example.com¥/page2.html">page2.html</A>
<A HREF="http://www.example.com/page3.html">page3.html</A>
<A HREF="http&#x3a;&#x2f;&#x2f;www.example.com&#x2f;page4.htm">page4.html</A>
</BODY>
</HTML>
```

構成項目 `[script-filtering]` `rewrite-absolute-with-absolute` が `no` に設定された WebSEAL フィルタリング処理後の HTML:

```
<HTML>
<BODY>
<A HREF="%2Fjct%2Findex.html">index.html</A>
<A HREF="¥/jct¥/page2.html">page2.html</A>
```

```
<A HREF="/jct/page3.html">page3.html</A>
<A HREF="#x2F;jct&x2F;page4.htm">page4.html</A>
</BODY>
</HTML>
```

構成項目 [script-filtering] rewrite-absolute-with-absolute が yes に設定された WebSEAL フィルタリング処理後の HTML:

```
<HTML>
<BODY>
<A HREF="http%3A%2F%2Fwww.webseal.com%2Fjct%2Findex.html">index.html</A>
<A HREF="http:%2F/www.webseal.com%2Fjct%2Fpage2.html">page2.html</A>
<A HREF="http://www.webseal.com/jct/page3.html">page3.html</A>
<A HREF="http&x3A;&x2F;&x2F;www.webseal.com&x2F;jct&x2F;page4.htm">
page4.html</A>
</BODY>
</HTML>
```

新規コンテンツ (MIME) タイプのフィルター操作の構成

追加の MIME タイプは、該当するエントリを [filter-content-types] スタンザに追加することによって、URL のフィルター操作に対して構成できます。

[filter-content-types] スタンザに追加される新規 MIME タイプのデフォルトのフィルター操作メカニズムは、スクリプトのフィルター操作です。スクリプトのフィルター操作を有効にするには、WebSEAL 構成ファイルを編集する必要があります (スクリプトのフィルター操作はデフォルトでは無効になっています)。

```
[script-filtering]
script-filter = yes
```

スクリプトのフィルター操作メカニズムによって、応答の内容が調べられます。また、このメカニズムはタグ・ベースのコンテンツに制限されません。ただし、スクリプトのフィルター操作によって確認および変更されるのは、絶対 URL のみです。487 ページの『スクリプトのフィルター操作による絶対 URL の変更』を参照してください。

また、[filter-content-types] スタンザに追加された新規 MIME タイプに対して、静的 URL のタグ・ベースのフィルター操作を有効にできます。新規に構成された MIME タイプに対してタグ・ベースの静的 URL フィルター操作を有効にするには、以下のように WebSEAL 構成ファイルの [server] スタンザ内の filter-nonhtml-as-xhtml スタンザ・エントリを「yes」に設定します。

```
[server]
filter-nonhtml-as-xhtml = yes
```

『タグ・ベースのフィルター操作に対するタグおよび属性』で説明されているように、追加のタグおよびタグ属性がフィルタリングされるように指定します。

タグ・ベースのフィルター操作に対するタグおよび属性

タグ・ベースの静的 URL フィルター操作は、WebSEAL 構成ファイルの [filter-url] スタンザで定義されている特定のタグおよび属性に制限されています。フィルター操作メカニズムによって、指定のタグおよびタグ属性の範囲内の URL のみが検索されます。

追加のタグおよびタグ属性にフィルターが掛けられるように指定するために、エントリーを **[filter-url]** スタンザに追加することができます。タグおよびタグ属性には、HTML、XML、XHTML、またはカスタム・データ定義が含まれます。

HTML META タグ

HTML META リフレッシュ・タグは必ずフィルター処理されます。以下に例を示します。

```
<META HTTP-EQUIV="Refresh" CONTENT="5;URL=http://server/url">
```

HTML BASE HREF タグ

[server] スタンザの **preserve-base-href** および **preserve-base-href2** エントリーを使用して、フィルター操作された HTML 文書の HTML BASE タグの HREF 属性を WebSEAL で処理する方法を制御できます。

注: **preserve-base-href** オプションが「yes」に設定されていない場合、**preserve-base-href2** の値は無効です。

preserve-base-href が「no」に設定されている場合は、以下の記述が適用されます。

- WebSEAL によってフィルター済み HTML 文書から BASE HREF 属性が削除されます。
- BASE HREF URL が junction サーバーと一致すると、WebSEAL によって、ページ上で検出されたサーバー相対リンクおよび相対リンクの前に junction ポイントおよびすべてのサブディレクトリが付加されます。
- BASE HREF URL と junction サーバーが一致しない場合、WebSEAL によって、ページ上のサーバー相対リンクおよび相対リンクの前に BASE HREF URL 属性全体が付加されます。このアクションは、ブラウザーによって HTML がレンダリングされた際に実行されます。

例 1 (BASE HREF と junction サーバーが一致)

- WebSEAL サーバー: www.webseal.com
- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example.com/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE>
</HEAD>
```



```
<BODY>
<A HREF="/jct/dir1/dir2/index.html">index.html</A>
</BODY>
</HTML>
```

例 2 (BASE HREF が junction サーバーと一致していない)

- WebSEAL サーバー: `www.webseal.com`

- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example2.com/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE>
</HEAD>
<BODY>
<A HREF="http://www.example2.com/dir1/dir2/index.html">index.html</A>
</BODY>
</HTML>
```

preserve-base-href が「yes」に設定されている場合は、以下の記述が適用されます。

- WebSEAL によってフィルター済み HTML 文書から BASE HREF 属性は削除されません。
- BASE HREF URL が junction サーバーと一致する場合、WebSEAL は BASE HREF URL を変更します。WebSEAL は、junction サーバーの名前を WebSEAL サーバー名で置き換え、junction サーバーの junction ポイントを置き換えます。ブラウザがその HTML をレンダリングし、変更された HREF URL をサーバー相対リンクまたは相対リンクの先頭に付加すると、結果のリンクでリソースを見つけることができます。
- BASE HREF URL と junction サーバーが一致しないと、WebSEAL によって HTML は変更されません。ブラウザがその HTML をレンダリングし、元の HREF URL をサーバー相対リンクまたは相対リンクの先頭に付加すると、結果のリンクでリソースを見つけることができます。
- WebSEAL は、BASE HREF タグを含む HTML 文書のフィルター操作を行うとき、次の文字が含まれていても、URL の元のエスケープまたはエンコードを保持しません。
 - URI エンコード
 - エスケープされたスラッシュ
 - 特殊文字エンコード

例 3 (URI エンコード、スラッシュ・エスケープ、および特殊文字の URL)

- WebSEAL サーバー: `www.webseal.com`

- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example2.com/dir1/">
</HEAD>
<BODY>
<A HREF="dir2%2Findex.html">index.html</A>
<A HREF="dir2¥/contents.html">contents.html</A>
<A HREF="dir2&#x2F;index2.html">index2.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE>
</HEAD>
<BODY>
<A HREF="http://www.example2.com/dir1/dir2/index.html">index.html</A>
<A HREF="http://www.example2.com/dir1/dir2/contents.html">contents.html</A>
<A HREF="http://www.example2.com/dir1/dir2/index2.html">index2.html</A>
</BODY>
</HTML>
```

例 4 (BASE HREF と junction サーバーが一致)

- WebSEAL サーバー: `www.webseal.com`

- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example.com/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.webseal.com/jct/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

例 5 (BASE HREF が junction サーバーと一致していない)

- WebSEAL サーバー: `www.webseal.com`

- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example2.com/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example2.com/dir1/dir2/">
</HEAD>
<BODY>
<A HREF="index.html">index.html</A>
</BODY>
</HTML>
```

例 6 (URI エンコード、スラッシュ・エスケープ、および特殊文字の URL)

- WebSEAL サーバー: `www.webseal.com`
- 以下の junction が作成されています。

```
server task webseal-server create -tcp -h www.example.com /jct
```

フィルター操作前の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.example.com/dir1/">
</HEAD>
<BODY>
<A HREF="dir2%2Findex.html">index.html</A>
<A HREF="dir2%/contents.html">contents.html</A>
<A HREF="dir2&#X2F;index2.html">index2.html</A>
</BODY>
</HTML>
```

フィルター操作後の HTML:

```
<HTML>
<HEAD>
<BASE HREF="http://www.webseal.com/jct/dir1/">
</HEAD>
<BODY>
<A HREF="dir2/index.html">index.html</A>
<A HREF="dir2/contents.html">contents.html</A>
<A HREF="dir2/index2.html">index2.html</A>
</BODY>
</HTML>
```

BASE タグを使用するページで無視するスキーム

[filter-schemes] スタンザには、junction アプリケーションからの応答時に WebSEAL によってフィルターが掛けられない URL スキームがリストされています。スキームはプロトコル ID です。このリストは、HTML BASE タグに HREF 属性を含む文書を WebSEAL が検出したときに使用されます。

[filter-schemes] スタンザにはデフォルトのスキームのセットが含まれています。

- ユーザーは、追加プロトコルを使用してリストを拡張できます。

- リストからエントリーを削除しないことをお勧めします。
- 「http」および「https」スキームは、常にフィルターが掛けられるようにハードコーディングされています。
- エントリーをスタンザに追加する際のスキーム名の末尾のコロン (:) はオプションです。コロンが付与されていない場合、WebSEAL はそれを想定します。

デフォルトのスキーム・エントリー:

```
[filter-schemes]
scheme = file
scheme = ftp
scheme = mailto
scheme = news
scheme = telnet
```

応答の中の URL でこのスタンザからのスキームが使用されず、スキームが「http」または「https」でない場合、WebSEAL では、スキームが欠落している junction サーバー (HTTP: または HTTPS:) と URL が同じスキームであると推測します。

例外 スタンザからのスキームを使用する HREF URL 属性を持つ HTML BASE タグが応答に含まれている場合は、WebSEAL によって URL がフィルター処理されます。

スクリプトのフィルター操作による絶対 URL の変更

このタスクについて

スクリプトに組み込まれている絶対 URL を処理できるようにするには、WebSEAL の追加構成が必要です。Web スクリプト言語には、JavaScript、VBScript、ASP、JSP、ActiveX などがあります。WebSEAL 構成ファイルの **[script-filtering]** スタンザの中の **script-filter** スタンザ・エントリーによって、組み込み絶対 URL のフィルター操作が有効または無効になります。スクリプトのフィルター操作はデフォルトでは無効になっています。

```
[script-filtering]
script-filter = no
```

手順

スクリプトのフィルター操作を有効にするには、以下のようにこのスタンザ・エントリーの値を「yes」に設定します。

```
[script-filtering]
script-filter = yes
```

スクリプトのフィルター操作メカニズムによって、応答の内容全体が調べられます。また、このメカニズムはタグ・ベースのコンテンツなどに制限されません。

script-filter メカニズムは、次のような、標準のスキーム、サーバー、リソース形式を持つ絶対 URL を想定しています。

`http://server/resource`

script-filter メカニズムは、リンクのスキームおよびサーバー部分を正しい junction 情報で置き換えます (相対パス名の形にします)。

`/junction-name/resource`

このフィルター・ソリューションでは、HTML コードに組み込まれているスクリプトの構文が解析されるため、処理オーバーヘッドが増加し、パフォーマンスが低下することがあります。設定はすべての junction に適用されます。WebSEAL 環境で組み込み絶対 URL のフィルター操作が要求される際にのみ **script-filter** スタンザ・エントリーを有効にします。

次の図は、この絶対 URL フィルター・ソリューションを示しています。

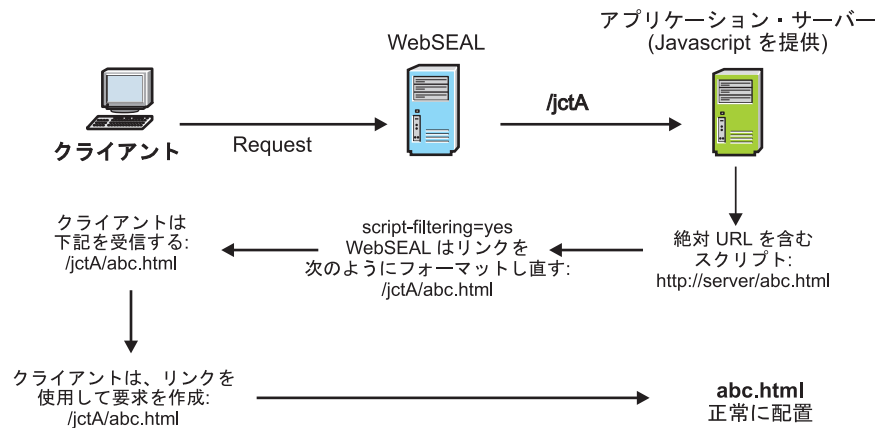


図 38. 絶対 URL のフィルター操作

rewrite-absolute-with-absolute オプションの構成

このタスクについて

通常、WebSEAL では、junction ポイントを追加し、形式をサーバー相対表記に変更することによって絶対 URL をフィルターに掛けます。絶対 URL をフィルターに掛けるためのこのルールは、タグ・ベースのフィルター操作およびスクリプト・フィルター操作に適用されます。オプションで、元の絶対 URL を相対 URL ではなく絶対 URL に書き直すように、WebSEAL を構成することもできます。

手順

この種類のフィルター操作を使用可能にするには、WebSEAL 構成ファイルの **[script-filtering]** スタンザ内の **rewrite-absolute-with-absolute** スタンザ・エントリーを「yes」に設定します。

```
[script-filtering]
rewrite-absolute-with-absolute = yes
```

rewrite-absolute-with-absolute が使用可能な場合、以下のバックエンド・サーバーからの応答の中の URL (WebSEAL jctA を介して WebSEAL に接続)

```
http://server/abc.html
```

は、以下のように変更されます。

```
http://webseal-hostname/jctA/abc.html
```

rewrite-absolute-with-absolute を使用可能にすると、タグ・ベースのフィルター操作およびスクリプト・フィルター操作が影響を受けます。以下を参照してください。

- 487 ページの『スクリプトのフィルター操作による絶対 URL の変更』
- 479 ページの『タグ・ベースの静的 URL のフィルター操作ルール』

フィルター操作による Content-Length ヘッダーの変更

このタスクについて

通常、バックエンド・サーバーからの応答の中の **Content-Length** ヘッダーは、戻されるコンテンツのサイズを示します。WebSEAL が URL をフィルターに掛けて、ページに含まれている URL のパスに junction 情報を付加すると、ページの実際のサイズは **Content-Length** ヘッダーが示すサイズより大きくなります。

WebSEAL は、実際にクライアントにストリームを書き込むまでは、新しいコンテンツの長さを知ることはできません。しかし、その時点では、新しい **Content-Length** ヘッダーを挿入するには遅すぎます。WebSEAL は、以下の方法でこの状況に対処します。

手順

1. WebSEAL は、元の **Content-Length** ヘッダーの値を **X-Old-Content-Length** という新しいヘッダーに入れます。

このヘッダーを検索するように作成されているアプレットまたはアプリケーションは、元の（フィルター操作の前の）**Content-Length** の値を知ることができます。

2. WebSEAL は、変更済みの（フィルター操作後の）**Content-Length** の値を `request.log` ファイルに記録します。
3. **Content-Length** ヘッダーは現れなくなります。

フィルター掛けされていないサーバー相対リンクでの制限

問題

WebSEAL は、クライアント・サイドのスクリプトによって生成されたサーバー相対 URL の処理のためのソリューションを、バックエンドの junction 先アプリケーション・サーバー上のリソースに提供します。クライアント・サイドでアプレットおよびスクリプトにより生成されたサーバー相対 URL には、最初は、パス表現の中に Junction ポイントの情報が入っていません。クライアントがリソースを要求したときは、WebSEAL は、junction Cookie または junction マッピング・テーブルを使用して、サーバー相対 URL の再処理を試みることができます。

ただし、この処理が行われる前に、実際には、要求は、WebSEAL サーバー自体のローカル Web スペースにあるリソースを指定します。URL の修正再処理は、WebSEAL が要求を受け取り、ACL チェックを実行してからでなければ行われません。

誤ったローカル・リソースまたは存在しないローカル・リソースを指定する未処理要求の ACL チェックが行われると、エラーが発生する場合があります。このエラーにより要求が停止される可能性があります。

例えば、処理中には、次のようなシーケンスが発生します。

1. クライアントが、クライアント・サイドの、スクリプト生成のサーバー相対 URL を使用して、リソースの要求をします。
2. サーバー相対 URL が WebSEAL によって、要求として受け取られます。

未処理の URL は、WebSEAL サーバー自体の Web スペースにあるリソースを指定します (ただし、これは、明らかに意図したリソースではありません)。

3. WebSEAL は、要求 URL に指定されているこのローカル・リソースに対して ACL チェックを実行します。
 - この ACL チェックが失敗した場合は、要求の処理はすべて停止し、クライアントは 403 エラー (禁止) を受け取ります。このエラーが発生したのは、ACL チェックが誤りの (多くの場合、存在していない) リソースに実行されたためです。
 - ACL チェックが成功し、リソースがローカル Web スペースにある場合は、このリソースが戻されます。このエラーの結果、クライアントは誤ったリソースを受け取ります。
 - ACL チェックが成功し、リソースがローカル Web スペースにない場合は、WebSEAL は要求 URL を変更し (Junction Cookie または junction マッピング・テーブル・メソッドを使用)、要求を内部で再処理します。これは正しい振る舞いです。
4. WebSEAL は、Junction ポイントが組み込まれている訂正済みパスが入った変更済み URL に対して、もう一度 ACL チェックを実行します。この変更済み URL を使用することによって、正しいリソースに ACL チェックを行えるようになりました。

対応策:

この問題を解決するには、次のようにします。

1. 常時、相対 URL リンクを生成するスクリプトを作成してください。絶対リンクおよびサーバー相対 URL は作成しないでください。
2. サーバー相対リンクを使用しなければならない場合は、同じリソース名とパスを、WebSEAL サーバーと junction 先アプリケーション・サーバーの両方で使用しないでください。
3. サーバー相対リンクを使用しなければならない場合は、禁止が強い ACL が、フィルター掛けされていない URL で指定される偽リソースに影響しないように ACL モデルを設計してください。

要求の中の URL の変更

URL がクライアント・サイド・アプリケーション (アプレットなど) により動的に生成される場合、または HTML コード内のスクリプトに組み込まれている場合は、障害が生じます。Web スクリプト言語には、JavaScript、VBScript、ASP、JSP、ActiveX などがあります。これらのアプレットおよびスクリプトは、ページがクライアント・ブラウザに到着すると実行されます。したがって、WebSEAL は、クライアント・サイドで動的に生成された URL に対しては標準フィルター・ルールを適用する機会がありません。

要求内の URL を変更するための 3 つのオプションが WebSEAL で使用可能であり、以下に示す優先順位に従って適用されます。

1. junction マッピング・テーブル
2. junction Cookie
3. HTTP リファラー・ヘッダー

このセクションでは、クライアント・サイドで動的に生成された **server-relative** リンク (junction バックエンド・アプリケーション・サーバーのリソースに対して要求を実行するために使用される) を処理するためのオプションについて説明します。

注: クライアント・サイドで生成された**絶対 URL** を取り扱うためのソリューションはありません。

この章で扱うトピックは以下のとおりです。

- 『junction マッピングによるサーバー相対 URL の変更』
- 493 ページの 『junction Cookie を使用したサーバー相対 URL の変更』
- 495 ページの 『junction Cookie JavaScript ブロックのコントロール』
- 498 ページの 『HTTP Referer ヘッダーを使用したサーバー相対 URL の変更』
- 499 ページの 『要求の中のサーバー相対 URL 処理のコントロール』

junction マッピングによるサーバー相対 URL の変更

クライアント・サイドでアプレットおよびスクリプトにより生成されたサーバー相対 URL には、最初は junction ポイントの情報は組み込まれていません。この URL はクライアント・サイドで生成されているため、WebSEAL はこの URL をフィルターに掛けることができません。

クライアントがこの URL を使用してリソースを要求したときは、WebSEAL は、junction マッピング・テーブルを使用してサーバー相対 URL の再処理を試みることができます。junction マッピング・テーブルでは、特定のターゲット・リソースが junction 名にマッピングされます。junction マッピングは、動的に生成されたサーバー相対 URL をフィルターに掛けるための、Cookie ベースのソリューションに代わる方法です。

WebSEAL は、サーバー相対 URL のロケーション情報を、junction マッピング・テーブルに入っているデータに付き合わせて検査します。WebSEAL は、テーブルの先頭から検索を開始し、下方向にテーブル全体を検索します。このトップダウン検索中に URL 内のパス情報がテーブル内のエントリーと一致する場合には、WebSEAL は、そのロケーションに関連する junction に要求を送信します。

このテーブルは、jmt.conf という ASCII テキスト・ファイルです。このファイルの名前は、WebSEAL 構成ファイルの **[junction]** スタンザに指定されています。

```
jmt-map = jmt.conf
```

テーブル内のデータ・エントリーの形式は、junction 名、スペース、およびリソース・ロケーション・パターンから構成されます。リソース・ロケーション・パターンは、ワイルドカード文字を使用して表現することもできます。

Junction マッピング構成ファイルの次の例では、2 つのバックエンド・サーバーが、**/jctA** と **/jctB** にある WebSEAL に junction されます。

```
#jmt.conf
#junction-name resource-location-pattern
/jctA /documents/release-notes.html
/jctA /travel/index.html
/jctB /accounts/*
/jctB /images/weather/*.jpg
```

jmt.conf マッピング・テーブルを作成する必要があります。このファイルは、デフォルトでは存在していません。ファイルを作成しデータを追加したら、WebSEAL に新規情報の知識を持たせるため、**jmt load** コマンドを使用して、データをロードしなければなりません。

```
pdadmin> server task server-name jmt load
JMT table successfully loaded.
```

以下の条件が、junction マッピング・テーブル・ソリューションに適用されます。

- junction マッピング・ソリューションでは、WebSEAL によってインターセプトされたインバウンド要求を処理します。WebSEAL 環境の外部のサーバーを示すフィルター操作前の絶対 URL を使用して実行される要求 (したがって、WebSEAL によってインターセプトされていない要求) は junction マッピング・テーブル・ソリューションでは処理されません。
- このソリューションには、**-j** オプションも junction Cookie も必要ありません。
- マッピング・テーブルは、セキュリティー・アドミニストレーターによるセットアップとアクティブ化が必要です。
- リソース・ロケーション・パターン・マッチングは、ローカル Web スペース全体にわたって固有でなければならず、また junction 先 Web アプリケーション・サーバー全体にわたっても固有でなければなりません。
- ファイルに重複したパターン・エントリーがある場合には、マッピング・テーブルはロードされません。ただし、WebSEAL は実行を継続します。
- マッピング・テーブルのロードでエラーがある場合には、マッピング・テーブルは利用できません。ただし、WebSEAL は実行を継続します。
- マッピング・テーブルが空であるか、テーブル・エントリーにエラーがある場合には、マッピング・テーブルはロードされません。ただし、WebSEAL は実行を継続します。
- マッピング・テーブルをロードする際にエラーが発生すると、WebSEAL サーバー・ログ・ファイル (webseald.log) に保守容易性のエントリーが記録されます。
- デフォルトで、WebSEAL は、junction マッピング・テーブルにリストされている、複数の junction にわたる非ドメイン Cookie (バックエンド・アプリケーションからの応答の中で戻される) の名前を変更します。WebSEAL は固有の Cookie の名前を作成して、その他の junction から戻される Cookie とのネーミング競合を回避するようにします。この機能を無効にする方法は 2 つあります。

501 ページの『複数の -j junction にわたるサーバーからの Cookie の処理』を参照してください。

499 ページの『要求の中のサーバー相対 URL 処理のコントロール』も参照してください。

junction Cookie を使用したサーバー相対 URL の変更

このセクションは、以下のトピックで構成されています。

- 『junction Cookie の概念』
- 494 ページの『junction Cookie をサポートする WebSEAL junction の構成』

junction Cookie の概念

バックエンドの junction アプリケーション・サーバーからの HTML ページには、クライアント・サイドのサーバー相対リンクを動的に生成する組み込みアプレットまたはスクリプトが含まれる場合があります。これらの URL はクライアント・サイドで動的に生成されているため、WebSEAL ではフィルターを掛けることができません。したがって、このようなサーバー相対 URL は、アプリケーション・サーバーが存在する junction ポイントを無視したまま、表記されます。

このセクションでは、クライアント・サイドで動的に生成されたサーバー相対 URL の変更に対する Cookie ベースのソリューションについて説明します。クライアントが junction サーバーからページを受け取り、このページの動的に生成されたサーバー相対 URL を使用してリソースを要求すると、WebSEAL では特別な Cookie を使用して URL の再処理を試みることができます。この Cookie には、適切な junction 情報が含まれています。

このソリューションを使用するには、最初に **-j** オプションを使用して、バックエンド・アプリケーション・サーバーに対して junction を作成する必要があります。以下のステップの手順は、プロセス・フローを示しています。

1. クライアントは、バックエンドの junction アプリケーション・サーバーから HTML ページを要求します。

ページがクライアントのブラウザにロードされた場合、ページには他のコンテンツ以外に、サーバー相対 URL を生成する組み込みアプレットが含まれています。

2. ページは、**-j** オプションによって作成された junction を介してクライアントに戻されます。

-j オプションによって、WebSEAL は HTML ページの先頭に JavaScript ブロックを付加します。

JavaScript の目的は、ブラウザの junction 識別 Cookie を設定することです。

3. ページがクライアントのブラウザにロードされると、JavaScript が実行され、junction 識別 Cookie がブラウザの Cookie キャッシュに設定されます。

この Cookie は junction の名前を含むセッション Cookie です。

4. ページの組み込みアプレットは動的に実行され、サーバー相対 URL が生成されます。
5. クライアントはこのサーバー相対 URL を使用してリソースの要求を出します。junction Cookie 情報は、この要求で HTTP ヘッダーとして送信されます。

`IV_JCT = /junction-name`

6. クライアント要求のサーバー相対 URL にはフィルターが掛けられていないため、ローカル・リソースの要求として WebSEAL に表示されます。
7. ローカルでリソースの検出に失敗すると、WebSEAL は、Cookie により提供される junction 情報を使用して、その要求を即時に再試行します。
8. URL 表記の正しい junction 情報を使用すると、リソースはバックエンド・アプリケーション・サーバーで正常に検出されます。

次の図は、サーバー相対 URL をフィルターに掛けるための junction Cookie ソリューションを示しています。

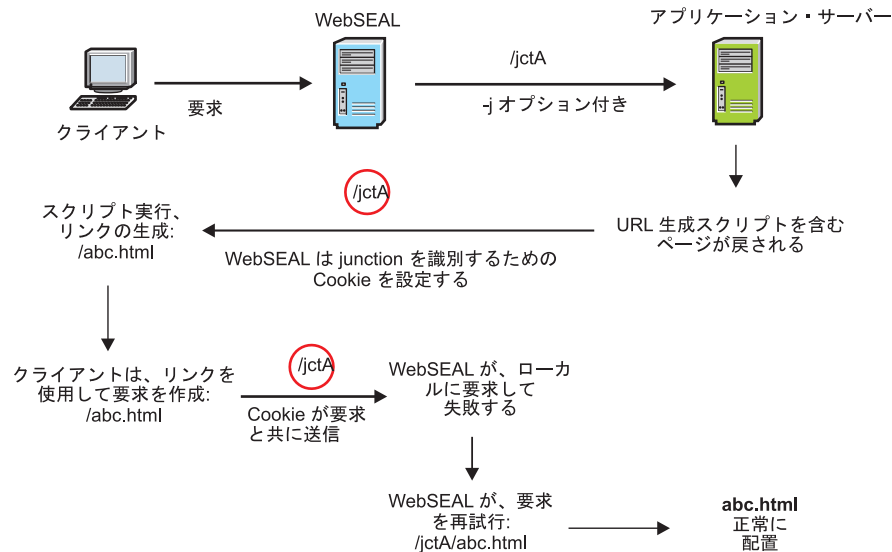


図 39. junction Cookie を使用したサーバー相対 URL の処理

junction Cookie をサポートする WebSEAL junction の構成

以下の一般的な構文を使用して、junction Cookie をサポートする junction を作成します。

```
pdadmin> server task instance-webseald-host create ... -j ...
```

以下の追加参照情報に関連情報が記載されています。

- 495 ページの『junction Cookie JavaScript ブロックのコントロール』。
- 499 ページの『要求の中のサーバー相対 URL 処理のコントロール』。
- 501 ページの『複数の -j junction にわたるサーバーからの Cookie の処理』。

WebSEAL は、動的に生成されたサーバー相対 URL を処理するための非 Cookie ベースの代替ソリューションを提供します。491 ページの『junction マッピングによるサーバー相対 URL の変更』を参照してください。

498 ページの『HTTP Referer ヘッダーを使用したサーバー相対 URL の変更』も参照してください。

junction Cookie JavaScript ブロックのコントロール

このセクションでは、**-j junction** オプションを使用して、サーバー相対 URL を変更する際に使用可能な以下の 2 つの構成オプションについて説明します (493 ページの『junction Cookie を使用したサーバー相対 URL の変更』を参照)。

- 『junction Cookie JavaScript ブロックの付加 (トレーラー)』
- 『HTML 4.01 準拠のための JavaScript ブロックの挿入 (inhead)』
- 496 ページの『複数の -j junction の junction Cookie のリセット (onfocus)』
- 497 ページの『XHTML 1.0 準拠 JavaScript ブロックの挿入 (xhtml10)』

junction Cookie JavaScript ブロックの付加 (トレーラー)

このタスクについて

-j junction オプションでは、文書を解釈するブラウザで junction 識別 Cookie を設定する JavaScript ブロックを挿入することによって、junction サーバーから戻される HTML 文書を変更します。デフォルトでは、JavaScript ブロックは、ページの先頭で、`<html>` タグの前に挿入されます。

このように JavaScript を付加する場所がページの先頭であるため、環境によっては HTML のレンダリングの問題が発生する可能性があります。この種類の問題が発生した場合、JavaScript ブロックを文書の最後に付加するように WebSEAL を構成することができます。

手順

バックエンド・サーバーによって戻されたページの最後に junction Cookie JavaScript ブロックを付加するように WebSEAL を構成するには、**-j junction** の作成時に **trailer** 引数を持つ **-J** オプションを追加します。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J trailer ...
```

HTML 4.01 に準拠するように正しい場所に JavaScript ブロックを挿入したい場合は、『HTML 4.01 準拠のための JavaScript ブロックの挿入 (inhead)』を参照してください。

HTML 4.01 準拠のための JavaScript ブロックの挿入 (inhead)

-J junction オプションでは、文書を解釈するブラウザで junction 識別 Cookie を設定する JavaScript ブロックを挿入することによって、junction サーバーから戻される HTML 文書を変更します。デフォルトでは、JavaScript ブロックは、ページの先頭で HTML コードの前に挿入されます。

このタスクについて

このように JavaScript を付加する場所がページの先頭であるため、環境によっては HTML のレンダリングの問題が発生する可能性があります。また、付加されるデフォルトの場所は、HTML 4.01 仕様に準拠していません。HTML 4.01 仕様では、`<script>` タグは `<head>` タグと `</head>` タグの間に配置しなければなりません。

手順

junction Cookie JavaScript ブロックを <head> </head> タグの間に挿入するように WebSEAL を構成するには (HTML 4.01 準拠)、**-J junction** の作成時に **inhead** 引数とともに **-J** オプションを追加します。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J inhead ...
```

xhtml10 引数も、その他の HTML 4.01 および XHTML 1.0 仕様に準拠するために使用します。497 ページの『XHTML 1.0 準拠 JavaScript ブロックの挿入 (xhtml10)』を参照してください。

trailer 引数は、HTML 4.01 仕様に準拠する必要がない場合に使用できます。495 ページの『junction Cookie JavaScript ブロックの付加 (トレーラー)』を参照してください。

複数の -j junction の junction Cookie のリセット (onfocus)

このタスクについて

1 つのクライアントの複数のインスタンスが複数の **-j junction** に同時にアクセスする環境では、JavaScript によって作成された最新の IV_JCT Cookie は、現在アクセス中の junction とは別の junction を誤って参照する場合があります。このような状態では、WebSEAL が誤った junction 情報を受け取ることになるため、リンクを正しく解決できません。

例えば、ユーザーが 2 つのブラウザー・ウィンドウを開き、各ウィンドウで 2 つの junction **jctA** および **jctB** のいずれかが示されているシナリオについて検討します。両方の junction とも **-j junction** オプションを使用して作成されています。

手順

1. 最初のブラウザー・ウィンドウでは、ユーザーは **jctA** にあるアプリケーション・サーバーからのページを要求します。
jctA の IV_JCT Cookie がブラウザーで設定されます。
2. 次に、ユーザーは最初のウィンドウを開いたまま、もう 1 つのブラウザー・ウィンドウに切り替え、**jctB** にあるアプリケーション・サーバーからのページを要求します。
jctB の IV_JCT Cookie がブラウザーで設定されます (**jctA** が置換されます)。
3. 次に、ユーザーは最初のブラウザー・ウィンドウに戻り、**jctA** にあるリソースへのリンクをクリックすると、誤った IV_JCT Cookie が WebSEAL に送信されます。

タスクの結果

この問題を解決するには、JavaScript の onfocus イベント・ハンドラーを使用するように WebSEAL を構成します。ユーザーがブラウザー・フォーカスのあるウィンドウから別のウィンドウに切り替え、必ず onfocus ハンドラーによって IV_JCT Cookie がリセットされます。

JavaScript onfocus イベント・ハンドラーを使用するには、**-j junction** の作成時に **onfocus** 引数を持つ **-J** オプションを追加します。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J onfocus ...
```

onfocus 引数を使用して junction を作成する場合は、**trailer** 引数も使用するようにしてください。**trailer** 引数を使用することによって、WebSEAL によって挿入された JavaScript が HTML フレーム・セットのレンダリングによって干渉されません。2 つの引数の間にはコンマ (,) を使用し、スペースは入れないでください。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J trailer,onfocus ...
```

495 ページの『junction Cookie JavaScript ブロックの付加 (トレーラー)』も参照してください。

HTML 4.01 および XHTML 1.0 仕様に準拠する必要がある場合は、『XHTML 1.0 準拠 JavaScript ブロックの挿入 (xhtml10)』を参照してください。

注: **-J** オプションに対して指定された引数が無効の場合、エラー・メッセージは表示されません。**-J junction** オプションが予想通り実行されない場合、引数が正しいかどうかを確認してください。

XHTML 1.0 準拠 JavaScript ブロックの挿入 (xhtml10)

-j junction オプションでは、文書を解釈するブラウザで junction 識別 Cookie を設定する JavaScript ブロックを挿入することによって、junction サーバーから戻される HTML 文書を変更します。デフォルト・スクリプトは、XHTML 1.0 仕様に準拠していません。

手順

XHTML 1.0 仕様 (および HTML 4.01 仕様) に準拠した junction Cookie JavaScript ブロックを挿入するように WebSEAL を構成するには、**-j junction** の作成時に **xhtml10** 引数を持つ **-J** オプションを追加します。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J xhtml10 ...
```

注:

- このオプションを使用して挿入された JavaScript は、挿入先の文書の Content-Type が text/html でない場合、実行されないことがあります。ただし、ほとんどのサイトが XHTML を送信する際に Content-Type:text/html を使用しているため、このことは通常問題になりません。
- **-J** オプションの **onfocus** 引数および **xhtml10** 引数は、相互に排他的です。**onfocus** も指定されている場合、WebSEAL は **xhtml10** を自動的に無視します。

xhtml10 引数を使用して junction を作成する場合は、**inhead** 引数も使用するようにしてください。**inhead** 引数を使用すると、HTML コード内部の JavaScript の配置が HTML 4.01 仕様に準拠するようになります。例えば次のように指定します (コマンド行フラグメント)。

```
pdadmin> server task instance-webseald-host create ... -j -J inhead,xhtml10 ...
```


495 ページの『HTML 4.01 準拠のための JavaScript ブロックの挿入 (inhead)』 も参照してください。

HTTP Referer ヘッダーを使用したサーバー相対 URL の変更

バックエンドの junction アプリケーション・サーバーからの HTML ページには、クライアント・サイドのサーバー相対リンクを動的に生成する組み込みアプレットまたはスクリプトが含まれる場合があります。これらの URL はクライアント・サイドで動的に生成されているため、WebSEAL ではフィルターを掛けることができません。したがって、このようなサーバー相対 URL は、アプリケーション・サーバーが存在する junction ポイントを無視したまま、表記されます。

このセクションでは、クライアント・サイドで動的に生成されたサーバー相対 URL の変更に対するソリューションについて説明します。このソリューションでは、HTTP 要求で標準的な Referer ヘッダーを使用します。WebSEAL は、要求内に junction Cookie が見つからない場合、または junction マッピング・テーブル・エントリーが要求に一致しない場合のみ、このソリューションを使用します。

HTTP 要求の Referer ヘッダーの情報をを使用して、組み込みアプレットまたはスクリプトを担当するアプリケーション・サーバーの junction ポイントを識別することができます。このソリューションは、動的に生成されたリンクが同一のアプリケーション・サーバーにあるリソースを指す (そのため、そのアプリケーション・サーバーが使用する同一の junction が必要となる) ことを前提とします。

バックエンド・アプリケーション・サーバーから戻された (組み込みアプレットまたはスクリプトが生成したリンクを含む) ページが、junction 名の知識を提供します。junction 名は、このページにある、クライアント・サイドで生成されたリンクを、ユーザーがクリックしたときに生成される要求の、Referer ヘッダーの URL 値に含まれます。例:

```
GET /back_end_app/images/logo.jpg
Referer: http://webseal/jctA/back_end_app
...
```

WebSEAL は、上記の要求 URL (/back_end_app/images/logo.jpg) を使用してリソースを検出できません。WebSEAL は、その要求の Referer ヘッダー内の情報を使用して、junction 名 **jctA** を追加で含むように要求 URL を変更することができます。例:

```
GET /jctA/back_end_app/images/logo.jpg
```

変更された URL を使用して、WebSEAL で正常にリソースを見つけることができます。これは、当然ながらリソース (logo.jpg) が同一のサーバー上にあることを前提としています。

環境によって、複数の junction にまたがるリソースを指すリンクがクライアント・サイドで生成された場合、URL 変更のための Referer ヘッダー・メソッドは信頼性が低くなります。このような環境では、junction マッピング・テーブル・ソリューションまたは junction Cookie ソリューションのいずれかを使用する必要があります。

も参照してください。

- 491 ページの『junction マッピングによるサーバー相対 URL の変更』

- 493 ページの『junction Cookie を使用したサーバー相対 URL の変更』

要求の中のサーバー相対 URL 処理のコントロール

このタスクについて

このセクションは、以下のトピックで構成されています。

- 『プロセス・ルートの要求の概念』
- 500 ページの『ルート要求処理の構成』

プロセス・ルートの要求の概念

このセクションでは、要求の中のサーバー相対 URL を処理する際に WebSEAL が実行するプロセスについて説明します。

process-root-requests スタンザ・エントリーによって、サーバー相対 URL を含む要求を WebSEAL が処理する順番を調整できます。例えば、最初に WebSEAL ルート junction で要求されたリソースを検索するように WebSEAL に指示することができます。リソースが「未検出」の場合、WebSEAL は 構成済みのポストプロセッシング・メカニズム (junction Cookie、junction マッピング・テーブル (JMT)、または Referer ヘッダー) などを使用して、要求の処理を継続します。

または、構成済みのポストプロセッシング・メカニズム (junction Cookie、junction マッピング・テーブル (JMT)、または Referer ヘッダー) などを使用して、WebSEAL にサーバー相対 URL 要求を最初に処理させることもできます。この方法の使用時にリソースが「未検出」の場合、WebSEAL はリソースのルート junction を検索します。

3 番目の構成では、ユーザーは特定のパス・パターンにフィルターを掛けることができます。

サーバー相対 URL を含む要求では、WebSEAL は、**process-root-requests** スタンザ・エントリーの構成済み設定に従い、リソースを検索します。「未検出」エラーが戻された場合、WebSEAL は、**process-root-requests** 構成に従い、要求の処理を継続します。

「サーバー・エラー」(500) などの他のエラーが戻された場合、WebSEAL はエラーをクライアントに戻し、リソースが検出されたと推測し、この時点で要求の処理を停止します。WebSEAL は、別の junction で要求の処理は試行しません。WebSEAL はこの状態が意図されたものとして機能します。

process-root-requests 構成の目的は、意図したリソースが確認される前に誤りのリソースにサーバー相対 URL の処理が行われることを防止することです。このアクションはパフォーマンス上の利点があり、さらに、偽の許可あるいはファイル・タイプの検査の失敗を避けられます。

ルート要求処理の構成

このタスクについて

ルート junction の処理を構成するには、WebSEAL 構成スタンザの **[server]** スタンザで **process-root-requests** エントリーを使用します。

Junction マッピング・メカニズムについては、以下のセクションを参照してください。

- 493 ページの『junction Cookie を使用したサーバー相対 URL の変更』
- 491 ページの『junction マッピングによるサーバー相対 URL の変更』

手順

WebSEAL 構成スタンザの **[server]** スタンザの **process-root-requests** エントリーを、ここに示す値のいずれかに設定します。デフォルト値は「always」です。

```
[server]
process-root-requests = always
```

有効な値は、以下のとおりです。

- **never**

WebSEAL は、構成済みのポストプロセッシング・メカニズムを使用して、最初にサーバー相対 URL 要求を処理します。メカニズムは、1) junction マッピング・テーブル (JMT)、2) junction Cookie、3) HTTP Referer ヘッダーの順序で試行されます。この方法の使用時にリソースが検出されない場合、WebSEAL はリソースのルート junction を検索します。

- **always**

WebSEAL は、構成済みのポストプロセッシング・メカニズム (junction Cookie、junction マッピング・テーブル (JMT)、または Referer ヘッダーなど) の使用を試行する前に、最初にルート junction でサーバー相対 URL 要求の処理を試行します。ルート junction がサービスを提供しているリソースのセットが大きくない場合、あるいは、この WebSEAL サーバーによってサービスが提供されている junction のセット用の ポストプロセッシング junction マッピング・メカニズムが構成されていない場合は、この設定は使用しないようにしてください。

- **filter**

すべてのルート junction 要求が調べられて、要求が、**[process-root-filter]** スタンザで指定されているパスのパターンで始まっているかが判別されます。指定されたパターンで始まっている場合は、それらの要求は、まずルート junction で処理されます。要求が、**[process-root-filter]** スタンザで指定されたパスのパターンで始まっていない場合は、即時に再マップされます。

process-root-requests = filter である場合は、ルート junction で処理されるルート junction 要求のパターンを指定する必要があります。**[process-root-filter]** スタンザを使用してください。パターンを指定する構文は、次のとおりです。

```
root = path_pattern
```

パス・パターンは、標準の WebSEAL ワイルドカード・パターンで表される必要があります。例:

```
[process-root-filter]
root = /index.html
root = /cgi-bin*
```

複数の -j junction にわたるサーバーからの Cookie の処理

このタスクについて

このセクションでは、バックエンド・アプリケーションによって生成され、複数の -j junction にわたるクライアントに戻される Cookie の、WebSEAL による処理方法について説明します。

- 『Cookie の処理: -j による Set-Cookie パス属性の変更』
- 502 ページの『Cookie の処理: -j による Set-Cookie 名前属性の変更』
- 503 ページの『Cookie 名の保存』
- 504 ページの『Cookie の処理: -I による固有の Set-Cookie 名前属性の確認』

Cookie の処理: -j による Set-Cookie パス属性の変更

junction ID Cookie をブラウザに提供するほかに、-j オプションを指定して構成されたか、または junction マッピング・テーブルにリストされた junction も、バックエンド・アプリケーションからの応答と一緒に送信される非ドメイン Cookie の処理をサポートしています。

ブラウザによる Cookie の処理:

サーバーからの応答の中の **Set-Cookie** ヘッダーに、**path** 属性 (例えば path=/xyz) が含まれている場合は、ブラウザは、要求 URL (戻されたページからアクティブ化される) がこのパスで始まっている (例えば /xyz/memo.html) 場合にのみ、Cookie を戻します。

問題:

Junction 環境に、応答内の可視 URL および組み込み URL を処理するためのソリューションが混在している場合は、ブラウザが Cookie を戻す能力に障害が出ることがあります。例えば、可視のサーバー相対 URL に対する WebSEAL の標準フィルター操作では、通常、URL 自体を変更するほかに、サーバー Cookie の **path** 属性の値に junction 名が追加されます (例えば、path=/jct/xyz)。URL パス名と Cookie の **path** の値の間にこのような一致が認められるため、ユーザーがリンクをアクティブ化したときに、ブラウザは Cookie を戻すことができます。

しかし、-j junction Cookie ベースのソリューションでは、URL 名に junction 名が追加されるのは、ユーザーがリンク (URL) をアクティブにした後 です。リンクをアクティブにした場合、事前変更された URL パス名 (/xyz/memo.html) と **Set-Cookie path** 属性値 (path=/jct/xyz) が一致しなくなります。要求によってサーバー Cookie は戻されません。

ソリューション:

-j オプションは、すべてのサーバー Cookie の **path** 属性の値を「/」に変換します (例えば、**path=/**)。すべてのサーバー相対パス名が「/」で始まっているため、元の **path** 属性が指定する要件に関係なく、すべてのサーバー Cookie が戻されます。

Cookie の処理: -j による Set-Cookie 名前属性の変更

-j オプションを指定して構成されたか、または junction マッピング・テーブルにリストされている junction でも、複数の junction にわたるサーバーから戻される Cookie を保存するためのソリューションが提供されます。

ブラウザによる Cookie の処理:

path 属性または **domain** 属性の一方または両方が固有のものでない場合、ブラウザは常に、保管されている Cookie を同じ **Set-Cookie name** 属性を含む、新たに到着した Cookie で置き換えます。

問題:

前のセクションでは、**-j junction** オプションが、**Set-Cookie** ヘッダーの **path** 属性の値をどのように変更するかについて説明しました。この変更によって、WebSEAL が応答ページに含まれている可視 URL と組み込み URL にそれぞれ異なるフィルター・ルールを適用する環境において、ブラウザが Cookie を戻せるようになります。

複数のバックエンド・サーバーが異なる junction を介して WebSEAL に接続されているシナリオ (例えば WebSphere 環境) においては、各サーバーが、同じ **name** 属性を持つ Cookie を送信する可能性があります。

junction が **-j** オプションを使用している場合は、各 Cookie の **path** 属性の値が同じ (**path=/**) になります。また、同じ WebSEAL サーバーがブラウザのコンタクト・ポイントなので、**domain** 属性も同じになります。これらの同じ Cookie はそれぞれ異なるバックエンド・アプリケーションから届くものですが、ブラウザはこのような同じ名前の Cookie を上書きしてしまいます。

ソリューション:

-j junction オプションには、バックエンド・アプリケーション・サーバーからの応答と共に戻される Cookie を固有の名前に変更するための追加機能があります。それは、**Set-Cookie** ヘッダーの **name** 属性の前に特殊ストリングを付加する機能です。このストリングには、ID (AMWEBJCT) および応答 (および Cookie) の配送を担当する特定 junction の名前が含まれています。感嘆符 (!) はストリング内の分離文字として使用されます。

AMWEBJCT!*jct-name*!

例えば、名前が ORDERID である Cookie が **/jctA** という名前の junction を介して到着した場合、この Cookie の名前は次のように変更されます。

AMWEBJCT!**jctA!ORDERID**

このデフォルトの Cookie 名前変更機能を使用不可に設定するには、503 ページの『Cookie 名の保存』を参照してください。

Cookie 名の保存

デフォルトで、WebSEAL は、**-j** オプションで作成されたか、junction マッピング・テーブルにリストされている、複数の junction にわたるバックエンド・アプリケーションからの応答の中で戻される Cookie の名前を変更します。WebSEAL は固有の Cookie の名前を作成して、その他の **-j** junction から戻される Cookie とのネーミング競合を回避するようにします。

WebSEAL では、**Set-Cookie** ヘッダーの **name** 属性の前に特殊ストリングを付加します。このストリングには、ID (AMWEBJCT) および応答 (および Cookie) の配送を担当する特定 junction の名前が含まれています。

```
AMWEBJCT!jct-name!
```

例えば、名前が ORDERID である Cookie が **/jctA** という名前の junction を介して到着した場合、この Cookie の名前は次のように変更されます。

```
AMWEBJCT!jctA!ORDERID
```

ただし、フロントエンド・ブラウザーおよびアプリケーションが、アプリケーションによって生成された特定の Cookie 名に依存している場合は、以下の Cookie 名前変更機能性を使用不可にできます。

- **-n** オプションによって構成された junction にわたるすべての Cookie。

『すべての Cookie の名前の保存』を参照してください。

- すべての junction にわたる WebSEAL 構成ファイルの **[preserve-cookie-names]** スタンザで構成される特定の Cookie。

『特定の Cookie の名前の保存』を参照してください。

すべての Cookie の名前の保存

このタスクについて

特定の **-j** junction にわたる非ドメイン Cookie の名前変更は、**-n** オプションでその junction を追加構成することによって、防止できます。**-n** オプションによって、非ドメイン Cookie の名前が変更されないように指定されます。例えば、クライアント・サイドのスクリプトが特定の名前の Cookie に依存している場合に、このオプションを使用します。

特定の Cookie の名前の保存

複数の junction にわたるバックエンド・アプリケーション・サーバーから戻される際に変更されない特定の Cookie の名前をリストすることができます。

このタスクについて

WebSEAL 構成ファイルの **[preserve-cookie-names]** スタンザ内の **name** スタンザ・エントリーを次のように使用すると、WebSEAL によって名前変更されない特定の Cookie の名前をリストできます。特定の Cookie の名前は既存の junction 全体で保護されます。

例

```
[preserve-cookie-names]
name = cookie-name1
name = cookie-name2
```

Cookie の処理: -I による固有の Set-Cookie 名前属性の確認

-j オプションを指定して **junction** が構成されると、バックエンド・サーバーからの応答の中の **Set-Cookie** ヘッダーによって **path** 属性値が「/」に変換され、**junction** ポイントを含めることによって、その **name** 属性が変更されます。**junction** ポイントによる **name** 属性を変更した結果、Cookie が相互に排他的とならない場合があります。

標準 **-j** オプションの操作:

ヘッダー:

```
Set-Cookie: ORDERID=123456; path=/orders
```

が **-j junction /sales** のバックエンド・サーバーから受信されると、以下の変更済みヘッダーがブラウザに送信されます。

```
Set-Cookie: AMWEBJCT!/sales!ORDERID=123456; path=/
```

ただし、**name** 属性が同じで **path** 値が異なる別の **Set-Cookie** ヘッダーが同一の **junction** を介して受信されると、変更済みヘッダーの名前およびパス情報はまったく同じになります。

例:

```
Set-Cookie: ORDERID=123456; path=/invoices
```

は以下のように変更されます。

```
Set-Cookie: AMWEBJCT!/sales!ORDERID=123456; path=/
```

2 番目の変更済み **Set-Cookie** ヘッダーの Cookie **name** および **path** は最初のヘッダーと同じため、最初のヘッダーが上書きされます。**junction** ポイントでは、**Set-Cookie** ヘッダーを一意的に識別できません。

ソリューション:

元の **path** 属性値 (**/orders** など) を Cookie の変更済み **name** に追加するために、追加の **-I** オプションを指定して、**-j junction** を構成することができます。このとき、Cookie 名は固有の名前となります。**-I** オプションを使用する際は、以下のルールが適用されます。

- **junction** サーバーからの **Set-Cookie** ヘッダーに **path** 属性が含まれる場合、その **path** の値は URI エンコード化され、**name** 属性を変更するために使用されます。
- **junction** サーバーからの **Set-Cookie** ヘッダーに **path** 属性が含まれない場合、要求 URI の **basedir** が抽出され、URI エンコード化され、**name** 属性を変更するために使用されます。

例えば、クライアントが `/dir1/dir2/mypage.html` に対する要求を行った場合、値 `/dir1/dir2` は URI エンコード化され、使用されます。

- その後、**Set-Cookie name** 属性は、junction ポイント (ルート junction「/」ではない場合) および URI エンコード **path** 値 (または **basedir** 値) を使用して変更されます。
- **Set-Cookie path** 属性の値は依然として「/」に変換されています。

例えば、ヘッダー:

```
Set-Cookie: ORDERID=123456; path=/orders
```

が **-j -I junction /sales** のバックエンド・サーバーから受信されると、以下の変更済みヘッダーがブラウザに送信されます。

```
Set-Cookie: AMWEBJCT!/sales/orders!ORDERID=123456; path=/
```

第 26 章 HTTP 変換

HTTP 変換ルールにより、HTTP 要求と HTTP 応答が WebSEAL を通過するときに、それらに対して変更を加えることができます。この機能では XSLT を使用します。保護オブジェクト・ポリシー (POP) を構成して、指定されたルールをトリガーすることができます。

WebSEAL アドミニストレータは、以下の変更を構成することができます。これらの変換は、(特に注記がある場合を除いて) HTTP 要求と HTTP 応答に適用できます。

- ヘッダーを追加する
- ヘッダーを削除する
- 既存のヘッダーを変更する
- URI を変更する (要求のみ)
- 方式を変更する (要求のみ)
- HTTP バージョンを変更する
- HTTP 状況コードを変更する (応答のみ)
- 状況の理由を変更する (応答のみ)
- Cookie を追加する
- Cookie を削除する
- 既存の Cookie を変更する
- 本体を追加する (応答のみ)

注:

1. 要求または応答の本体を変更することはできません。同様に、WebSEAL によって挿入された Cookie やヘッダーを変更することはできません。例えば、**Host**、**iv-user**、および **iv-creds junction** ヘッダーなどです。
2. lib/html ディレクトリの下にある WebSEAL ページは、**HTML サーバー応答ページ**と呼ばれます。これらの応答ページは、次のようにグループ化されます。
 - アカウント管理ページ。
 - エラー・メッセージ・ページ。

これらの応答ページのロケーションは、**[acct-mgt]** スタンザと **[content]** スタンザで構成できます。

HTTP 変換ルール

WebSEAL が受信した HTTP 要求と HTTP 応答は、XML オブジェクトとして表現され、XSL 変換を使用して操作することができます。

XSLT ルールを使用することにより、HTTP 要求と HTTP 応答が WebSEAL を通過するときに、それらに適用する変更内容を表現することができます。 WebSEAL は、HTTP 変換で以下の 2 つの入力を使用します。

- HTTP 要求または HTTP 応答の XML 表現。
- 要求または応答の変更方法を決める XSLT。

変換からの出力は、HTTP 要求または HTTP 応答に対する必要な変更の概要を示した XML 文書です。

注:

- XSLT ルールは、ルール・ファイルに書き込まれます。ルール・ファイルを変更した場合は、変更を有効にするために WebSEAL サーバーを再始動する必要があります。
- ヘッダー・フィールドは、XML の問題を回避するために URL エンコードを行う必要があります。WebSEAL は、変換プロセス中に URL エンコードされたヘッダー値を使用します。

Extensible Stylesheet Language Transformation (XSLT)

Extensible Stylesheet Language (XSL) は、ルールを指定するために使用する言語で、Extensible Markup Language (XML) は、ルールへの入力を形成するデータに使用する言語です。XML と XSL を組み合わせることにより、ルール評価プログラムへの入力およびルール自体の両方を表現する、プラットフォームに依存しない方法が提供されます。

XML を使用して、構造化された標準的な方法で複合データ型をテキスト形式で表現することができます。このテキスト形式を使用することで、XML データを処理するためのルールを、プラットフォームおよびプログラミング言語特性を考慮する必要なく書き込むことができます。

XSL Transformation (XSLT) ルールを使用して、XML 文書を XML、PDF、HTML、またはその他の形式の文書に変換することができます。変換ルールは、XSL スタイル・シート内に XSL テンプレートとして定義する必要があります。このルールは有効な XSL テンプレート・ルール形式で記述する必要があります。XSL は、XML データを分析して評価する継承機能を所有し、これは e-business モデルにおけるデータ表記の標準になりつつあります。

HTTP 要求オブジェクト

以下の例は、**HTTPRequest** の XML 表記を示したものです。

```
<?xml version="1.0" encoding="UTF-8"?>
<HTTPRequest>
  <RequestLine>
    <Method>GET</Method>
    <URI>/en/us/</URI>
    <Version>HTTP/1.1</Version>
  </RequestLine>
  <Headers>
    <Header name="User-Agent">curl%2F7.18.2%20(i486-pc-linux-gnu)%20libcurl%2F7.18.2%20openssl%2F0.9.8g%20zlib%2F1.2.3.3%20libidn%2F1.8</Header>
    <Header name="Host">www.ibm.com</Header>
```

```

    <Header name="Accept">*</Header>
  </Headers>
  <Cookies>
    <Cookie name="PD-S-SESSION-ID">2_orQUNJCbjdxqIEdDPMXj31UHMxuU3hRCU</Cookie>
  </Cookies>
</HTTPRequest>

```

HTTP 応答オブジェクト

以下の例は、**HTTPResponse** の XML 表記を示したものです。

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponse>
  <ResponseLine>
    <Version>HTTP/1.1</Version>
    <StatusCode>200</StatusCode>
    <Reason>OK</Reason>
  </ResponseLine>
  <Headers>
    <Header name="Date">Thu%2C%2016%20Sep%202010%2010%3A57%3A52%20GMT</Header>
    <Header name="Server">IBM_HTTP_Server</Header>
    <Header name="ContentLength">4096</Header>
    <Header name="Content-Type">text%2Fhtml%3Bcharset%3DUTF-8</Header>
    <Header name="Content-Language">en-US</Header>
  </Headers>
  <Cookies>
    <Cookie name="PD-S-SESSION-ID">
      <Content>2_orQUNJCbjdxqIEdDPMXj31UHMxuU3hRCUtpN7xe6J1xZhxt0</Content>
      <Path>/</Path>
      <Domain>domainA.ibm.com</Domain>
      <Expires>Wed, 09 Jun 2021 10:18:14 GMT</Expires>
      <Secure>1</Secure>
      <HTTPOnly>0</HTTPOnly>
    </Cookie>
  </Cookies>
</HTTPResponse>

```

HTTP 応答の置換

指定した HTTP 要求の場合に、まったく新しい HTTP 応答を作成することができます。この動作を実現するため、基本 XML エレメント内に **action="replace"** を含んだ **HTTPResponseChange** XML 文書を生成するように HTTP 要求を変更します。

例:

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponseChange action="replace">
  <Version>HTTP/1.1</Version>
  <StatusCode>503</StatusCode>
  <Reason>Not Implemented</Reason>
  <Body>%3Ch1%3EError%3C%2Fh1%3E%0A%3Cp%3EInvalid%20cookie%20%3C%2Fp%3E</Body>
</HTTPResponseChange>

```

シナリオ例については、522 ページの『シナリオ 5: 既知の HTTP 要求に対する応答の生成』を参照してください。

HTTPResponseChange 文書は、HTTP 要求または HTTP 応答の変更の結果として生成されます。

WebSEAL は、HTTP 要求変更の結果として **action="replace"** を含んだ **HTTPResponseChange** 文書を受信すると、次のように動作します。

- 通常の処理の流れを中断し、通常の方法で応答を生成しない。
- 入力された XML 文書に基づいて新規の HTTP 応答を作成する。

この機能を使用して、特定の HTTP 要求をインターセプトし、事前定義された応答を返すことができます。

注: WebSEAL は同様に、HTTP 応答変更の結果として **action="replace"** を含んだ **HTTPResponseChange** 文書を受信すると、次のように動作します。

- 元の応答を破棄する。
- XML で指定されている応答を使用する。

XSL Transformation ルール

HTTP 要求および応答の内容の変換に、有効な XSLT 文書を使用することができます。

XSL Transformation は、必須の変更を定義する XML 文書を出力する必要があります。出力文書には、HTTP 要求または HTTP 応答に対して行う必要がある変更を記述する、一連の XML エLEMENTが含まれます。

重要: XSLT 文書は慎重に作成してください。XSL Transformation ルールは、実稼働環境に実装する前に十分に確認およびテストしてください。不正な構文または不正な形式の XSLT によって、エラーまたは予期しない動作が発生する可能性があります。

以下の表に、WebSEAL が変換された文書内で要求する基本 XML ELEMENTを示します。

表 37. 基本ELEMENT

ソース・ドキュメント	基本 XML ELEMENT
HTTP 要求	<HTTPRequestChange>
HTTP 応答	<HTTPResponseChange>

XSL Transformation ルールは、HTTP 入力の内容を処理する必要があります。内容には以下が含まれます。

- **ResponseLine/RequestLine** ELEMENT。
- **Headers** ELEMENT。
- **Cookies** ELEMENT。
- **Body** ELEMENT。(HTTPResponseChange のみ)

変換された XML 文書に **RequestLine/ResponseLine** のELEMENTが含まれる場合、WebSEAL は対応する変更を HTTP 要求/応答に適用します。

WebSEAL によるヘッダーの変換方法を決定するには、XSLT 文書の **Header** ELEMENTに **action** 属性が必要です。使用できるアクションは以下のとおりです。

1. **add** - 特定の名前および値を持つ新規ヘッダーを追加します。

2. **update** - 既存のヘッダーの値を更新します (ヘッダーが存在しない場合は追加されます)。
3. **remove** - 特定の名前および値を持つヘッダーを除去します。

WebSEAL による Cookie の変換方法を決定するには、XSLT 文書の **Cookie** エlement に **action** 属性が必要です。使用できるアクションは以下のとおりです。

1. **add** - 特定の名前および値を持つ新規 Cookie を追加します。
2. **update** - 既存の Cookie の値を更新します。(Cookie が存在しない場合は追加されます。)
3. **remove** - 特定の名前を持つ Cookie を除去します。

注: Cookie は、要求内と応答内で表示が異なります。応答内にのみ、名前と値以外の属性が含まれます。Cookie を更新する場合は、Cookie 名と、更新対象にするフィールドを指定します。Cookie を追加する場合、指定する必要がある最低限のフィールドは、Cookie 名と値です。

オプションで、HTTP 応答に本体を挿入するために、**Body** エlement を含めることができます。**Body** の内容は、エンコードされた URL にする必要があります。WebSEAL は、応答の作成時に内容をデコードします。WebSEAL は、HTTP 応答内の既存の本体を、この **Body** エlement に指定されている新規内容に置き換えます。この Element にアクションは必要ありません。

注: 要求内の本体の内容を置き換えることはできません。

再処理に関する考慮事項

HTTP 変換ルールにより要求の URI またはホスト・ヘッダーが変更された場合、WebSEAL は変換された要求を再処理します。

この再処理により、変換が WebSEAL 許可をバイパスしないようになります。また、この動作は、アドミニストレーターが要求を別の Junction に送信するための HTTP 変換ルールを定義することも意味します。

WebSEAL が再処理 (および許可) を実行するのは、最初の HTTP 変換においてのみです。関連するオブジェクト・スペースに適切な POP が付加されている場合、変換された要求に対してもう一度 HTTP 変換が行われます。513 ページの『保護オブジェクト・ポリシー (POP)』を参照してください。ただし、WebSEAL は、これらの後続の変換の結果作成された新規要求を再処理しません。

XSLT テンプレート

以下の表に、ご使用の WebSEAL サーバー上にある XSLT テンプレート・ファイルを示します。これらのテンプレートは、ローカル管理インターフェース (LMI) を使用して入手できます。「**セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)**」>「**グローバル設定**」>「**HTTP 変換**」に移動します。「**新規**」をクリックして、使用可能なテンプレートのいずれかを使用する HTTP 変換ルール・ファイルを作成します。

表 38. XSLT テンプレート・ファイル

テンプレート	説明
Request	HTTP 要求用のサンプル XSLT テンプレート。
Response	HTTP 応答用のサンプル XSLT テンプレート。

注: 変換ルールを作成する場合は、ヘッダー名に関する HTTP 仕様との整合性を維持するために、ヘッダー名の大文字小文字を区別しない突き合わせを使用します。

サンプルの XSLT 文書を使用したシナリオは、513 ページの『HTTP 変換のシナリオ例』を参照してください。

構成

効率を高めるため、トランザクションでの変換プロセスは必要な場合にのみ実行することが重要です。**[http-transformations]** スタンザおよび関連付けられている POP を使用して、HTTP 変換プロセスを必要とするオブジェクトを構成できます。

オブジェクト・スペース内の適切なオブジェクトに POP を付加する必要があります。その POP には、リソースの名前が拡張属性として含まれている必要があります。このリソース名は、**[http-transformations]** 構成スタンザのエントリー名と突き合せられます。この突き合せ対象の **[http-transformations]** スタンザ・エントリーの値には、適用される XSLT ルールが書き込まれたファイルの名前が指定されています。

構成ファイルの更新

[http-transformations] スタンザを使用して HTTP 変換リソースを定義できます。

このスタンザの構成エントリーは次のフォーマットにする必要があります。

resource-name = resource-file

説明:

resource-name

HTTP 変換リソースの名前。

resource-file

リソース XSL ファイルの名前。

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の **[http-transformations]** スタンザを参照してください。

保護オブジェクト・ポリシー (POP)

オブジェクト・スペースの該当部分に事前定義 XSLT リソースを使用可能にするには、POP を使用します。このメカニズムによって、HTTP 変換を行う必要があるリソースを指定できます。

POP には、HTTPTransformation という名前の拡張属性と、以下の形式の値が必要です。

- Request=*resource*、または
- Response=*resource*

説明:

resource

WebSEAL 構成ファイルに定義されている HTTP 変換リソース名のいずれかを指定します。

各要求および応答に対して、1 つの HTTP 変換のみが実行されます。Request 値または Response 値に複数の **HTTPTransformation** 属性が存在する場合は、最初の属性が選択されます。

以下に POP の例を示します。

```
pdadmin sec_master> pop show http-transformation-pop attribute HTTPTransformation
HTTPTransformation
    Request=resource_a
    Response=resource_b
```

HTTP 変換のシナリオ例

以下のシナリオは、HTTP 変換プロセスでの入力文書と出力文書の例です。

シナリオ 1: URI、ヘッダー、および Cookie の変更 (HTTPRequest)

このシナリオでは、RequestLine/URI エlement、および元の HTTP 要求内のヘッダー・Element と Cookie Element の変更方法を示します。

この例では、HTTP 要求に対して以下の変更が加えられます。

1. /test を既存の URI 値に付加します。
2. 値が **VALUE_A** の **NAME_A** という名前のヘッダーがない場合は、その新規ヘッダーを追加します。
3. **NAME_B** ヘッダー値を更新して **UPDATED_B** にします。
4. **NAME_C** というヘッダーを除去します。
5. **MY_COOKIE** という Cookie を追加します。
6. **EXISTING_COOKIE** Cookie の内容を更新して **NEW_COOKIE_VALUE** にします。

入力文書

このシナリオでは、以下の入力文書サンプルを使用します。

HTTP 要求

```
<?xml version="1.0" encoding="UTF-8"?>
<HttpRequest>
  <RequestLine>
    <Method>GET</Method>
    <URI>/en/us/</URI>
    <Version>HTTP/1.1</Version>
  </RequestLine>
  <Headers>
    <Header name="Host">www.ibm.com</Header>
    <Header name="NAME_B">ORIGINAL_B</Header>
    <Header name="NAME_C">ORIGINAL_C</Header>
  </Headers>
  <Cookies>
    <Cookie name="EXISTING_COOKIE">2_orQUNJCbjdxqIEdDPMXj31UHMxuU3hRCU...</Cookie>
  </Cookies>
</HttpRequest>
```

XSLT ルール

注: これらのルールは、関連付けられた POP により要求リソースとして定義される XSL 文書内に保管する必要があります。512 ページの『構成』を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <!--Firstly, strip any space elements -->
  <xsl:strip-space elements="*" />

  <!--
    Perform a match on the root of the document. Output the required
    HTTPRequestChange elements and then process templates.
  -->
  <xsl:template match="/">
    <HttpRequestChange>
      <xsl:apply-templates />
    </HttpRequestChange>
  </xsl:template>

  <!--
    Do nothing to the Method.
  -->
  <xsl:template match="//HttpRequest/RequestLine/Method" />

  <!--
    Match on the URI. Append "test/" to the URI.
  -->
  <xsl:template match="//HttpRequest/RequestLine/URI">
    <URI>
      <xsl:value-of select="node()" />
      test/
    </URI>
  </xsl:template>

  <!--
    Do nothing to the Version
  -->
  <xsl:template match="//HttpRequest/RequestLine/Version" />
```

```

<!--
  Match on the Headers. Add a new header called NAME_A if
  it does not exist.
-->
<xsl:template match="//HTTPRequest/Headers">
  <xsl:choose>
    <xsl:when test="Header/@name='NAME_A'" />
    <xsl:otherwise>
      <Header action="add" name="NAME_A">
        VALUE_A
      </Header>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:apply-templates select="//HTTPRequest/Headers/Header" />
</xsl:template>

<!-- Process the header elements -->
<xsl:template match="//HTTPRequest/Headers/Header">
  <xsl:choose>
    <!-- Update the value of the NAME_B header -->
    <xsl:when test="@name = 'NAME_B'">
      <Header action="update" name="NAME_B">
        UPDATED_B
      </Header>
    </xsl:when>
    <!-- Delete the NAME_C header -->
    <xsl:when test="contains(@name, 'NAME_C')">
      <Header action="remove" name="NAME_C">
        <xsl:value-of select="node()" />
      </Header>
    </xsl:when>
  </xsl:choose>
</xsl:template>

<!--
  Match on the Cookies. Add a new cookie called MY_COOKIE if
  it does not exist.
-->
<xsl:template match="//HTTPRequest/Cookies">
  <xsl:choose>
    <xsl:when test="Cookie/@name='MY_COOKIE'" />
    <xsl:otherwise>
      <Cookie action="add" name="MY_COOKIE">
        MY_COOKIE_VALUE
      </Cookie>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:apply-templates select="//HTTPRequest/Cookies/Cookie" />
</xsl:template>

<!-- Process the cookie elements -->
<xsl:template match="//HTTPRequest/Cookies/Cookie">
  <xsl:choose>
    <!-- Update the value of the EXISTING_COOKIE cookie -->
    <xsl:when test="@name = 'EXISTING_COOKIE'">
      <Cookie action="update" name="EXISTING_COOKIE">
        NEW_COOKIE_VALUE
      </Cookie>
    </xsl:when>
  </xsl:choose>
</xsl:template>
</xsl:stylesheet>

```

出力 XML 文書

このシナリオでは、XSL 変換によって次の XML 文書が出力されます。この文書には、元の HTTP 要求に対して WebSEAL が実行する変更の概要が示されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<HTTPRequestChange>
  <URI>/en/us/test/</URI>
  <Header action="add" name="NAME_A">VALUE_A</Header>
  <Header action="update" name="NAME_B">UPDATED_B</Header>
  <Header action="remove" name="NAME_C">ORIGINAL_C</Header>
  <Cookie action="add" name="MY_COOKIE">MY_COOKIE_VALUE</Cookie>
  <Cookie action="update" name="EXISTING_COOKIE">NEW_COOKIE_VALUE</Cookie>
</HTTPRequestChange>
```

シナリオ 2: ヘッダーのみの変更 (HTTPResponse)

このシナリオでは、HTTP 応答内のヘッダーの変更方法を示します。この例の XSLT は、値が **VALUE_A** の **RESPONSE_A** という名前のヘッダーがない場合、その新規ヘッダーを追加します。

入力文書

このシナリオでは、以下の入力文書サンプルを使用します。

HTTP 応答

```
<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponse>
  <ResponseLine>
    <Version>HTTP/1.1</Version>
    <StatusCode>200</StatusCode>
    <Reason>OK</Reason>
  </ResponseLine>
  <Headers>
    <Header name="Date">Thu%2C%2016%20Sep%202010%2010%
      %3A57%3A52%20GMT</Header>
    <Header name="Server">IBM_HTTP_Server</Header>
    <Header name="Content-Type">text%2Fhtml%3Bcharset%3DUTF-8</Header>
    <Header name="Content-Language">en-US</Header>
  </Headers>
  <Cookies>
    <Cookie name="PD-S-SESSION-ID">
      <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
      <Path>/</Path>
      <Domain>domainA.com</Domain>
      <Expires>Wed, 09 Jun 2021 10:18:14 GMT</Expires>
      <Secure>1</Secure>
      <HTTPOnly>0</HTTPOnly>
    </Cookie>
  </Cookies>
</HTTPResponse>
```

XSLT ルール

注: これらのルールは、関連付けられた POP により応答リソースとして定義される XSL 文書内に保管する必要があります。512 ページの『構成』を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">
```

```

<!--Firstly, strip any space elements -->
<xsl:strip-space elements="*" />

<!--
Perform a match on the root of the document. Output the required
HTTPResponseChange elements and then process templates.
-->
<xsl:template match="/">
  <HTTPResponseChange>
    <xsl:apply-templates />
  </HTTPResponseChange>
</xsl:template>

<!--
Do nothing to the Version
-->
<xsl:template match="//HTTPResponse/ResponseLine/Version" />

<!--
Do nothing to the StatusCode
-->
<xsl:template match="//HTTPResponse/ResponseLine/StatusCode" />

<!--
Do nothing to the Reason
-->
<xsl:template match="//HTTPResponse/ResponseLine/Reason" />

<!--
Match on the Headers. Add a new header called RESPONSE_A
if it does not exist.
-->
<xsl:template match="//HTTPResponse/Headers">
  <xsl:choose>
    <xsl:when test="Header/@name='RESPONSE_A'"/>
    <xsl:otherwise>
      <Header action="add" name="RESPONSE_A">
        VALUE_A
      </Header>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<!--
Do nothing to the Cookies
-->
<xsl:template match="//HTTPResponse/Cookies" />

</xsl:stylesheet>

```

出力 XML 文書

このシナリオでは、XSL 変換によって次の XML 文書が出力されます。この文書には、元の HTTP 応答に対して WebSEAL が実行する変更の概要が示されています。

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponseChange>
  <Header action="add" name="RESPONSE_A">VALUE_A</Header>
</HTTPResponseChange>

```

シナリオ 3: ResponseLine/StatusCode のみの変更 (HTTPResponse)

このシナリオでは、HTTP 応答内の StatusCode エlementと Reason エlementの変更方法を示します。この例の XSLT は、以下の更新を行います。

- 503 状況コードを 501 に変更します。
- 状況コードが更新されると、その理由も変更を反映するように更新されます。501 状況の理由は「Not Implemented」です。

入力文書

このシナリオでは、以下の入力文書サンプルを使用します。

HTTP 応答

```
<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponse>
  <ResponseLine>
    <Version>HTTP/1.1</Version>
    <StatusCode>503</StatusCode>
    <Reason>Service Unavailable</Reason>
  </ResponseLine>
  <Headers>
    <Header name="Date">Thu%2C%2016%20Sep%202010%2010%3A57%3A52%20GMT</Header>
    <Header name="Server">IBM_HTTP_Server</Header>
    <Header name="Content-Type">text%2Fhtml%3Bcharset%3DUTF-8</Header>
    <Header name="Content-Language">en-US</Header>
  </Headers>
  <Cookies>
    <Cookie name="PD-S-SESSION-ID">
      <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
      <Path>/</Path>
      <Domain>domainA.com</Domain>
      <Expires>Wed, 09 Jun 2021 10:18:14 GMT</Expires>
      <Secure>1</Secure>
      <HTTPOnly>0</HTTPOnly>
    </Cookie>
  </Cookies>
</HTTPResponse>
```

XSLT ルール

注: これらのルールは、関連付けられた POP により応答リソースとして定義される XSL 文書内に保管する必要があります。512 ページの『構成』を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <!--Firstly, strip any space elements -->
  <xsl:strip-space elements="*" />

  <!--
  Perform a match on the root of the document. Output the required
  HTTPResponseChange elements and then process templates.
  -->
  <xsl:template match="/">
    <HTTPResponseChange>
      <xsl:apply-templates />
    </HTTPResponseChange>
  </template>
```



```

        </HTTPResponseChange>
    </xsl:template>

    <!--
    Do nothing to the Version
    -->
    <xsl:template match="//HTTPResponse/ResponseLine/Version" />

    <!--
    If the original StatusCode is 503 then update the
    StatusCode to 501.
    -->
    <xsl:template match="//HTTPResponse/ResponseLine/StatusCode">
        <xsl:choose>
            <xsl:when test="503">
                <StatusCode>501</StatusCode>
            </xsl:when>
        </xsl:choose>
    </xsl:template>

    <!--
    Update the Reason to match the StatusCode change.
    -->
    <xsl:template match="//HTTPResponse/ResponseLine/Reason">
        <xsl:choose>
            <xsl:when test="'Service Unavailable'">
                <Reason>Not Implemented</Reason>
            </xsl:when>
        </xsl:choose>
    </xsl:template>

    <!--
    Do nothing to the Headers.
    -->
    <xsl:template match="//HTTPResponse/Headers" />

    <!--
    Do nothing to the Cookies.
    -->
    <xsl:template match="//HTTPResponse/Cookies" />

</xsl:stylesheet>

```

出力 XML 文書

このシナリオでは、XSL 変換によって次の XML 文書が出力されます。この文書には、元の HTTP 応答に対して WebSEAL が実行する変更の概要が示されています。

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponseChange>
  <StatusCode>501</StatusCode>
  <Reason>Not Implemented</Reason>
</HTTPResponseChange>

```

シナリオ 4: Cookie のみの変更 (HTTPResponse)

このシナリオでは、HTTP 応答内の Cookie の追加、変更、および削除方法を示します。この例の XSLT は、以下の更新を行います。

- **NEW_COOKIE** という Cookie を追加する。
- **EXISTING_COOKIE** Cookie ドメインを更新して **domainB.com** にする。
- **OLD_COOKIE** という Cookie を削除する。

入力文書

このシナリオでは、以下の入力文書サンプルを使用します。

HTTP 応答

```
<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponse>
  <ResponseLine>
    <Version>HTTP/1.1</Version>
    <StatusCode>503</StatusCode>
    <Reason>Service Unavailable</Reason>
  </ResponseLine>
  <Headers>
    <Header name="Date">Thu%2C%2016%20Sep%202010%2010%3A57%3A52%20GMT</Header>
    <Header name="Server">IBM_HTTP_Server</Header>
    <Header name="Content-Type">text%2Fhtml%3Bcharset%3DUTF-8</Header>
    <Header name="Content-Language">en-US</Header>
  </Headers>
  <Cookies>
    <Cookie name="EXISTING_COOKIE">
      <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
      <Path>/</Path>
      <Domain>domainA.com</Domain>
      <Expires>Wed, 09 Jun 2021 10:18:14 GMT</Expires>
      <Secure>1</Secure>
      <HTTPOnly>0</HTTPOnly>
    </Cookie>
    <Cookie name="OLD_COOKIE">
      <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
      <Path>/</Path>
      <Domain>domainA.com</Domain>
      <Expires>Mon, 07 Jun 2021 11:18:21 GMT</Expires>
      <Secure>1</Secure>
      <HTTPOnly>0</HTTPOnly>
    </Cookie>
  </Cookies>
</HTTPResponse>
```

XSLT ルール

注: これらのルールは、関連付けられた POP により応答リソースとして定義される XSL 文書内に保管する必要があります。512 ページの『構成』を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <!--Firstly, strip any space elements -->
  <xsl:strip-space elements="*" />

  <!--
  Perform a match on the root of the document. Output the required
  HTTPResponseChange elements and then process templates.
  -->
  <xsl:template match="/">
    <HTTPResponseChange>
      <xsl:apply-templates />
    </HTTPResponseChange>
  </xsl:template>

  <!--
  Do nothing to the Version
```

```

-->
<xsl:template match="//HTTPResponse/ResponseLine/Version" />

<!--
  Do nothing to the StatusCode
-->
<xsl:template match="//HTTPResponse/ResponseLine/StatusCode" />

<!--
  Do nothing to the Reason
-->
<xsl:template match="//HTTPResponse/ResponseLine/Reason" />

<!--
  Do nothing to the Headers.
-->
<xsl:template match="//HTTPResponse/Headers" />

<!--
  Match on the Cookies. Add a new cookie called NEW_COOKIE if
  it does not exist.
-->
<xsl:template match="//HTTPResponse/Cookies">

  <xsl:choose>
    <xsl:when test="Cookie/@name='NEW_COOKIE'" />
    <xsl:otherwise>
      <Cookie action="add" name="NEW_COOKIE">
        <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
        <Path>/</Path>
        <Domain>domainA.com</Domain>
        <Expires>Mon, 07 Jun 2021 10:12:14 GMT</Expires>
        <Secure>1</Secure>
        <HTTPOnly>0</HTTPOnly>
      </Cookie>
    </xsl:otherwise>
  </xsl:choose>

  <!-- Update the value of the EXISTING_COOKIE cookie -->
  <xsl:if test="Cookie/@name='EXISTING_COOKIE'">
    <Cookie action="update" name="EXISTING_COOKIE">
      <Domain>domainB.com</Domain>
    </Cookie>
  </xsl:if>

  <!-- Delete the OLD_COOKIE cookie -->
  <xsl:if test="Cookie/@name='OLD_COOKIE'">
    <Cookie action="remove" name="OLD_COOKIE" />
  </xsl:if>

</xsl:template>
</xsl:stylesheet>

```

出力 XML 文書

このシナリオでは、XSL 変換によって次の XML 文書が出力されます。この文書では、元の HTTP 応答に対して WebSEAL が実行する変更を定義します。

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponseChange>
  <Cookie action="add" name="NEW_COOKIE">
    <Content>2_orQUNJCbjdxqIEdDPMXj31UiHMXuU3hRCUtpN7xe6J1xZhxt0</Content>
    <Path>/</Path>
    <Domain>domainA.com</Domain>
    <Expires>Mon, 07 Jun 2021 10:12:14 GMT</Expires>
    <Secure>1</Secure>
  </Cookie>

```

```

    <HTTPOnly>0</HTTPOnly>
  </Cookie>
  <Cookie action="update" name="EXISTING_COOKIE">
    <Domain>domainB.com</Domain>
  </Cookie>
  <Cookie action="remove" name="OLD_COOKIE"></Cookie>
</HTTPResponseChange>

```

シナリオ 5: 既知の HTTP 要求に対する応答の生成

このシナリオでは、**HTTPResponseChange** 文書を使用して要求から応答を直接生成する方法を示します。このシナリオでは、HTTP 要求内に「**invalid-cookie**」という名前の Cookie が存在する場合、無効な Cookie が検出されたことを示す HTTP 応答が XSL 変換によって生成されます。

入力文書

このシナリオでは、以下の入力文書サンプルを使用します。

HTTP 要求

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPRequest>
  <RequestLine>
    <Method>GET</Method>
    <URI>/en/us/</URI>
    <Version>HTTP/1.1</Version>
  </RequestLine>
  <Headers>
    <Header name="User-Agent">curl%2F7.18.2%20(i486-pc-linux-gnu)%20libcurl%2F7.18.2%20openssl%2F0.9.8g%20zlib%2F1.2.3.3%20libidn%2F1.8</Header>
    <Header name="Host">www.ibm.com</Header>
    <Header name="Accept">*</Header>
  </Headers>
  <Cookies>
    <Cookie name="invalid-cookie">0</Cookie>
  </Cookies>
</HTTPRequest>

```

XSLT ルール

注: これらのルールは、関連付けられた POP により応答リソースとして定義される XSL 文書内に保管する必要があります。512 ページの『構成』を参照してください。

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">

  <!-- Firstly, strip any space elements -->
  <xsl:strip-space elements="*" />

  <!--
    Perform a match on the root of the document. Output the required
    HTTPRequestChange elements and then process templates.
  -->
  <xsl:template match="/">
    <xsl:apply-templates />
  </xsl:template>

  <!--
    Do nothing with Method
  -->

```

```

<xsl:template match="//HTTPRequest/RequestLine/Method" />

<!--
Do nothing with URI
-->
<xsl:template match="//HTTPRequest/RequestLine/URI"/>

<!--
Do nothing with Version
-->
<xsl:template match="//HTTPRequest/RequestLine/Version" />

<!--
Do nothing with Headers
-->
<xsl:template match="//HTTPRequest/Headers" />

<!--
Check for the presence of a cookie name 'invalid-cookie'
-->
<xsl:template match="//HTTPRequest/Cookies/Cookie">
  <xsl:choose>
    <xsl:when test="@name = 'invalid-cookie'">
      <HTTPResponseChange action="replace">
        <Version>HTTP/1.1</Version>
        <StatusCode>503</StatusCode>
        <Reason>Not Implemented</Reason>
        <Header name="Date" action="add">Thu%2C%2016%20Sep%202010%2010</Header>
        <Header name="Server" action="add">IBM_HTTP_Server</Header>
        <Header name="Content-Type" action="add">text%2Fhtml%3Bcharset%3DUTF-8</Header>
        <Header name="Content-Language" action="add">en-US</Header>
        <Body>%3Ch1%3EError%3C%2Fh1%3E%0A%3Cp%3EInvalid%20cookie%20%3C%2Fp%3E</Body>
      </HTTPResponseChange>
    </xsl:when>
  </xsl:choose>
</xsl:template>

</xsl:stylesheet>

```

出力 XML 文書

このシナリオでは、XSL 変換によって次の XML 文書が出力されます。この文書では、WebSEAL が元の HTTP 要求に対して生成する応答を定義します。

```

<?xml version="1.0" encoding="UTF-8"?>
<HTTPResponseChange action="replace">
  <Version>HTTP/1.1</Version>
  <StatusCode>503</StatusCode>
  <Reason>Not Implemented</Reason>
  <Header name="Date" action="add">Thu%2C%2016%20Sep%202010%2010</Header>
  <Header name="Server" action="add"></Header>
  <Header name="Content-Type" action="add">text%2Fhtml%3Bcharset%3DUTF-8</Header>
  <Header name="Content-Language" action="add">en-US</Header>
  <Body>%3Ch1%3EError%3C%2Fh1%3E%0A%3Cp%3EInvalid%20cookie%20%3C%2Fp%3E</Body>
</HTTPResponseChange>

```

変換エラー

ほとんどのエラーは、サーバー・ログに出力されるか、エラー・ページとしてブラウザに返されます。

注: さらに詳細な出力を得るには、**pdweb.http.transformation** コンポーネントを使用して HTTP 変換プロセスをトレースすることができます。このコンポーネントは、機密情報が含まれている可能性がある要求内のヘッダー情報をトレースします。例えば、BA ヘッダーなどです。

無効なルール・ファイル

無効なルール・ファイルが構成の一部として WebSEAL に入力されると、次のような結果になります。

- サーバーが始動に失敗する。
- 該当するエラーがログに記録される。

未定義のリソース

WebSEAL 構成ファイルで定義されていない POP 内のリソースを指定すると、警告メッセージがサーバー・ログに出力されます。

HTTP 変換エラー

HTTP 変換プロセスの実行中にエラーが発生すると、「501 Internal Server Error」がブラウザーに返されます。

WebSEAL エラー応答

エラーの原因となった junction またはオブジェクトに付加された該当する POP が存在する場合、WebSEAL エラー応答 (HTTP 変換プロセスとは無関係) は変換されます。

第 27 章 HTTP を介した Microsoft RPC

RPC over HTTP は、Microsoft Outlook クライアントが HTTP を介して Microsoft Exchange サーバーにアクセスするための Microsoft プロトコルです。RPC over HTTP プロトコルは、要求データ用と応答データ用にそれぞれ 1 つの HTTP 接続を使用します。これらの HTTP 接続は、長期間存続します。このプロトコルは、複数の要求/応答を単一の HTTP 要求にトンネリングします。

標準的な使用シナリオは、企業ネットワーク外にいる Outlook ユーザーが社内の Exchange サーバーに接続するケースです。この接続は、HTTP を使用してリバース・プロキシ（WebSEAL など）にアクセスすることによって実現されます。HTTP 接続は企業ネットワーク内で終了し、構成済みの IIS が RPC コマンドを Exchange サーバーにリレーします。

WebSEAL での RPC over HTTP のサポート

RPC over HTTP プロトコルのバージョン 2 が使用されている場合、WebSEAL はリバース・プロキシとして機能することができます。54 ページの『Microsoft RPC over HTTP のサポート』を参照してください。

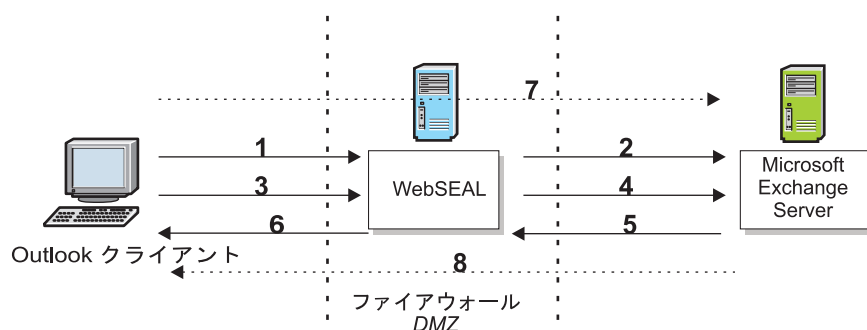


図 40. HTTP を介した WebSEAL RPC

図 40 は、RPC over HTTP プロトコルを使用する標準的なシナリオを示しています。

1. Microsoft Outlook が、WebSEAL への **RPC_IN_DATA** 要求を作成します。この要求の BA ヘッダーには、ユーザー名とパスワードが含まれています。
2. WebSEAL が要求を認証し、許可します。WebSEAL は要求 (BA ヘッダーを含む) を Microsoft Exchange に渡します。
3. Outlook が、WebSEAL への **RPC_OUT_DATA** 要求を作成します。この要求の BA ヘッダーには、ユーザー名とパスワードが含まれています。
4. WebSEAL が要求を認証し、許可します。WebSEAL は要求 (BA ヘッダーを含む) を Exchange に渡します。
5. Exchange が **200 OK** 応答を WebSEAL に返します。
6. WebSEAL が応答を Outlook に転送します。

7. Outlook が、**RPC_IN_DATA** 接続上の WebSEAL を介して、RPC データを Exchange に送信します。
8. Exchange が、**RPC_OUT_DATA** 接続上の WebSEAL を介して、RPC データを Outlook に送信します。

junction 構成

WebSEAL を Outlook と Exchange の間の RPC over HTTP 要求のリバース・プロキシとして使用するには、透過パス Junction または仮想ホスト junction を使用する必要があります。RPC over HTTP 要求の発行時、Outlook クライアントは、Exchange サーバーと通信するように構成されている junction IIS サーバーの URI `/rpc/rpcproxy.dll` へのアクセスを試みます。

WebSEAL および Exchange サーバーでユーザーを認証するには、junction を作成するときに **-b ignore** パラメーターを使用する必要があります。このパラメーターにより、WebSEAL が認証で使用する BA ヘッダーが、Exchange サーバーと通信する IIS サーバーでの認証でも必ず使用されるようになります。詳しくは、527 ページの『認証の制限』を参照してください。

この構成では SSL junction を使用する必要があります。Outlook は、BA 認証の使用時には HTTP をサポートしないからです。

透過パス Junction

次のコマンドは、透過パス Junction の作成方法を示しています。

```
server task instance_name-webseald-host_name create -t ssl
-h exchange_host -p exchange_port -b ignore -x /rpc
```

説明:

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。server list コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

exchange_host

Exchange サーバーの DNS ホスト名または IP アドレスを指定します。

exchange_port

Exchange サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。

仮想ホスト junction

次のコマンドは、仮想ホスト Junction の作成方法を示しています。

```
server task instance_name-webseald-host_name virtualhost create -t ssl
-h exchange_host -p exchange_port -v virtual_host -b ignore exchange
```

説明:

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。server list コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

exchange_host

Exchange サーバーの DNS ホスト名または IP アドレスを指定します。

exchange_port

Exchange サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。

virtual_host

Exchange サーバーに送信される要求の Host ヘッダーの値を指定します。

POP 構成

Outlook と Exchange の間の通信に使用される Junction 上をストリーミングする要求および応答を制御するには、以下のようにして POP を構成する必要があります。

1. **response-buffer-control** 属性と **request-buffer-control** 属性を **bypass** に設定する POP を作成します。例:

```
pdadmin> pop create streaming
pdadmin> pop modify streaming set attribute response-buffer-control bypass
pdadmin> pop modify streaming set attribute request-buffer-control bypass
```

2. この POP を、RPC over HTTP 通信に使用している Junction に付加します。

透過パス Junction の例:

```
pdadmin> pop attach /WebSEAL/webseal-server/rpc streaming
```

仮想ホスト Junction の例:

```
pdadmin> pop attach /WebSEAL/webseal-server/@exchange streaming
```

詳しくは、472 ページの『リソース単位でのバッファリングのバイパス』を参照してください。

認証の制限

Outlook によって WebSEAL に提示される資格情報は、RPC データ内と Exchange への認証で使用されます。WebSEAL ユーザー資格情報は、Exchange 用の AD 資格情報と一致しなければなりません。

HTTPS を介した BA ヘッダーを使用して Exchange に接続するように WebSEAL サーバーを構成する必要があります。**[ba]** スタンザ内の **ba-auth** の値を https に設定する必要があります。

```
ba-auth = https
```

ユーザーの BA 資格情報は、Junction を介して Exchange サーバーに渡す必要があります。Junction の作成時に、**-b ignore** オプションを指定します。

注:

1. Microsoft NT LAN Manager (NTLM) 認証はサポートされません。

2. **/RpcWithCert** エンドポイントはサポートされていません。このエンドポイントではクライアント証明書認証が必要です。WebSEAL は、クライアント証明書を使用して構成済みの Junction を認証することができません。

タイムアウトの考慮事項

RPC over HTTP 要求用のリバース・プロキシとして WebSEAL を構成する場合は、WebSEAL 構成ファイルの **[server]** スタンザ内に以下の値を設定します。

- `client-connect-timeout`
- `intra-connection-timeout`

これらのタイムアウト構成エントリは、Outlook クライアント接続に影響します。データ・ストリームのために接続をより長くオープン状態にしておくには、大きなタイムアウト値を使用します。例:

```
[server]
client-connect-timeout = 36000
intra-connection-timeout = 36000
```

これらのタイムアウト限度に達すると、Outlook クライアントは Exchange サーバーへの接続を再折衝します。

WebSEAL サーバー・ログのエラー

RPC over HTTP の性質上、ソケット・エラーが、WebSEAL サーバー・ログに出力される可能性があります。

注: **[logging]** スタンザ内の **server-log-cfg** 構成エントリは、ロギング・エージェント構成を指定します。

例:

```
2010-10-26-05:45:25.836+10:00I----- 0x38AD5424 webseald ERROR wiv socket
WsSslListener.cpp 1737 0xafb3fb90
DPWIV1060E Could not read from socket (406)
2010-10-26-05:45:25.838+10:00I----- 0x38AD5425 webseald ERROR wiv socket
WsSslListener.cpp 1658 0xafb3fb90
DPWIV1061E Could not write to socket (406)
```

これらのエラー・メッセージが表示されるのは、要求/応答ストリーミングがアクティブであり、完全な `content-length` が受信される前に接続がクローズされた場合です。

これらのエラー・メッセージが表示されないようにするには、WebSEAL 構成ファイルの **[ssl]** スタンザ内の **suppress-client-ssl-errors** を `true` に設定します。この構成ファイルの場所について詳しくは、38 ページの『構成ファイルの名前とロケーション』を参照してください。

ワーカー・スレッドの考慮事項

Microsoft Outlook は、RPC over HTTP を使用して Microsoft Exchange と通信するときに、複数の HTTP 接続を作成します。各 Outlook クライアント接続は、WebSEAL 内で複数のワーカー・スレッドを使用します。各 Outlook クライアントが WebSEAL 内で使用するワーカー・スレッドの数は、10 を超える場合があります。これらのスレッドは、Outlook クライアントが Exchange サーバーへの接続を維持する時間の間、保持されます。WebSEAL を経由する、すべての RPC over HTTP クライアント接続で、多数のワーカー・スレッドを使用すると、ワーカー・スレッドが枯渇する場合があります。

WebSEAL では、使用可能なワーカー・スレッドの数が限られています。スレッド数の制限により、任意の時点で WebSEAL にアクセス可能なクライアントの数が制限されます。Outlook および Exchange 環境内で RPC over HTTP とともに使用するために WebSEAL をデプロイする場合は、使用するワーカー・スレッドの数を考慮してください。詳細については、58 ページの『ワーカー・スレッドの割り振り』を参照してください。

第 28 章 コマンド・オプションの要約: 標準 junction

pdadmin ユーティリティには対話式コマンド行プロンプトが用意されていて、ユーザーは、これを使用して、WebSEAL junction タスクを実行できます。この付録では、標準 WebSEAL junction を作成するための **pdadmin server task** コマンドについて説明します。**pdadmin** ユーティリティの完全な構文の説明については、

「*IBM Security Access Manager for Web: Command Reference*」を参照してください。仮想ホストの junction に対するオプションについては、573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』で説明します。

注: また、Security Access Manager Web Portal Manager を使用して junction を管理できます。詳しくは、Web Portal Manager のオンライン・ヘルプ画面を参照してください。

この章で扱うトピックは以下のとおりです。

- 『pdadmin サーバー・タスクによる junction の作成』
- 532 ページの『junction のサーバー・タスク・コマンド』
- 533 ページの『初期サーバーの junction の作成』
- 541 ページの『既存 junction へのサーバーの追加』

pdadmin サーバー・タスクによる junction の作成

始める前に

pdadmin を使用する前に、**sec_master** などの管理許可を持つユーザーとしてセキュア・ドメインにログインする必要があります。

例:

```
pdadmin> login
Enter User ID: sec_master
Enter Password:
pdadmin>
```

手順

WebSEAL junction を作成するには、**pdadmin server task create** コマンドを使用します。

```
pdadmin> server task instance_name-webseald-host_name create options
```

例えば、単一の WebSEAL インスタンスの構成済みの名前が **web1** であり、**www.pubs.com** と名付けられたホストにインストールされている場合、完全なサーバー名は次のようになります。

```
web1-webseald-www.pubs.com
```

完全なサーバー名を正しい形式で表示するには、**pdadmin server list** コマンドを使用します。

```
pdadmin> server list
web1-webseald-www.pubs.com
```

詳しくは、707 ページの『付録 B. コマンド・リファレンス』または「*IBM Security Access Manager for Web: Command Reference*」の **pdadmin server task create** の参照ページを参照してください。

junction のサーバー・タスク・コマンド

pdadmin server task では以下の **junction** コマンドが使用できます。

コマンド	説明
add	既存の junction ポイントに新しいサーバーを追加します。 構文: <code>add -h host-name options junction-point</code> 541 ページの『既存 junction へのサーバーの追加』を参照してください。
create	初期サーバー用の新規 junction を作成します。 構文: <code>create -t type -h host-name options junction-point</code> 533 ページの『初期サーバーの junction の作成』を参照してください。
delete	指定した junction ポイントを除去します。 構文: <code>delete junction-point</code>
jmt load jmt clear	jmt load コマンドは、動的に生成されたサーバー相対 URL の処理を取り扱うための junction マッピング・テーブル・データ (<code>jmt.conf</code>) を WebSEAL に提供します。 jmt clear コマンドは、 junction マッピング・テーブル・データを WebSEAL から除去します。
list	このサーバーの構成済みのすべての junction ポイントをリストします。 構文: リスト

コマンド	説明
offline	この junction にあるサーバーをオフライン操作状態に設定します。指定されたサーバーに追加で送信される要求はありません。サーバーが指定されていない場合、この junction にあるすべてのサーバーがオフライン操作状態に設定されます。 構文: <pre>offline [-i server_uuid] junction_point</pre>
online	この junction にあるサーバーをオンライン操作状態に設定します。サーバーは通常操作を再開します。サーバーが指定されていない場合、この junction にあるすべてのサーバーがオンライン操作状態に設定されます。 構文: <pre>online [-i server_uuid] junction_point</pre>
remove	junction ポイントから指定したサーバーを除去します。 構文: <pre>remove -i server-id junction-point</pre> 特定のサーバーの ID を判別するには、show コマンドを使用します。
show	junction の詳細を表示します。 構文: <pre>show junction-point</pre>
throttle	この junction にあるサーバーをスロットル操作状態に設定します。この状態にある間は、確立済みのユーザー・セッションからの要求のみが処理されます。サーバーが指定されていない場合、この junction にあるすべてのサーバーがスロットル操作状態に設定されます。 構文: <pre>throttle [-i server_uuid] junction_point</pre>

初期サーバーの junction の作成

操作: 新規 junction ポイントを作成し、初期サーバーを junction します。

構文:

```
create -t type -h host-name options junction-point
```

junction タイプ

<code>-t type</code>	Junction のタイプ。tcp、ssl、tcpproxy、sslproxy、local、mutual のいずれかです。 <code>-t tcp</code> のデフォルトのポートは 80 です。<code>-t ssl</code> のデフォルトのポートは 443 です。 **必須**。422 ページの『標準 WebSEAL junction の構成』を参照してください。
ホスト名	
<code>-h host-name</code>	ターゲット・バックエンド・サーバーの DNS ホスト名または IP アドレス。 **必須**。422 ページの『標準 WebSEAL junction の構成』を参照してください。
一般オプション	
標準 junction タイプ	
<code>-a address</code>	ターゲット・バックエンド・サーバーとの通信時に WebSEAL が使用するローカル IP アドレスを指定します。このオプションを指定しない場合、WebSEAL ではオペレーティング・システムにより指定されるデフォルトのアドレスを使用します。 特定の junction にアドレスを指定した場合、WebSEAL は、指定されたローカル・アドレスにバインドして、その junction 先サーバーとのすべての通信を行うように変更されます。
<code>-f</code>	既存の junction を強制的に置き換えます。 453 ページの『新規 junction の強制適用』を参照してください。
<code>-i</code>	WebSEAL サーバーは、URL を大文字小文字を区別せずに処理します。 467 ページの『大文字小文字を区別しない URL のサポート』を参照してください。

<code>-q location</code>	WebSEAL に query_contents プログラム・ファイルの正しい名前とそのファイルの検索場所を提供します。デフォルトでは、Windows ファイルは query_contents.exe と呼ばれ、UNIX ファイルは query_contents.sh と呼ばれます。デフォルトでは、WebSEAL は、バックエンド Web サーバーの cgi_bin ディレクトリーでファイルを探します。 バックエンド Windows および UNIX Web サーバーの場合は必須です。 438 ページの『Windows ベースの Web サーバーでの query_contents のインストールおよび構成』を参照してください。
<code>-T resource/resource-group</code>	GSO リソースまたはリソース・グループの名前。-b gso オプションの場合に必須で、このオプションでのみ使用されます。 602 ページの『GSO 使用可能 WebSEAL junction の構成』を参照してください。
<code>-w</code>	Windows ファイル・システム・サポート。 468 ページの『Windows ファイル・システムへの junction』を参照してください。
TCP および SSL junction タイプ	
<code>-p port</code>	バックエンドのサード・パーティー・サーバーの TCP ポート。デフォルトは、TCP junction の場合は 80 で、SSL junction の場合は 443 です。 423 ページの『TCP タイプの標準 junction の作成』および 423 ページの『SSL タイプの標準 junction の作成』を参照してください。
ステートフル junction 448 ページの『ステートフル junction』を参照してください。	
<code>-s</code>	Junction がステートフル・アプリケーションをサポートすることを指定します。デフォルトでは、junction はステートフルではありません。
<code>-u UUID</code>	ステートフル junction (-s) を使用して WebSEAL に接続されたバックエンド・サーバーの UUID を指定します。
mutual junction 448 ページの『ステートフル junction』を参照してください。	

-p <i>HTTP port</i>	バックエンド・サード・パーティー・サーバーの HTTP ポート。 424 ページの『mutual junction の作成』を参照してください。
-P <i>HTTPS port</i>	バックエンド・サード・パーティー・サーバーの HTTPS ポート。 424 ページの『mutual junction の作成』を参照してください。
基本認証および SSL 証明書を介した相互認証 443 ページの『相互認証 SSL junction』を参照してください。	
-B	WebSEAL は、BA ヘッダー情報を使用して、バックエンド・サーバーに対して認証します。-U および -W オプションが必要です。
-D " <i>DN</i> "	バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張するものです。
-K " <i>key-label</i> "	バックエンド・サーバーに対して認証するのに使用される WebSEAL のクライアント・サイド証明書の鍵ラベル。
-U " <i>username</i> "	WebSEAL ユーザー名。-B と共に使用して、BA ヘッダー情報をバックエンド・サーバーに送信します。
-W " <i>password</i> "	WebSEAL パスワード。-B と共に使用して、BA ヘッダー情報をバックエンド・サーバーに送信します。
プロキシ junction (-t tcpproxy または -t sslproxy が必要) 446 ページの『TCP および SSL のプロキシ junction の作成』を参照してください。	
-H <i>host-name</i>	プロキシ・サーバーの DNS ホスト名または IP アドレス。
-P <i>port</i>	プロキシ・サーバーの TCP ポート。
HTTP ヘッダーへの ID 情報の提供	
-b <i>BA-value</i>	WebSEAL サーバーがバックエンド・サーバーに HTTP 基本認証 (BA) ヘッダーのクライアント識別情報を渡す方法を定義します。次のいずれか 1 つになります。 filter (デフォルト)、 ignore 、 supply 、 gso 590 ページの『HTTP BA ヘッダーを使用したシングル・サインオン』を参照してください。

<code>-c header-types</code>	junction を介して HTTP ヘッダーに、Security Access Manager に固有のクライアント識別情報を挿入します。 <i>header-types</i> 引数には、Access Manager HTTP ヘッダー・タイプ iv-user、iv-user-l、iv-groups、iv-creds、all を任意に組み合わせて組み込むことができます。 596 ページの『HTTP ヘッダーで提供される ID 情報』を参照してください。
<code>-e encoding-type</code>	Junction に対して HTTP ヘッダーを生成するときに使用するエンコード方式を指定します。このエンコード方式は、-c Junction オプションと tag-value の両方を指定して生成されたヘッダーに適用されます。エンコードに可能な値は以下のとおりです。 • utf8_bin • utf8_uri • lcp_bin • lcp_uri 471 ページの『HTTP ヘッダー・データ用の UTF-8 エンコード』を参照してください。
<code>-I</code>	Cookie の処理: <code>-I</code> により、Set-Cookie ヘッダー名属性が固有であることが保証されます。 504 ページの『Cookie の処理: <code>-I</code> による固有の Set-Cookie 名前属性の確認』を参照してください。
<code>-j</code>	スクリプト生成のサーバー相対 URL を処理する junction 識別を Cookie に入れて提供します。 493 ページの『junction Cookie を使用したサーバー相対 URL の変更』を参照してください。

-J {trailer,inhead, onfocus,xhtml10}	junction Cookie JavaScript ブロックをコントロールします。 -J trailer は、junction Cookie JavaScript をバックエンド・サーバーから戻される HTML ページの（前ではなく）後ろに付加するときに使用します。 -J inhead は、HTML 4.01 準拠のために <head> タグと </head> タグの間に JavaScript ブロックを挿入するときに使用します。 -J onfocus は、JavaScript で onfocus イベント・ハンドラーを使用して、正しい junction Cookie が「複数の junction と複数のブラウザー・ウィンドウ」シナリオで使われるようにするときに使用します。 -J xhtml10 は、HTML 4.01 および XHTML 1.0 準拠である JavaScript ブロックを挿入するときに使用します。 このオプションについて詳しくは、495 ページの『junction Cookie JavaScript ブロックのコントロール』を参照してください。
-k	セッション Cookie をバックエンド・ポータル・サーバーに送信します。 466 ページの『Junction ポータル・サーバーへのセッション Cookie の引き渡し』を参照してください。
-n	非ドメイン Cookie の名前を変更しないように指定します。クライアント・サイド・スクリプトが Cookie の名前に依存している場合に使用してください。 デフォルトでは、junction が JMT にリストされている場合や -j junction オプションが使用されている場合、junction から戻される非ドメイン Cookie の名前の前に AMWEBJCT_junction_point_ を付加します。 503 ページの『Cookie 名の保存』を参照してください。
-r	junction を介して HTTP ヘッダーに着信 IP アドレスを挿入します。 599 ページの『HTTP ヘッダーのクライアント IP アドレス (-r)』を参照してください。
junction フェアネス 59 ページの『junction 用のワーカー・スレッドの割り振り (Junction フェアネス)』を参照してください。	
-I percent-value	ワーカー・スレッドの消費量のソフト限界を定義します。
-L percent-value	ワーカー・スレッドの消費量のハード限界を定義します。

WebSphere シングル・サインオン (LTPA) junction 604 ページの『IBM WebSphere へのシングル・サインオン (LTPA)』を参照してください。	
-A	junction が LTPA Cookie (トークン) をサポートできるようにします。LTPA バージョン 1 Cookie (LtpaToken) および LTPA バージョン 2 Cookie (LtpaToken2) の両方がサポートされます。LTPA バージョン 1 Cookie は、デフォルトで指定されています。LTPA バージョン 2 Cookie は、 -2 オプションを追加で使用して指定する必要があります。 -F および -Z オプションも必要です。
-2	-A オプションで使用するこのオプションは、LTPA バージョン 2 Cookie (LtpaToken2) が使用されることを指定します。 -2 オプションを指定しない -A オプションは、LTPA バージョン 1 Cookie (LtpaToken) が使用されることを指定します。
-F "keyfile"	LTPA Cookie データの暗号化に使用する鍵ファイルの名前。 -A オプションの場合のみ有効です。
-Z "keyfile-password"	LTPA Cookie データの暗号化に使用される鍵格納ファイルのパスワード。 -A オプションの場合のみ有効です。
Tivoli Federated Identity Manager SSO junctions 587 ページの『Tivoli Federated Identity Manager を使用したシングル・サインオン』	
-Y	junction の Tivoli Federated Identity Manager シングル・サインオン (SSO) を使用可能にします。 注: このオプションを使用する前に、まず、junction 全体で Tivoli Federated Identity Manager シングル・サインオンをサポートするように WebSEAL 構成ファイルを構成する必要があります。
WebSEAL 間 SSL junction 447 ページの『SSL を介した WebSEAL から WebSEAL への junction』を参照してください。	
-C	SSL を介したフロントエンド WebSEAL サーバーとバックエンド WebSEAL サーバーの間の相互認証。 -t ssl または -t sslproxy タイプが必要です。
フォーム・シングル・サインオン 607 ページの『フォーム・シングル・サインオン認証』を参照してください。	
-S <i>file_name</i>	フォーム・シングル・サインオン構成ファイルの名前。

仮想ホスト 470 ページの『仮想ホストへの標準 Junction』を参照してください。	
<code>-v virtual-host-name[:HTTP-port]</code>	バックエンド・サーバー上で表される仮想ホスト名。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。 <code>mutual</code> の <code>junction</code> では、この値は HTTP 要求に使用される仮想ホストに対応します。 バックエンド <code>junction</code> サーバーの仮想インスタンスの 1 つに結合中であるために、そのサーバーが Host ヘッダーを予期している場合は、<code>-v</code> を使用してください。ブラウザーからのデフォルト HTTP ヘッダー要求は、バックエンド・サーバーに複数の名前と複数の仮想サーバーがあることを認識しません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。
<code>-V virtual-host-name[:HTTPS-port]</code>	バックエンド・サーバー上で表される仮想ホスト名。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。この値は HTTPS 要求に使用される仮想ホストに対応します。この値は <code>mutual</code> な <code>junction</code> にのみ使用されます。 バックエンド <code>junction</code> サーバーの仮想インスタンスの 1 つに結合中であるために、そのサーバーが Host ヘッダーを予期している場合は、<code>-v</code> を使用してください。ブラウザーからのデフォルトの HTTPS ヘッダー要求は、バックエンド・サーバーが複数の名前と複数の仮想サーバーを持っていることを認識していません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。
透過的 junction 427 ページの『透過パス Junction』を参照してください。	
<code>-x</code>	透過パス Junction を作成します。
Junction ポイント	
バックエンド・アプリケーション・サーバーの名前空間のルートがマウントされている WebSEAL 名前空間にあるロケーションの名前。 **必須**。422 ページの『標準 WebSEAL junction の構成』を参照してください。	

既存 junction へのサーバーの追加

操作: 既存の junction ポイントに新しいサーバーを追加します。

構文:

```
add -h host-name options junction-point
```

ホスト名	
<code>-h host-name</code>	追加するターゲット・バックエンド・サーバーの DNS ホスト名または IP アドレス。 **必須**。422 ページの『標準 WebSEAL junction の構成』を参照してください。
一般オプション	
標準 junction タイプ	
<code>-a address</code>	ターゲット・バックエンド・サーバーとの通信時に WebSEAL が使用するローカル IP アドレスを指定します。このオプションを指定しない場合、WebSEAL ではオペレーティング・システムにより指定されるデフォルトのアドレスを使用します。 特定の junction のアドレスを指定すると、WebSEAL は、その junction サーバーとのすべての通信の場合にこのローカル・アドレスにバインドします。
<code>-i</code>	WebSEAL サーバーは、URL を大文字小文字を区別せずに処理します。 467 ページの『大文字小文字を区別しない URL のサポート』を参照してください。
<code>-q url</code>	query_contents スクリプトの相対パス。デフォルトでは、WebSEAL は <code>/cgi_bin/</code> 内を検索して query_contents を見つけようとします。これとは別のディレクトリーに入っている場合、または query_contents のファイル名が異なる場合は、このオプションを使用して、該当のファイルへの新しい URL を WebSEAL に指示してください。バックエンド Windows サーバーの場合は必須です。 438 ページの『Windows ベースの Web サーバーでの query_contents のインストールおよび構成』を参照してください。
<code>-w</code>	Windows ファイル・システム・サポート。 468 ページの『Windows ファイル・システムへの junction』を参照してください。
TCP および SSL junction タイプ	

<code>-p port</code>	バックエンドのサード・パーティー・サーバーの TCP ポート。デフォルトは、TCP junction の場合は 80 で、SSL junction の場合は 443 です。 423 ページの『TCP タイプの標準 junction の作成』および 423 ページの『SSL タイプの標準 junction の作成』を参照してください。
mutual junction タイプ	
<code>-p HTTP-port</code>	バックエンド・サード・パーティー・サーバーの HTTP ポート。 424 ページの『mutual junction の作成』を参照してください。
<code>-P HTTPS-port</code>	バックエンド・サード・パーティー・サーバーの HTTPS ポート。 424 ページの『mutual junction の作成』を参照してください。
ステートフル junction 448 ページの『ステートフル junction』を参照してください。	
<code>-u UUID</code>	ステートフル junction (-s) を介して WebSEAL に接続されたバックエンド・サーバーの UUID を指定します。
SSL を介した相互認証 443 ページの『相互認証 SSL junction』を参照してください。	
<code>-D "DN"</code>	バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張するものです。
プロキシ junction (-t tcpproxy または -t sslproxy が必要) 446 ページの『TCP および SSL のプロキシ junction の作成』を参照してください。	
<code>-H host-name</code>	プロキシ・サーバーの DNS ホスト名または IP アドレス。
<code>-P port</code>	プロキシ・サーバーの TCP ポート。
仮想ホスト 470 ページの『仮想ホストへの標準 Junction』を参照してください。	

<code>-v virt-host-name</code>	バックエンド・サーバー上で表される仮想ホスト名。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。 <code>mutual</code> の <code>junction</code> では、この値は HTTP 要求に使用される仮想ホストに対応します。 バックエンド <code>junction</code> サーバーの仮想インスタンスの 1 つに <code>junction</code> しているので、そのサーバーがホスト名ヘッダーの受信を予期している場合は、-V を使用してください。ブラウザからのデフォルトの HTTP ヘッダー要求は、バックエンド・サーバーが複数の名前と複数の仮想サーバーを持っていることを認識していません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。
<code>-V virt-host-name</code>	バックエンド・サーバー上で表される仮想ホスト名。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。この値は HTTPS 要求に使用される仮想ホストに対応します。この値は <code>mutual</code> な <code>junction</code> にのみ使用されます。 バックエンド <code>junction</code> サーバーの仮想インスタンスの 1 つに <code>junction</code> しているので、そのサーバーがホスト名ヘッダーの受信を予期している場合は、-V を使用してください。ブラウザからのデフォルトの HTTPS ヘッダー要求は、バックエンド・サーバーが複数の名前と複数の仮想サーバーを持っていることを認識していません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。
Junction ポイント	
この既存の <code>junction</code> ポイントにサーバーを追加します。	

第 8 部 仮想ホスト機能 (Virtual Hosting)

第 29 章 仮想ホスト junction

この章では、以下のトピックについて説明します。

- 『仮想ホスト junction の概念』
- 552 ページの『リモート・タイプ仮想ホスト junction の作成』
- 556 ページの『シナリオ 1: リモート仮想ホスト junction』
- 558 ページの『仮想ホスト junction のインターフェースの定義』
- 560 ページの『シナリオ 2: インターフェースを持つ仮想ホスト junction』
- 564 ページの『既存の WebSEAL 機能での仮想ホストの使用』
- 568 ページの『シナリオ 3: 仮想ホストの拡張構成』
- 570 ページの『仮想ホスト junction の制限』

仮想ホスト junction の概念

このセクションは、以下のトピックで構成されています。

- 『標準 WebSEAL junction』
- 548 ページの『URL フィルターの課題』
- 548 ページの『仮想ホスティング』
- 548 ページの『仮想ホスト Junction のソリューション』
- 550 ページの『仮想ホスト junction が無視するスタンザおよびスタンザ・エンタリー』
- 551 ページの『オブジェクト・スペース内で表される仮想ホスト』

標準 WebSEAL junction

Security Access Manager は、保護されたバックエンド Web アプリケーション・サーバーへの要求を認証し許可するための製品です。バックエンド Web サーバーは、サーバーへの直接アクセスをブロックし、その要求を WebSEAL ヘルパーティングすることで保護されます。WebSEAL では、各要求を受け入れ、許可されればその要求を保護された Web サーバーに渡します。WebSEAL は、何らかの応答をクライアントに戻します。

WebSEAL は、単一のホスト Web サーバーとして動作します。WebSEAL が多数のバックエンド Web サーバーを保護しながら、単一のホスト・サーバーとして動作し続けられるようにするために、WebSEAL ではすべてのバックエンド・サーバーの文書スペースが単一の文書スペースにマージされます。*junctioning* という用語は、バックエンド Web サーバーの文書スペースを WebSEAL の単一で統一された仮想文書スペースにマージするアクションを表しています。

junction 間での通信を成功させるためには、WebSEAL の単一のホスト文書スペースの一部として表示されたときにその URL が正しくなるように、WebSEAL で、保護された Web サーバーから戻された HTML 応答文書の絶対 URL およびサーバー相対 URL をフィルターする必要があります。WebSEAL の junction 機能は、

junction 先バックエンド・システム上のリソースにアクセスするために必要なサーバーおよびパスの情報を変更します。URL に junction の識別が含まれていないと、junction バックエンド・サーバー上のリソースへのリンクは失敗します。

URL フィルターの課題

WebSEAL は、junction バックエンド・アプリケーション・サーバーからの応答で戻される URL のフィルターと処理に対して、多数のソリューションをサポートしています。これらのソリューションでは常に、WebSEAL で URL を検索して HTML コンテンツを構文解析する必要があります。HTML は進化し続ける複雑な仕様であるため、HTML の構文解析も同様に複雑です。

さらに難しいのは、クライアント・サイドで動的に URL を生成してフィルターを回避する組み込み JavaScript の機能です。標準 junction にわたる URL フィルターの処理についての WebSEAL ソリューションの詳細は、475 ページの『第 25 章 Junction リソースへの URL の変更』に記載しています。

仮想ホスティング

仮想ホスティング という用語は、1 つのマシン上に複数のサーバーを明確なホスト名で区別して維持するという動作を表しています。仮想ホスティングでは、まったく別のサイトとして表示される複数の Web サービスをそれぞれ異なるホスト名と URL で実行できます。例えば、以下の 2 つの仮想ホストは同一マシン上に存在することができます。

- www.exampleA.com
- www.exampleB.com

仮想ホストは、ホスト・サーバーの文書スペースの固有のセクションを使用するように内部的に構成されています。ただし、外部ユーザーが必要なのは個別のドメイン名を使用して仮想ホストを参照するのみで、その他の追加のパス情報を知る必要はありません。

HTTP/1.1 の仕様では、クライアント・ブラウザーはすべての要求に HTTP **Host** ヘッダーを含む必要があるため、仮想ホスティングを使用したリソースへのアクセスが可能となります。**Host** ヘッダーには、要求されたリソースが置かれたサーバーのホスト名が記載されます。

仮想ホスト Junction のソリューション

WebSEAL では仮想ホスティングをサポートします。仮想ホスト Junction を介することにより、URL フィルターの制限がなくなります。

WebSEAL は仮想ホスト Junction を使用して、ローカルまたはリモート仮想ホストと通信することができます。WebSEAL では、クライアント要求に HTTP **Host** ヘッダーを使用して、それらの要求を Junction サーバーまたはローカル・コンピューター上にある適切な文書スペースに送信します。

ユーザーは、WebSEAL サーバーのホスト名と変更される可能性のあるリソース・パス (<http://webseal/junction/resource>) を間接的に使用するのではなく、Junction サーバーのホスト名 (<http://protected-server/resource>) を直接使用し

て、リソースにアクセスできます。 junction サーバーのホスト名を使用したリソースへの直接アクセスでは、URL フィルターは不要です。

仮想ホスト Junction では、Junction Web サーバー上の元のコンテンツと同じ形式で応答ページのコンテンツを保存します。クライアントは、これらの応答ページ上にある変更されていない絶対 URL リンクおよびサーバー相対 URL リンクを使用して、リソースを正しく見つけることができます。

仮想ホスト junction の構成では、外部 DNS により、すべての仮想ホスト名を WebSEAL サーバーの IP アドレス (複数可) にマップする必要があります。ユーザーが junction サーバーのホスト名への要求を作成すると、その要求は WebSEAL に送付されます。

HTTP/1.1 仕様では、要求に HTTP **Host** ヘッダーを含めることが求められています。WebSEAL は、要求の URL ではなく、**Host** ヘッダーの値を使用して、要求をディスパッチングするのに適切な仮想ホスト junction を選択します。

WebSEAL では適切な仮想ホスト Junction を次のように選択します。

- **Host** ヘッダーが要求内にあり、その値が構成済み仮想ホスト Junction のホスト名と一致する場合、WebSEAL では仮想ホスト Junction が使用されます。

仮想ホストがプロトコルの標準外のポートを使用する場合は、**Host** ヘッダーにポート番号を含める必要があります。TCP の標準ポートは 80、SSL の標準ポートは 443 です。

WebSEAL が仮想ホスト Junction に送信する **Host** ヘッダーには、仮想ホスト Junction の作成時に **-v** によって指定された値が含まれています。

- その他の場合、WebSEAL は要求の URL に基づき、標準 Junction を使用します。例えば、WebSEAL は以下の状況で標準 Junction を使用します。
 - **Host** ヘッダーの値がどの仮想ホスト Junction (**-v vhost_name[:port]**) にも一致しない場合。
 - HTTP/1.0 要求などに **Host** ヘッダーがない場合。

デフォルトでは、仮想ホスト Junction が標準 Junction よりも優先されます。

[junction] スタンザで **match-vhj-first** 構成エントリーを使用して、この動作を逆にすることができます。この構成エントリーが **no** の場合、WebSEAL は要求に一致する標準 Junction を検索します。一致するものがない場合、WebSEAL は **Host** ヘッダーを確認し、仮想ホスト Junction が要求を処理できるかどうかを判別します。

セッションを維持する環境で **match-vhj-first** を **no** に設定する場合は、**shared-domain-cookie** 構成エントリーを **yes** に設定する必要があります。デフォルトでは、WebSEAL は仮想ホスト Junction ごとに別のセッション・キャッシュを維持し、すべての標準 Junction に対して別のセッション・キャッシュを維持します。セッション・アフィニティを維持するには、**shared-domain-cookie** パラメーターを使用して、要求を処理する Junction に関係なく、WebSEAL が同じセッションを使用できるようにします。また、**session-cookie-domain** スタンザを使用して、セッション Cookie を共用するドメインを指定することもできます。これらの

構成エントリーについて詳しくは、「*Security Access Manager WebSEAL 構成スタンザ・リファレンス*」を参照してください。

標準 Junction とは異なり、WebSEAL は仮想ホスト Junction を文書スペースのマウント・ポイントとして定義しません。WebSEAL は、常に WebSEAL の文書スペースのルートにある仮想ホスト Junction の指定により、仮想ホスト・リソースにアクセスします。この指定は、*仮想ホスト・ラベル* と呼ばれます。

仮想ホスト名が **HTTP Host** ヘッダー内にある Junction サーバーは、サーバー相対 URL または絶対 URL が含まれる可能性がある独自の応答を返します。WebSEAL は、その応答をフィルターせずに直接クライアントに戻します。Junction サーバーからの応答に含まれる、そのサーバー自体を参照する絶対 URL は、サーバー IP アドレスではなく、仮想ホスト名を使用する必要があります。クライアントは、これらの応答ページ上にある変更されていない絶対 URL リンクおよびサーバー相対 URL リンクを使用して、リソースを正しく見つけることができます。

仮想ホスト Junction のその他の機能には次のようなものがあります。

- HTTP プロトコルと HTTPS プロトコルの両方のサポート。クライアントと WebSEAL 間で両方のプロトコルをサポートするには、2 つの仮想ホスト Junction を使用する必要があります。プロトコルごとに別の仮想ホスト Junction を使用します。

仮想ホスト Junction は単一のプロトコル (ポート) にのみ応答します。Junction サーバーは、そのサーバー上で固有の SSL と TCP ポートを使用して 2 つの Junction を作成できるように、HTTP と HTTPS の両方をサポートする必要があります。

- WebSEAL は、独自の複数のローカル仮想ホスト Junction を提供して、保護されたローカル・コンテンツを提供することもできます。
- 仮想ホスト junction では、多数の構成オプションを標準 WebSEAL junction と共有しています。仮想ホスト Junction ではいくつかの標準 Junction オプションがサポートされていません。仮想ホスト junction に固有の新規オプションもいくつかあります。
- 各 WebSEAL インスタンスでは、複数のインターフェースまたはポート、あるいはその両方をサポートできます。複数のインターフェースおよびポートを使用して、listen を行う HTTPS インターフェースとポートごとに SSL 証明書を構成することができます。

仮想ホスト名が、WebSEAL が listen 中のさまざまなインターフェースまたはポートに解決される場合、WebSEAL はさまざまな証明書を接続クライアントに提示できます。

仮想ホスト junction が無視するスタンザおよびスタンザ・エントリー

仮想ホスト junction は、junction サーバーからの HTML 応答ページにある URL の構文解析やフィルターを行いません。このため、以下の WebSEAL 構成ファイルのスタンザおよびスタンザ・エントリーは、仮想ホスト junction では無視されます。

- [server]、preserve-base-href
- [server]、process-root-requests
- [process-root-filter]
- [junction]、jmt-map
- [filter-url]
- [filter-events]
- [filter-schemes]
- [filter-content-types]
- [script-filtering]
- [preserve-Cookie-names]
- [junction]、allow-backend-domain-cookies

オブジェクト・スペース内で表される仮想ホスト

標準 WebSEAL junction は、保護オブジェクト・スペース内では WebSEAL の文書スペース内のマウント・ポイントとして以下の形式で表されます。

```
/WebSEAL/instance-name/junction-name/path
```

仮想ホスト junction は、常に WebSEAL の文書スペースのルートにある仮想ホスト junction の指定により、保護オブジェクト・スペース内に表されます。この指定は、仮想ホスト・ラベル と呼ばれます。例:

```
/WebSEAL/instance-name/@vhost-label/path
```

以下の例に、ラベル、**support.ibm.com-http** の付いた仮想ホスト junction 上のディレクトリー、pubs にあるリソース、readme.html での表記を示します。保護オブジェクト・スペースでの絶対パスは次のように表示されます。

```
/WebSEAL/instance-name/@support.ibm.com-http/pubs/readme.html
```

-g オプションを使用して作成された仮想ホスト junction も保護オブジェクト・スペース内に表示されるので、管理 ACL をその junction 上に配置することができます。ただし、その junction で保護されたディレクトリーおよびリソースは表示されません。それらのディレクトリーおよびリソースは、1 次 junction でのみ可視となります (**-g** オプションにより、WebSEAL は強制的に単一のオブジェクト・スペースのみを認識するようになります)。例:

```
/WebSEAL/instance-name/@support.ibm.com-http/pubs/readme.html
/WebSEAL/instance-name/@support.ibm.com-https
```

仮想ホスト junction の構成

このセクションは、以下のトピックで構成されています。

- 552 ページの『リモート・タイプ仮想ホスト junction の作成』
- 554 ページの『ローカル・タイプ仮想ホスト junction の作成』

リモート・タイプ仮想ホスト junction の作成

pdadmin ユーティリティーの **server task...virtualhost** コマンドを使用すれば、仮想ホスト junction を構成できます。また、Web ポータル・マネージャー (WPM) を使用して仮想ホスト junction を構成することもできます。

以下の例 (1 行で入力します) では、**pdadmin server task virtualhost create** コマンドの構文 (1 行で入力します) を指定しています。

```
pdadmin> server task instance_name-webseald-host_name virtualhost create
options vhost-label
```

以下の表では、共通で必須の **virtualhost create** オプションについて説明しています。

表 39. リモート・タイプの仮想ホスト junction のオプション

オプション	説明
-t type	Junction のタイプ。 tcp 、 ssl 、 tcpproxy 、 sslproxy のいずれかです。 すべての仮想ホスト junction に必須。
-h host-name	ターゲット・バックエンド・サーバーの DNS ホスト名または IP アドレス。 TCP junction および SSL junction に同一のホスト名を使用できます。各仮想ホストのポートによりそれぞれが区別されるため、それぞれが固有のものとして考慮されます。 tcp 、 ssl 、 tcpproxy 、および sslproxy タイプの junction で必須。
-v vhost-name[:port]	要求の HTTP Host ヘッダーが、 -v オプションで指定されている仮想ホスト名およびポート番号に一致する場合、WebSEAL は要求を処理するために仮想ホスト Junction を選択します。 -v オプションは、バックエンド・サーバーに送信される要求の Host ヘッダーの値を指定するためにも使用されます。 仮想ホストがプロトコルの標準外のポートを使用する場合、ポート番号が必要です。TCP の標準ポートは 80、SSL の標準ポートは 443 です。 tcp 、 ssl 、 tcpproxy 、および sslproxy タイプの junction に -v が指定されていない場合、junction は、 -h host および -p port オプション (またはそのデフォルト) に記載された情報から選択されます。

表 39. リモート・タイプの仮想ホスト *junction* のオプション (続き)

オプション	説明
<code>-g vhost-label</code>	クライアントと WebSEAL 間で HTTP および HTTPS の両方のプロトコルをサポートする必要がある場合、同一の仮想ホスト (-h) に対してプロトコル (-t) ごとに 1 つずつ、計 2 つの <i>junction</i> が必要になります。デフォルトでは、<i>junction</i> (プロトコルによって区別されるもののみ) が単一のオブジェクト・スペースをポイントしている場合でも、各 <i>junction</i> はそれ自体の固有の保護オブジェクト・スペースを認識します。 -g オプションを使用すると、2 番目に追加された <i>junction</i> は、最初の <i>junction</i> と同じ保護オブジェクト・スペースを共用します。この単一オブジェクト・スペースの参照により、ユーザーは各保護オブジェクト上で単一アクセス制御リスト (ACL) を維持できます。 最初の <i>junction</i> に対して -g を使用した 2 番目の仮想ホスト <i>junction</i> が存在する場合、最初の仮想ホスト <i>junction</i> は削除できません。このような操作を試行すると、エラー・メッセージが戻されます。 このオプションは、Junction ペア (補完プロトコルを使用する 2 つの Junction) にのみ該当します。このオプションでは、2 つを超える Junction の関連付けはサポートしていません。 これはオプションです。

仮想ホスト・ラベル:

仮想ホスト・ラベル (*vhost-label*) は、単に、仮想ホスト *junction* の名前です。

- *junction* のラベルは、保護オブジェクト・スペース (Web ポータル・マネージャー (WPM)) の表示内で *junction* を表すのに使用されます。
- デフォルトでは、仮想ホスト *junction* は、常に WebSEAL オブジェクト・スペースのルートにマウントされています。
- **pdadmin** ユーティリティでは、このラベルを使用して *junction* を参照できます。
- 仮想ホスト *junction* のラベルは、WebSEAL の各インスタンス内で固有である必要があります。
- ラベルは、保護オブジェクト・スペース内で仮想ホスト *junction* を表すのに使用されるため、ラベル名にスラッシュ文字 (/) を含めることはできません。

TCP および SSL 仮想ホスト *junction* の例:

556 ページの『シナリオ 1: リモート仮想ホスト *junction*』を参照してください。

560 ページの『シナリオ 2: インターフェースを持つ仮想ホスト *junction*』を参照してください。

参照:

virtualhost junction コマンドの要約については、573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

pdadmin ユーティリティの完全な構文の説明については、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。

ローカル・タイプ仮想ホスト junction の作成

ローカル仮想ホスト junction (**-t localtcp** および **-t localssl**) は、WebSEAL サーバー上にローカルに存在する特定のコンテンツのマウント・ポイントです。Junction されたリモート・サーバーのコンテンツと同様に、ローカル junction のコンテンツは、WebSEAL の統一された保護オブジェクト・スペース・ビューに組み込まれます。

以下のオプションは、ローカル仮想ホスト junction に適しています。

表 40. ローカル・タイプの仮想ホスト junction のオプション

オプション	説明
-t type	junction のタイプ (localtcp または localssl)。必須
-g vhost-label	-g オプションを使用すると、2 番目に追加された junction は、最初の junction と同じ保護オブジェクト・スペースを共有します。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。
-v vhost-name[:port]	要求の HTTP Host ヘッダーが、 -v オプションで指定されている仮想ホスト名およびポート番号に一致する場合、WebSEAL は要求を処理するために仮想ホスト Junction を選択します。 -v オプションは、バックエンド・サーバーに送信される要求の Host ヘッダーの値を指定するためにも使用されます。 仮想ホストがプロトコルの標準外のポートを使用する場合、ポート番号が必要です。TCP の標準ポートは 80、SSL の標準ポートは 443 です。 localtcp および localssl タイプの junction では、 -v オプションは必須です。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。

表 40. ローカル・タイプの仮想ホスト *junction* のオプション (続き)

オプション	説明
<code>-z replica-set-name</code>	オプション。仮想ホスト <i>junction</i> のセッションを管理するレプリカ・セットを指定し、複数の仮想ホスト間でログイン・セッションをグループ化または分離する機能を提供します。 仮想ホスト <i>junction</i> のレプリカ・セットを指定する -z が使用されない場合、仮想ホスト <i>junction</i> は自動的に、仮想ホスト名に一致するレプリカ・セットに割り当てられます。例えば、仮想ホスト名が vhostA.example.com である場合、レプリカ・セットは vhostA.example.com という名前になります。仮想ホスト <i>junction</i> に使用されるレプリカ・セットは、WebSEAL 構成ファイルの [replica-sets] スタンザ内に存在しなければなりません。 363 ページの『第 20 章 SMS を使用した WebSEAL の構成』を参照してください。
<code>-f</code>	既存の <i>junction</i> を強制的に置き換えます。 453 ページの『新規 <i>junction</i> の強制適用』を参照してください。
<code>-l percent-value</code>	ワーカー・スレッドの消費量のソフト限界を定義します。 59 ページの『<i>junction</i> 用のワーカー・スレッドの割り振り (Junction フェアネス)』を参照してください。
<code>-L percent-value</code>	ワーカー・スレッドの消費量のハード限界を定義します。 59 ページの『<i>junction</i> 用のワーカー・スレッドの割り振り (Junction フェアネス)』を参照してください。

ローカル仮想ホスト *junction* の例:

最初のコマンド (1 行で入力します) により、**p.s.com** という **Host** ヘッダー値を持つ HTTP 要求に応答するローカル仮想 *junction* が作成されます。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t localtcp -v p.s.com vhost-local-ps-http
```

2 番目のコマンド (1 行で入力します) により、**p.s.com:444** という **Host** ヘッダー値を持つ HTTPS 要求に応答するローカル仮想 *junction* が作成されます。 **-g** オプションは、この SSL *junction* と最初の TCP *junction* を組みにし、それらが同じ保護オブジェクト・スペースを共用できるようにします。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t localssl -v p.s.com:444 -g vhost-local-ps-http vhost-local-ps-https
```

シナリオ 1: リモート仮想ホスト junction

以下のシナリオでは、単一のバックエンド・サーバーにある 2 つのリモート仮想ホストに対して junction サポートがセットアップされています。ステップを進むごとに付随する図を参照してください。

必須アーキテクチャー:

- デフォルトでは、WebSEAL の構成ファイルは、すべての IP アドレスをサポートするように設定されています。

```
[server]
network-interface = 0.0.0.0
```

この仮想ホスト・シナリオでは、WebSEAL (**webseal.ibm.com**) は、特定のネットワーク・アドレスを使用するように構成されています。

```
[server]
network-interface = 9.0.0.3
```

- このセットアップでは、WebSEAL サーバーは、1 つの junction バックエンド・サーバーで 2 つの仮想ホストを保護しています。
 - 仮想ホスト **a.b.com** (サーバー **cruz1.ibm.com** 上)
 - 仮想ホスト **x.y.com** (サーバー **cruz1.ibm.com** 上)
- junction 保護サーバー (**cruz1.ibm.com**) への直接アクセスは、適切なファイアウォール保護により防止されています。ブラウザが仮想ホスト名の検索に使用する外部 DNS エントリーが、IP アドレス 9.0.0.3 にある WebSEAL をポイントするように構成されているため、ユーザーはこのブロックされたアクセスに気付きません。

外部 DNS	
a.b.com	9.0.0.3
x.y.com	9.0.0.3

- 仮想ホスト **a.b.com** は、HTTP アクセスのみを受け入れます。
- 仮想ホスト **x.y.com** は、セキュア HTTPS アクセスを受け入れます。

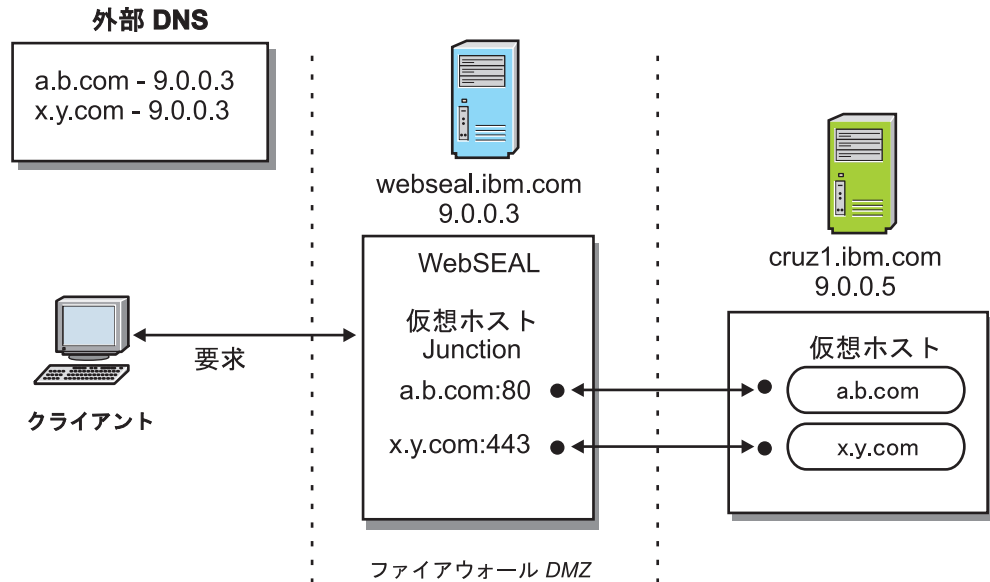


図 41. 仮想ホスト junction シナリオ 1

手順:

1. 以下の **pdadmin** コマンド (1 行で入力します) により、WebSEAL への TCP (HTTP) 要求にある **Host: a.b.com** ヘッダーに応答する **vhost-ab-http** と名付けられた (ラベルを付けられた) 仮想ホスト Junction が作成されます。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t tcp -h cruz1.ibm.com -v a.b.com vhost-ab-http
```

2. 以下のコマンド (1 行で入力します) により、WebSEAL への SSL (HTTPS) 要求にある **Host: x.y.com** ヘッダーに応答する **vhost-xy-https** と名付けられた (ラベルを付けられた) 仮想ホスト Junction が作成されます。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t ssl -h cruz1.ibm.com -v x.y.com vhost-xy-https
```

3. クライアント・ユーザーにより、HTML ページ上にある次の (例) リンクがクリックされます。

<http://a.b.com/doc/readme.txt>

このリソースへの要求 (例) は、次のように表示されます。

```
GET /doc/readme.txt HTTP/1.1
Host: a.b.com
User-Agent: Mozilla 4.0 (X; I; Linux-2.0.35i586)
Accept: image/gif, image/jpeg, */*
```

DNS により、要求されたサーバー (**a.b.com**) への通信を WebSEAL ホスト (9.0.0.3) に送付することが決定されます。

WebSEAL は、**Host** ヘッダーを検出し、バックエンド・サーバー **cruz1.ibm.com** 上にある仮想ホスト **a.b.com** に対する junction を介してその要求を送付します。

仮想ホスト junction のインターフェースの定義

WebSEAL は、複数のインターフェースを listen するように構成することができます。複数インターフェース機能は、複数の仮想ホスト junction に対する証明書サポート (SSL) の設定時に重要となります。デジタル証明書には、アクセス中のホストの名前が含まれています。そのため、SSL 用に構成された仮想ホストごとに固有の証明書を交換する必要があります。証明書とホスト間で名前の不一致があると、ブラウザから警告メッセージが生成されます。

このセクションは、以下のトピックで構成されています。

- 『デフォルトのインターフェース仕様』
- 『追加インターフェースの定義』

デフォルトのインターフェース仕様

ネットワーク・インターフェースとは、HTTP や HTTPS ポート設定、IP アドレス、ワーカー・スレッド設定、および証明書処理設定といった特定のグループの設定に対する値の組み合わせと定義されています。

WebSEAL インスタンスに対する単一のデフォルト・インターフェースは、WebSEAL 構成ファイルにある、以下のスタンザ・エントリーの値で定義されます。

```
[server]
http
http-port
https
https-port
worker-threads
network-interface

[ssl]
webseal-cert-keyfile-label

[certificate]
accept-client-certs
```

追加インターフェースの定義

このタスクについて

追加インターフェースを構成するには、名前をカスタマイズした各インターフェースを WebSEAL 構成ファイルの **[interfaces]** スタンザ内に定義します。

それぞれのインターフェース定義には、プロパティのリストを含めます。ほとんどのプロパティでは、WebSEAL 構成ファイルにある、デフォルトのインターフェース仕様の一部である対応するスタンザ・エントリー名を模倣します (『デフォルトのインターフェース仕様』を参照してください)。

カスタム・インターフェース指定は、次の形式が使用されます。

```
[interfaces]
interface-name = property=value[;property=value[;...]]
```

以下の表に、カスタム・インターフェースの構成に使用可能なプロパティと値をリストします。

表 41. 追加インターフェース定義に有効なプロパティおよび値

プロパティ	値	説明
http-port	- ポート番号 - disabled (デフォルト)	指定された network-interface 上の HTTP 要求を listen するためのポート番号。値を disabled に設定することもできます。 インターフェースを定義する際には、http-port または https-port のいずれかを指定する必要があります。
https-port	- ポート番号 - disabled (デフォルト)	指定された network-interface 上の HTTPS 要求を listen するためのポート番号。値を disabled に設定することもできます。 インターフェースを定義する際には、http-port または https-port のいずれかを指定する必要があります。
worker-threads	<i>count</i> - default (デフォルト)	このインターフェース上でのみ受信した要求を処理するのに使用されるワーカー・スレッドの数。 default 値は、デフォルト・インターフェースに属するワーカー・スレッド・プールの使用の指定に使用できます (558 ページの『デフォルトのインターフェース仕様』を参照してください)。
network-interface	<i>ip-address</i> 0.0.0.0 (デフォルト)	指定された http-port または https-port 上の要求を listen するための IP アドレス。 IPv4 と IPv6 の両方の形式がサポートされています。
certificate-label	<i>key-file-label</i>	pdsrv.kdb 鍵データベース・ファイル内の証明書のラベル名。 https-port が指定されている場合にのみ有効です。 これは、WebSEAL がクライアントを認証するのに使用するサーバー・サイド証明書です。
accept-client-certs	- never (デフォルト) - required - optional - prompt_as_needed	WebSEAL によるクライアント・サイド証明書の処理方法を指定します。 https-port が指定されている場合にのみ有効です。 164 ページの『クライアント・サイド証明書認証』を参照してください。

プロパティ値の構文ルール:

- セミコロン (;)、二重引用符 (")、または円記号 (¥) を含む値では、円記号 (¥) を前に置く必要があります。
- 先頭または末尾にスペースを含む値を指定するには、二重引用符 (") を使用する必要があります。
- セミコロン (;) が二重引用符で囲まれた値の中にある場合、円記号 (¥) を前に置く必要はありません。

例

```
[interfaces]
support = network-interface=9.0.0.8;https-port=444;certificate-label=WS6;
worker-threads=16
```

この例 (1 行で入力します) では、以下のプロパティを持つ「support」と名付けられたインターフェースが作成されます。

- WebSEAL が、HTTPS ポート 444 上の IP アドレス 9.0.0.8 にある要求を listen できるようにします。
- HTTP ポートのデフォルトは「使用不可」です。
- WebSEAL は、WebSEAL データベース・ファイルに保管された「WS6」と名付けられたサーバー・サイド証明書を使用して SSL クライアントを認証します。
- このインターフェースでは、サービス要求に対して 16 個の独自のワーカー・スレッド・プールを使用します。
- デフォルトでは、このインターフェースは認証にクライアント・サイド証明書を必要としません (プロンプトを出しません)。

注: インターフェースに割り当てられた複数の DNS 別名を持つように WebSEAL を構成する場合、それぞれの別名 (およびインターフェース) が仮想ホスト junction に割り当てられるようにする必要があります。WebSEAL は、仮想ホスト junction に割り当てられていないインターフェース定義に割り当てられた複数の DNS 別名をサポートしません。

シナリオ 2: インターフェースを持つ仮想ホスト junction

以下のシナリオでは、次のものをセットアップします。

- 仮想ホスト junction は、異なる WebSEAL インターフェース上に作成されます。
- 2 つの junction は、共通の保護オブジェクト・スペースを共有するように構成されています。

ステップを進むごとに付随する図を参照してください。

必須アーキテクチャー:

- WebSEAL では、以下のプロトコルで 3 つの仮想ホストを保護しています。
 - HTTP および HTTPS を介した **a.b.com** (ホスト **cruz1.ibm.com** 上)
 - HTTP を介した **w.x.com** (ホスト **cruz2.ibm.com** 上)
 - HTTPS を介した **y.z.com** (ホスト **cruz2.ibm.com** 上)
- junction 保護サーバー (**cruz1.ibm.com** > **cruz2.ibm.com**) への直接アクセスは、適切なファイアウォール保護により防止されています。ブラウザーが仮想ホスト名

の検索に使用する外部 DNS エントリーが、IP アドレス 9.0.0.3 または 9.0.0.4 にある WebSEAL をポイントするように構成されているため、ユーザーはこのブロックされたアクセスに気付きません。

- 仮想ホストは、外部 DNS 内で、WebSEAL サーバーをポイントするように構成されています。

外部 DNS	
a.b.com	9.0.0.3
x.y.com	9.0.0.3
y.z.com	9.0.0.4

- WebSEAL サーバーは、以下のホスト名でブラウザに認識されています。
 - **webseal.ibm.com** (WebSEAL の真のホスト名)
 - **a.b.com**
 - **w.x.com**
 - **y.z.com**
- WebSEAL は、2 つのインターフェース用に構成されています (**a.b.com** > **y.z.com** に対して HTTPS を介した固有のサーバー・サイド証明書のサービス提供を許可するため)。
 - 9.0.0.3
 - 9.0.0.4

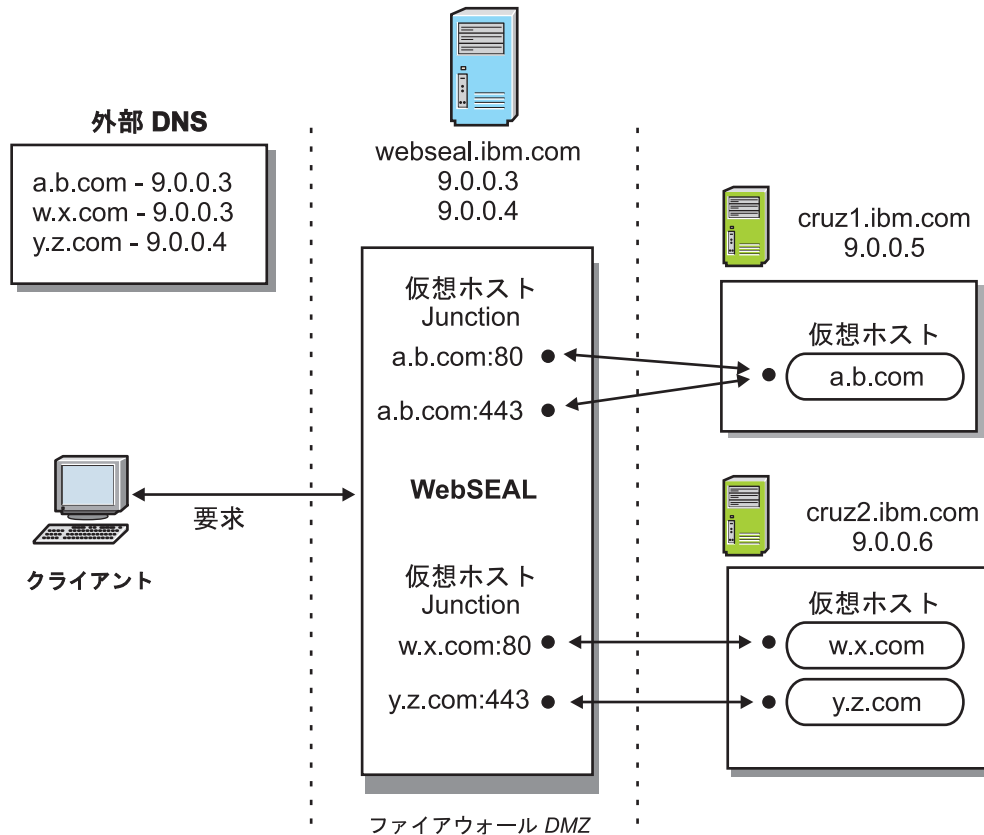


図 42. 仮想ホスト junction シナリオ 2

手順 - 一般的なセットアップ:

1. 2 つの必須インターフェースのうち最初のものを使用してデフォルトの WebSEAL をインストールし構成します (**a.b.com:443** との SSL 通信をサポートするため)。

```
[server]
network-interface = 9.0.0.3
http = yes
http-port = 80
https = yes
https-port = 443
```

2. ブラウザーと **a.b.com** 仮想ホスト間の SSL 通信 (ポート 443 を使用) をサポートするために、サーバー・サイド証明書 (この例では、**ab** と名付けられている) を WebSEAL の pdsrv.kdb 鍵ファイル・データベースにインストールします。この証明書は、認証局 (CA) により生成され署名される必要があります。WebSEAL は、インターフェースの代わりにこの証明書を提示してクライアント・ブラウザを認証します。

```
[ssl]
webseal-cert-keyfile-label = ab
```

注: WebSEAL は、**[ssl]** スタンザで指定されたクライアント証明書に使用されているものを共有するのではなく、junction SSL 操作に別個の証明書鍵データベースを構成するオプションを提供します。詳しくは、405 ページの『WebSEAL 鍵データベース・ファイルの構成』、および「*IBM Security Web Gateway*

Appliance: Web リバース・プロキシ・スタンザ・リファレンス」内の **jct-cert-keyfile** オプションの説明を参照してください。

3. **y.z.com:443** との SSL 通信をサポートするように 2 番目のインターフェースを構成します。

```
[interfaces]
yz-interface = network-interface=9.0.0.4; certificate-label=yz; https-port=443
```

4. ブラウザーと **y.z.com** 仮想ホスト 間の SSL 通信 (ポート 443 を使用) をサポートするために、サーバー・サイド証明書 (この例では、**yz** と名付けられている) を WebSEAL の pdsrv.kdb 鍵ファイル・データベースにインストールします。この証明書は、認証局 (CA) により生成され署名される必要があります。WebSEAL は、インターフェースの代わりにこの証明書を提示してクライアント・ブラウザを認証します。

注: WebSEAL は、**[ssl]** スタンザで指定されたクライアント証明書に使用されているものを共有するのではなく、junction SSL 操作に別個の証明書鍵データベースを構成するオプションを提供します。詳しくは、405 ページの『WebSEAL 鍵データベース・ファイルの構成』、および「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」内の **jct-cert-keyfile** オプションの説明を参照してください。

5. (例えば、標準 WebSEAL junction によって) 要求された場合に、1 次 WebSEAL ホスト名が確実に使用されるようにするために、WebSEAL 構成ファイルの **web-host-name** スタンザ・エントリーに、適切な名前を値として割り当てます。

```
[server]
server-name = webseal.ibm.com-default
web-host-name = webseal.ibm.com
```

手順 - 仮想ホスト Junction の作成:

1. **a.b.com** への HTTP および HTTPS 通信をサポートするため、2 つの仮想ホスト junction (1 行で入力します) を作成します。**-g** オプションを使用して、2 つの junction が同じオブジェクト・スペースを共用できるようにします。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t tcp -h cruz1.ibm.com -v a.b.com vhost-ab-tcp
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t ssl -h cruz1.ibm.com -v a.b.com -g vhost-ab-tcp vhost-ab-ssl
```

2. **w.x.com:80** との通信をサポートするため、仮想ホスト junction (1 行で入力します) を作成します。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t tcp -h cruz2.ibm.com -v w.x.com vhost-wx-tcp
```

3. **y.z.com:443** との通信をサポートするため、仮想ホスト junction (1 行で入力します) を作成します。

```
pdadmin> server task default-webseald-webseal.ibm.com virtualhost create
-t ssl -h cruz2.ibm.com -v y.z.com vhost-yz-ssl
```

既存の WebSEAL 機能での仮想ホストの使用

このセクションは、以下のトピックで構成されています。

- 『仮想ホストを使用した e-community シングル・サインオン』
- 565 ページの『仮想ホストを使用したクロスドメイン・シングル・サインオン』
- 566 ページの『仮想ホスト junction を使用した動的 URL』
- 566 ページの『仮想ホスト・シングル・サインオンのためのドメイン・セッション Cookie の使用』
- 568 ページの『Junction スロットル』

仮想ホストを使用した e-community シングル・サインオン

e-community シングル・サインオンを使用して、単一の WebSEAL インスタンスにある複数の仮想ホスト間でシングル・サインオンを実行することができます。仮想ホストはすべて、その WebSEAL インスタンスに属する同一の構成ファイルを共有します。

仮想ホスト junction をサポートするため、e-community シングル・サインオンでは以下のことを許可しています。

- 単一の仮想ホストが、複数の仮想ホストを持つマシン上でマスター認証サーバー (MAS) として動作できます。
- 異なるドメインをサポートする複数の仮想ホストを持つ環境に対して、ドメインごとのシングル・サインオン鍵を指定できます。

e-community に対する構成の強化には次のものがあります。

- 複数の仮想ホストを持つマシン上で、**master-authn-server** スタンザ・エントリーを使用し、いずれかの仮想ホストを MAS に指定します。

```
[e-community-sso]
is-master-authn-server = yes
master-authn-server = virtual-host
```

- **[e-community-domain-keys]** スタンザは、標準 WebSEAL junction での使用にのみ適しています。

```
[e-community-domain-keys]
```

- **[e-community-domains]** スタンザには、仮想ホストがサポートするドメインがリストされます。

```
[e-community-domains]
```

- **[e-community-domain-keys:domain]** スタンザには、**[e-community-domains]** スタンザで定義されたそれぞれのドメインに適切な鍵が記載されます。

```
[e-community-domain-keys:domain]
```

例:

- 1 つの WebSEAL インスタンス
- e-community を形成する 3 つの仮想ホスト (4 つの仮想ホスト junction):

```
www.ibm.com:80
www.ibm.com:443
www.lotus.com:80
www.tivoli.com:80
```

注: `www.ibm.com:80` および `www.ibm.com:443` は、仮想ホスト `junction` のプロトコル・ペアです。これらは `-g` 仮想ホスト `junction` オプションで作成されているため、同じオブジェクト・スペースを共有します。これらのサーバーは、単一の仮想ホスト `www.ibm.com` です。

- MAS は、仮想ホスト `www.ibm.com` です。
- WebSEAL 構成ファイル:

```
[e-community-sso]
is-master-authn-server = yes
master-authn-server = www.ibm.com

[e-community-domains]
name = ibm.com
name = tivoli.com
name = lotus.com

[e-community-sso-domain-keys:ibm.com]
ibm.com = ibm.key
tivoli.com = ibm-tivoli.key
lotus.com = ibm-lotus.key

[e-community-sso-domain-keys:tivoli.com]
tivoli.com = tivoli.key
ibm.com = ibm-tivoli.key

[e-community-sso-domain-keys:lotus.com]
lotus.com = lotus.key
ibm.com = ibm-lotus.key
```

仮想ホストを使用したクロスドメイン・シングル・サインオン

クロスドメイン・シングル・サインオンを使用して、単一の WebSEAL インスタンスにある複数の仮想ホスト間でシングル・サインオンを実行することができます。仮想ホストはすべて、その WebSEAL インスタンスに属する同一の構成ファイルを共有します。ただし、仮想ホストを使用したクロスドメイン・シングル・サインオンには、特定の構成上の制限があります。

仮想ホストを使用した `e-community` シングル・サインオンとは異なり、クロスドメイン・シングル・サインオン構成では、異なるドメインをサポートする複数の仮想ホストを持つ環境に対して、ドメインごとのシングル・サインオン鍵を指定できません。WebSEAL インスタンスに関連付けられたすべての仮想ホストは、鍵の構成に使用される 1 つの `[cdsso-peers]` スタンザを共有する必要があります。そのため仮想ホストは、各ドメインが使用する共通の鍵を共有して、別の指定されたドメインと通信する必要があります。

以下の例では、2 つの仮想ホストが単一の WebSEAL インスタンス上にあります。

- `a.a.com`
- `b.b.com`

どちらのドメインも別個のエンティティに所有されており、これらのエンティティのそれぞれに、もう 1 つの WebSEAL サーバー `c.c.com` との別個の CDSO 配置があります。理想的なのは、`a.a.com` および `b.b.com` が、`c.c.com` とのシングル・サインオンに使用する別個の鍵を持つようにすることです。理想的な構成は次のようになります。

```
[cdsso-peers]
# used by a.a.com to communicate with c.c.com
c.c.com = c-a.key
# used by b.b.com to communicate with c.c.com
c.c.com = c-b.key
```

[cdsso-peers] スタンザでは、指定されたターゲット・ドメインに 1 つの鍵のみを指定可能であるため、この構成は単一の WebSEAL インスタンス (仮想ホスト a.a.com および b.b.com の両方をホストする) では不可能です。

許可される唯一の構成では、a.a.com および b.b.com の両方が同一の鍵を使用するように強制します。例:

```
[cdsso-peers]
# used by both a.a.com and b.b.com to communicate with c.c.com
c.c.com = c-ab.key
```

a.a.com および b.b.com ドメインのそれぞれの所有者が、同一の鍵を共用する条件を受け入れる必要があります。

さらに、その 1 つの鍵について暗号漏えいがあった場合、a.a.com および b.b.com の両方が信頼できなくなります。

仮想ホスト junction を使用した動的 URL

dynurl.conf 構成ファイルでは、仮想ホスト junction は次のように表現されます。

```
/@vhost-label
```

例えば、以下の dynurl.conf 構成ファイル・エントリーでは、**test-http** とラベル付けされた仮想ホスト Junction が /mapfrom ディレクトリー下のすべての URL を保護オブジェクト /@test-http/mapto にマップするように指定されています。

```
/@test-http/mapto/@test-http/mapfrom/*
```

標準 WebSEAL junction および仮想ホスト junction は、dynurl.conf 構成ファイル内で共存できます。例:

```
/mapto/@test-http/mapfrom/*
/@test-http/mapto/mapfrom/*
```

仮想ホスト・シングル・サインオンのためのドメイン・セッション Cookie の使用

このタスクについて

WebSEAL 環境では、ドメイン Cookie を使用してシングル・サインオンをサポートできるほか、同じ WebSEAL インスタンスの複数の仮想ホスト junction にわたる単一の資格情報の共有をサポートすることができます。あるいは、複数の仮想ホストにわたるシングル・サインオンをサポートするように SMS 環境を構成することもできます。SMS 環境では、異なる WebSEAL インスタンスの間にセッションを分散できます。

WebSEAL は通常、Cookie ベースのセッションの識別にホスト Cookie を使用します。ブラウザは、ホスト Cookie を発信元ホストに戻すだけです。仮想ホスト環境でホスト Cookie を使用すると、仮想ホストは独自のログインと資格情報を持つようになります。

ドメイン Cookie は、すべての仮想ホストが同じ WebSEAL インスタンス上にあり、同じネットワーク・ドメイン名を含む場合に使用します。

SMS なしの環境では、WebSEAL の追加の構成項目を設定して、同じ WebSEAL インスタンスでの複数の仮想ホスト junction にわたるシングル・サインオンを処理する必要があります。WebSEAL 構成ファイルの **[session]** スタンザの **shared-domain-cookie** 構成項目は **yes** に設定する必要があります。SMS 環境ではこの構成項目を使用する必要はありません。SMS 環境では、この項目を **no** に設定するか、一切定義しないでください。

標準 WebSEAL junction および仮想ホスト junction は両者ともドメイン Cookie をサポートできます。

- 特定の仮想ホスト junction に対するセッション Cookie で使用されるドメインは、**[session-cookie-domains]** スタンザのエントリーに最も一致したものとなります。
- 特定の標準 WebSEAL junction に対するセッション Cookie で使用されるドメインは、**[server]** スタンザ内の **web-host-name** エントリーの値に最も一致したものとなります。

一致するものがない場合は、ホスト・タイプの Cookie が使用されます。

同一ドメインにあるその他の WebSEAL インスタンスも、特定の WebSEAL インスタンスに対して構成されたものと同じドメイン Cookie を受信します。ユーザーは、特定の WebSEAL インスタンスに対する WebSEAL セッション Cookie の名前をカスタマイズできます。WebSEAL インスタンスの構成ファイルでは、TCP および SSL の両方の Cookie に対するデフォルトの名前が提供されます。

```
[session]
tcp-session-cookie-name = PD-H-SESSION-ID
ssl-session-cookie-name = PD-S-SESSION-ID
```

332 ページの『セッション Cookie 名のカスタマイズ』も参照してください。

手順

ドメイン Cookie を使用可能にするには、WebSEAL 構成ファイルを変更して、ドメイン・タイプの Cookie の使用に適したドメインの名前を指定してください。例:

```
[session-cookie-domains]
domain = ibm.com
domain = cruz.tivoli.com
```

例

対応する例:

```
[session-cookie-domains]
domain = ibm.com
domain = tivoli.com
```


仮想ホストでドメイン Cookie を使用するときの技術上の注意

- **セキュリティ警告** 特定のドメインに対するホストのコレクションに、信頼されていないホストが存在することは可能です。
- ドメイン Cookie を使用するには、すべての仮想ホストが同じ DNS ドメイン内にある必要があります。
- WebSEAL を単独で使用して、同じ WebSEAL インスタンス内の仮想ホスト Junction を介してシングル・サインオンを実行することができます。あるいは、複数の WebSEAL インスタンスに分散できる、仮想ホスト Junction を介してシングル・サインオンを実行するように SMS 環境を構成することもできます。
- 仮想ホスト Junction を介してシングル・サインオンを実現するために SMS を使用する場合は、WebSEAL の [session] スタンザの shared-domain-cookie 構成項目を使用可能にしないでください。
- 381 ページの『セッション・レルム内でのシングル・サインオン』も参照してください。

Junction スロットル

junction スロットルについて詳しくは、455 ページの『Junction スロットル』を参照してください。

シナリオ 3: 仮想ホストの拡張構成

以下のシナリオは、シナリオ 2 に基づいています。このシナリオでは、以下のセットアップが追加されます。

- フォーム認証
- e-community を使用したシングル・サインオン
- 固有のリソースへの認証アクセス

必須アーキテクチャー:

- すべてのホストとプロトコル間でシングル・サインオンが必要です。各仮想ホストには、独自のセッション資格情報があります。標準 e-community SSO ログアウト制限が適用されます。
- フォーム・ログインが必要です
- 以下のリソースに対するアクセスは、認証済みのみである必要があります。
 - http(s)://a.b.com/sales/
 - https://w.x.com/doc/
 - https://y.z.com/code/

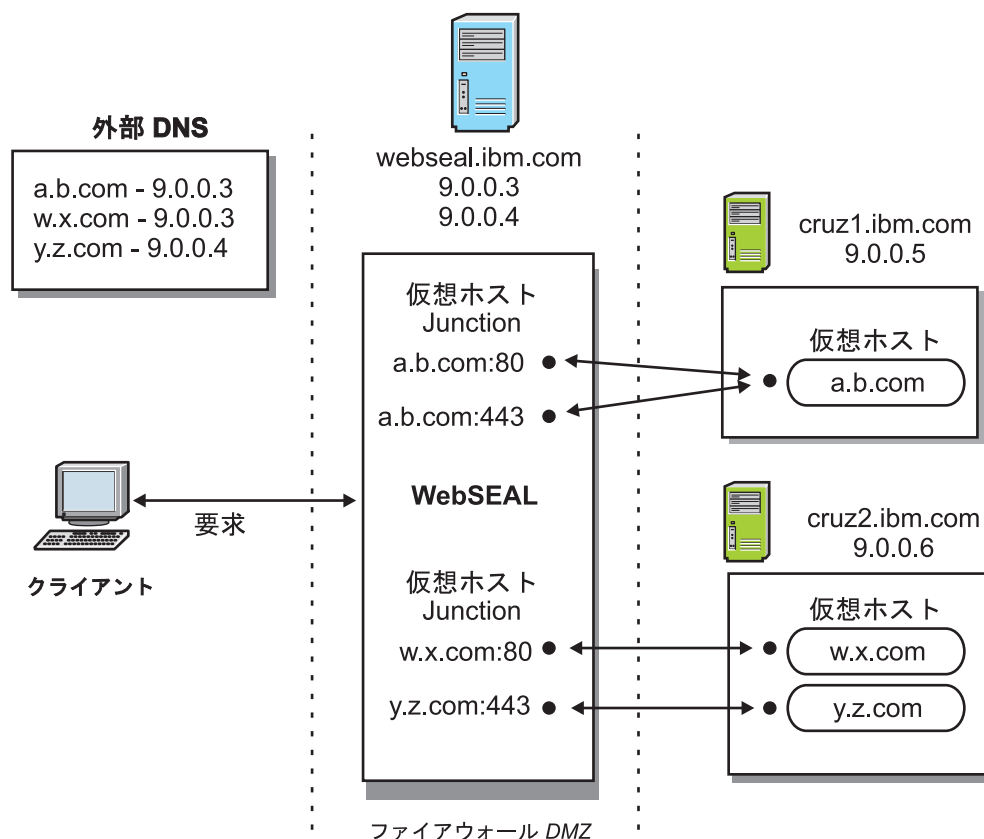


図 43. 仮想ホスト junction シナリオ 3

手順 - 一般的なセットアップ:

1. HTTP および HTTPS の両方のフォーム認証を使用可能にします。

```
[ba]
ba-auth = none
```

```
[forms]
forms-auth = both
```

2. e-community シングル・サインオンを構成します。

```
[e-community-sso]
e-community-sso-auth = both
e-community-name = ecomm
is-master-authn-server = yes
master-authn-server = a.b.com
```

```
[e-community-domain-keys]
```

```
[e-community-domains]
name = b.com
name = x.com
name = z.com
name = ibm.com
```

```
[e-community-domain-keys:b.com]
b.com = bb.key
x.com = bx.key
z.com = bz.key
```

```
[e-community-domain-keys:x.com]
x.com = xx.key
```

```
b.com = bx.key
```

```
[e-community-domain-keys:z.com]  
z.com = zz.key  
b.com = bz.key
```

3. LMI を使用して、以下の鍵を生成します。
 - bb.key
 - bx.key
 - bz.key
 - xx.key
 - zz.key
4. WebSEAL を再始動し、**sec_master** として **pdadmin** コマンドにログインします。

手順 - /restricted ディレクトリーへのアクセスの制御:

1. 一般的な非認証アクセスに対して、**open** (制限されていない) ACL を作成します。

```
pdadmin> sec_master> acl create open  
pdadmin> sec_master> acl modify open set user sec_master TcmdbsvaBRlrx  
pdadmin> sec_master> acl modify open set any-other Trx  
pdadmin> sec_master> acl modify open set unauthenticated Trx  
pdadmin> sec_master> acl modify open set group iv-admin TcmdbsvaBRxl  
pdadmin> sec_master> acl modify open set group webseal-servers Tgmdbsrxl
```

2. 認証を必要とするアクセスに対して、**restricted** ACL を作成します。

```
pdadmin> sec_master> acl create restricted  
pdadmin> sec_master> acl modify restricted set group iv-admin TcmdbsvaBRxl  
pdadmin> sec_master> acl modify restricted set group webseal-servers Tgmdbsrxl  
pdadmin> sec_master> acl modify restricted set user sec_master TcmdbsvaBRlrx  
pdadmin> sec_master> acl modify restricted set any-other Trx  
pdadmin> sec_master> acl modify restricted set unauthenticated T
```

3. **open** ACL をデフォルトの WebSEAL インスタンスに付加します。

```
pdadmin> sec_master> acl attach /WebSEAL/webseal.ibm.com-default open
```

4. **restricted** ACL を **a.b.com** 上の /sales ディレクトリーに付加します (1 行で入力します)。

```
pdadmin sec_master> acl attach  
/WebSEAL/webseal.ibm.com-default/@vhost-ab-tcp/sales restricted
```

5. **restricted** ACL を **w.x.com** 上の /doc ディレクトリーに付加します (1 行で入力します)。

```
pdadmin sec_master> acl attach  
WebSEAL/webseal.ibm.com-default/@vhost-wx-tcp/doc restricted
```

6. **restricted** ACL を **y.z.com** 上の /code ディレクトリーに付加します (1 行で入力します)。

```
pdadmin sec_master> acl attach  
/WebSEAL/webseal.ibm.com-default/@vhost-yz-ssl/code restricted
```

仮想ホスト junction の制限

このセクションは、以下のトピックで構成されています。

- 571 ページの『仮想ホストでの SSL セッション ID は使用不可』

仮想ホストでの SSL セッション ID は使用不可

SSL セッション IDは、ブラウザーと、同じ WebSEAL インスタンス上にあるさまざまな仮想ホスト間のログイン・セッションを維持するために使用することはできません。

第 30 章 コマンド・オプションの要約: 仮想ホスト junction

pdadmin ユーティリティには対話式コマンド行プロンプトが用意されていて、ユーザーは、これを使用して、WebSEAL 仮想ホスト junction タスクを実行できます。このセクションでは、WebSEAL 仮想ホスト junction を作成するための **pdadmin server task virtualhost** コマンドについて説明します。**pdadmin** ユーティリティの完全な構文の説明については、「*IBM Security Access Manager for Web: Command Reference*」を参照してください。標準 WebSEAL junction のオプションについては、531 ページの『第 28 章 コマンド・オプションの要約: 標準 junction』で説明します。

注: また、Security Access Manager Web Portal Manager を使用して仮想ホスト junction を管理できます。詳しくは、Web Portal Manager のオンライン・ヘルプ画面を参照してください。

この章で扱うトピックは以下のとおりです。

- 『pdadmin サーバー・タスクによる仮想ホスト junction の作成』
- 574 ページの『仮想ホスト junction のサーバー・タスク・コマンド』
- 576 ページの『仮想ホスト junction の作成』
- 582 ページの『仮想ホスト junction へのサーバーの追加』

pdadmin サーバー・タスクによる仮想ホスト junction の作成

始める前に

pdadmin を使用する前に、**sec_master** などの管理許可を持つユーザーとしてセキュア・ドメインにログインする必要があります。

例:

```
pdadmin> login
Enter User ID: sec_master
Enter Password:
pdadmin>
```

手順

仮想ホスト junction を作成するには、**pdadmin server task virtualhost create** コマンドを使用します (1 行で入力します)。

```
pdadmin> server task instance_name-webseald-host_name virtualhost create
options vhost-label
```

例えば、単一の WebSEAL インスタンスの構成済みの名前が **web1** であり、**www.pubs.com** と名付けられたホストにインストールされている場合、完全なサーバー名は次のようになります。

```
web1-webseald-www.pubs.com
```

完全なサーバー名を正しい形式で表示するには、**pdadmin server list** コマンドを使用します。

```
pdadmin> server list
web1-webseald-www.pubs.com
```

詳しくは、707 ページの『付録 B. コマンド・リファレンス』または「*IBM Security Access Manager for Web: Command Reference*」の **pdadmin server task virtualhost create** の参照ページを参照してください。

仮想ホスト junction のサーバー・タスク・コマンド

pdadmin server task virtualhost では以下の仮想ホスト junction コマンドが使用できます。

コマンド	説明
virtualhost add	既存の仮想ホスト junction に新しいサーバーを追加します。 構文: <pre>virtualhost add options vhost-label</pre> 582 ページの『仮想ホスト junction へのサーバーの追加』を参照してください。
virtualhost create	初期サーバー用の新規仮想ホスト junction を作成します。 構文: <pre>virtualhost create options vhost-label</pre> 576 ページの『仮想ホスト junction の作成』を参照してください。
virtualhost delete	指定した仮想ホスト junction を除去します。 仮想ホスト junction を、別の仮想ホスト junction が -g オプションを使用して参照している場合、参照されている仮想ホスト junction は削除できません。このような操作を試行すると、エラー・メッセージが戻されます。 構文: <pre>virtualhost delete vhost-label</pre>
virtualhost list	すべての構成済み仮想ホスト junction をラベル名でリストします。 構文: <pre>virtualhost list</pre>

コマンド	説明
virtualhost offline	この仮想ホスト junction にあるサーバーの操作状態をオフラインにします。指定されたサーバーに追加で送信される要求はありません。サーバーを指定しないと、この仮想ホスト junction にあるすべてのサーバーの操作状態がオフラインになります。 構文: <pre>virtualhost offline [-i server_uuid] vhost_label</pre>
virtualhost online	この仮想ホスト junction にあるサーバーの操作状態をオンラインにします。サーバーは通常操作を再開します。サーバーを指定しないと、この仮想ホスト junction にあるすべてのサーバーの操作状態がオンラインになります。 構文: <pre>virtualhost online [-i server_uuid] vhost_label</pre>
virtualhost remove	仮想ホスト junction から指定したサーバーを除去します。 構文: <pre>virtualhost remove -i server-id vhost-label</pre> 特定のサーバーの ID を判別するには、virtualhost show コマンドを使用します。
virtualhost show	指定されたラベルでの仮想ホスト junction の詳細を表示します。 構文: <pre>virtualhost show vhost-label</pre>
virtualhost throttle	この仮想ホスト junction にあるサーバーの操作状態をスロットルにします。指定されたサーバーが継続して処理するのは、このコマンドを呼び出す前に WebSEAL とのセッションを作成したユーザーからの要求のみです。サーバーを指定しないと、この仮想ホスト junction にあるすべてのサーバーの操作状態がスロットルになります。 構文: <pre>virtualhost throttle [-i server_uuid] vhost_label</pre>

仮想ホスト junction の作成

操作: 新規仮想ホスト junction を作成します。

構文:

```
virtualhost create -t type -h host-name options vhost-label
```

junction タイプ	
-t <i>type</i>	仮想ホスト junction のタイプ。 tcp 、 ssl 、 tcpproxy 、 sslproxy 、 localtcp 、 localssl のいずれかです。 -t tcp のデフォルトのポートは 80 です。-t ssl のデフォルトのポートは 443 です。 **必須**。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。
ホスト名	
-h <i>host-name</i>	ターゲット・バックエンド・サーバーの DNS ホスト名または IP アドレス。 **必須** (tcp 、 ssl 、 tcpproxy 、 sslproxy タイプの junction の場合)。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。
仮想ホスト・オプション	
-v <i>vhost-name[:port]</i>	要求の HTTP Host ヘッダーが、-v オプションで指定されている仮想ホスト名およびポート番号に一致する場合、WebSEAL は要求を処理するために仮想ホスト Junction を選択します。 -v オプションは、バックエンド・サーバーに送信される要求の Host ヘッダーの値を指定するためにも使用されます。 仮想ホストがプロトコルの標準外のポートを使用する場合、ポート番号が必要です。TCP の標準ポートは 80、SSL の標準ポートは 443 です。 tcp 、 ssl 、 tcpproxy 、および sslproxy タイプの junction に -v が指定されていない場合は、-h <i>host</i> および -p <i>port</i> オプションに含まれている情報から junction が選択されます。 localtcp および localssl タイプの junction では、-v オプションは必須です。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。

<code>-g vhost-label</code>	-g オプションを指定すると、2 番目の追加仮想ホスト <code>junction</code> が最初の仮想ホスト <code>junction</code> と同じ保護オブジェクト・スペースを共有します。 このオプションは、Junction ペア (補完プロトコルを使用する 2 つの Junction) にのみ該当します。このオプションでは、2 つを超える Junction の関連付けはサポートしていません。 オプション。552 ページの『リモート・タイプ仮想ホスト <code>junction</code> の作成』を参照してください。
一般オプション	
TCP および SSL <code>junction</code> タイプ	
<code>-a address</code>	ターゲット・バックエンド・サーバーとの通信時に WebSEAL が使用するローカル IP アドレスを指定します。このオプションを指定しない場合、WebSEAL ではオペレーティング・システムにより指定されるデフォルトのアドレスを使用します。 特定の <code>junction</code> のアドレスを指定すると、WebSEAL は、その <code>junction</code> サーバーとのすべての通信の場合にこのローカル・アドレスにバインドします。
<code>-f</code>	既存の仮想ホスト <code>junction</code> を強制的に置換 (上書き) します。 453 ページの『新規 <code>junction</code> の強制適用』を参照してください。
<code>-i</code>	WebSEAL サーバーは、URL を大文字小文字を区別せずに処理します。 467 ページの『大文字小文字を区別しない URL のサポート』を参照してください。
<code>-p port</code>	バックエンドのサード・パーティー・サーバーの TCP ポート。デフォルトは、TCP <code>junction</code> の場合は 80 で、SSL <code>junction</code> の場合は 443 です。 423 ページの『TCP タイプの標準 <code>junction</code> の作成』および 423 ページの『SSL タイプの標準 <code>junction</code> の作成』を参照してください。

<code>-q path</code>	WebSEAL に query_contents プログラム・ファイルの正しい名前とそのファイルの検索場所を提供します。デフォルトでは、Windows ファイルは query_contents.exe と呼ばれ、UNIX ファイルは query_contents.sh と呼ばれます。デフォルトでは、WebSEAL は、バックエンド Web サーバーの cgi_bin ディレクトリーでファイルを探します。 バックエンド Windows および UNIX Web サーバーの場合は必須です。 438 ページの『Windows ベースの Web サーバーでの query_contents のインストールおよび構成』を参照してください。
<code>-T resource/resource-group</code>	GSO リソースまたはリソース・グループの名前。-b gso オプションの場合に必須で、このオプションでのみ使用されます。 602 ページの『GSO 使用可能 WebSEAL junction の構成』を参照してください。
<code>-w</code>	Windows 32 ビット (Win32) ファイル・システム・サポート。 468 ページの『Windows ファイル・システムへの junction』を参照してください。
ステートフル junction 448 ページの『ステートフル junction』を参照してください。	
<code>-s</code>	仮想ホスト junction がステートフル・アプリケーションをサポートすることを指定します。デフォルトでは、junction はステートフルではありません。
<code>-u UUID</code>	ステートフル仮想ホスト junction (-s) を介して WebSEAL に接続されたバックエンド・サーバーの UUID を指定します。
基本認証および SSL 証明書を介した相互認証 443 ページの『相互認証 SSL junction』を参照してください。	
<code>-B</code>	WebSEAL は、BA ヘッダー情報を使用して、バックエンド仮想ホストに対して認証します。-U および -W オプションが必要です。
<code>-D "DN"</code>	バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張するものです。
<code>-K "key-label"</code>	バックエンド仮想ホストに対する認証に使用する WebSEAL のクライアント・サイド認証の鍵ラベル。

<code>-U "username"</code>	WebSEAL ユーザー名。-B と共に使用して、BA ヘッダー情報をバックエンド・サーバーに送信します。
<code>-W "password"</code>	WebSEAL パスワード。-B と共に使用して、BA ヘッダー情報をバックエンド・サーバーに送信します。
プロキシ junction (-t tcpproxy または -t sslproxy が必要) 446 ページの『TCP および SSL のプロキシ junction の作成』を参照してください。	
<code>-H host-name</code>	プロキシ・サーバーの DNS ホスト名または IP アドレス。
<code>-P port</code>	プロキシ・サーバーの TCP ポート。
HTTP ヘッダーへの ID 情報の提供	
<code>-b BA-value</code>	WebSEAL サーバーがバックエンド仮想ホストに HTTP 基本認証 (BA) ヘッダーへのクライアント識別情報を渡す方法を定義します。次のいずれか 1 つになります。 filter (デフォルト)、ignore、supply、gso 590 ページの『HTTP BA ヘッダーを使用したシングル・サインオン』を参照してください。
<code>-c header-types</code>	仮想ホスト junction を介して HTTP ヘッダーに、Security Access Manager に固有のクライアント識別情報を挿入します。<i>header-types</i> 引数には、Security Access Manager HTTP ヘッダー・タイプ iv-user、iv-user-l、iv-groups、iv-creds、all を任意に組み合わせて組み込むことができます。 596 ページの『HTTP ヘッダーで提供される ID 情報』を参照してください。
<code>-e encoding-type</code>	仮想ホスト Junction に対して HTTP ヘッダーを生成するときに使用するエンコード方式を指定します。このエンコード方式は、-c Junction オプションと tag-value の両方を指定して生成されたヘッダーに適用されます。エンコードに可能な値は以下のとおりです。 • utf8_bin • utf8_uri • lcp_bin • lcp_uri 471 ページの『HTTP ヘッダー・データ用の UTF-8 エンコード』を参照してください。
<code>-I</code>	無効。仮想ホスト junction では Cookie 処理は不要であるため、このオプションは無効です。

-j	無効。仮想ホスト junction では junction Cookie 解決は不要であるため、このオプションは無効です。
-J trailer[,onfocus]	無効。仮想ホスト junction では junction Cookie 解決は不要であるため、このオプションは無効です。
-k	セッション Cookie をバックエンド仮想ホストに送信します。 466 ページの『Junction ポータル・サーバーへのセッション Cookie の引き渡し』を参照してください。
-n	無効。仮想ホスト junction では junction Cookie 解決は不要であるため、このオプションは無効です。
-r	仮想ホスト junction を介して HTTP ヘッダーに着信 IP アドレスを挿入します。 599 ページの『HTTP ヘッダーのクライアント IP アドレス (-r)』を参照してください。
junction フェアネス 59 ページの『junction 用のワーカー・スレッドの割り振り (Junction フェアネス)』を参照してください。	
-l <i>percent-value</i>	ワーカー・スレッドの消費量のソフト限界を定義します。
-L <i>percent-value</i>	ワーカー・スレッドの消費量のハード限界を定義します。
WebSphere シングル・サインオン (LTPA) junction 604 ページの『IBM WebSphere へのシングル・サインオン (LTPA)』を参照してください。	
-A	仮想ホスト junction が LTPA Cookie (トークン) をサポートできるようにします。LTPA バージョン 1 Cookie (LtpaToken) および LTPA バージョン 2 Cookie (LtpaToken2) の両方がサポートされます。LTPA バージョン 1 Cookie は、デフォルトで指定されています。LTPA バージョン 2 Cookie は、 -2 オプションを追加で使用して指定する必要があります。 -F および -Z オプションも必要です。
-2	-A オプションで使用するこのオプションは、LTPA バージョン 2 Cookie (LtpaToken2) が使用されることを指定します。 -2 オプションを指定しない -A オプションは、LTPA バージョン 1 Cookie (LtpaToken) が使用されることを指定します。
-F " <i>keyfile</i> "	LTPA Cookie データの暗号化に使用する鍵ファイルの名前。 -A オプションの場合のみ有効です。

-Z "keyfile-password"	LTPA Cookie データの暗号化に使用される鍵格納ファイルのパスワード。-A オプションの場合のみ有効です。
Tivoli Federated Identity Manager SSO junctions 587 ページの『Tivoli Federated Identity Manager を使用したシングル・サインオン』	
-Y	junction の Tivoli Federated Identity Manager シングル・サインオン (SSO) を使用可能にします。 注: このオプションを使用する前に、まず、junction 全体で Tivoli Federated Identity Manager シングル・サインオンをサポートするように WebSEAL 構成ファイルを構成する必要があります。
WebSEAL 間 SSL junction 447 ページの『SSL を介した WebSEAL から WebSEAL への junction』を参照してください。	
-C	SSL を介したフロントエンド WebSEAL サーバーとバックエンド WebSEAL サーバーの間の相互認証。-t ssl または -t sslproxy タイプが必要です。
フォーム・シングル・サインオン 607 ページの『フォーム・シングル・サインオン認証』を参照してください。	
-S path	フォーム・シングル・サインオン構成ファイルの名前。
透過パス Junction	
-x	無効。
SMS	

<code>-z replica-set-name</code>	SMS 環境: オプション。仮想ホスト junction のセッションを管理するレプリカ・セットを指定し、複数の仮想ホスト間でログイン・セッションをグループ化または分離する機能を提供します。 仮想ホスト junction のレプリカ・セットを指定する -z が使用されない場合、仮想ホスト junction は自動的に、仮想ホスト名に一致するレプリカ・セットに割り当てられます。例えば、仮想ホスト名が vhostA.example.com である場合、レプリカ・セットは vhostA.example.com という名前になります。仮想ホスト junction に使用されるレプリカ・セットは、WebSEAL 構成ファイルの [replica-sets] スタanzas内に存在しなければなりません。 SMS 以外の環境: このオプションは SMS 以外の環境では使用できません。 363 ページの『第 20 章 SMS を使用した WebSEAL の構成』を参照してください。
仮想ホスト junction ラベル	
仮想ホスト・ラベル (<i>vhost-label</i>) は、単に、仮想ホスト junction の名前です。この junction ラベルは、保護オブジェクト・スペースの表示 (Web Portal Manager) で junction を示すために使用されます。pdadmin ユーティリティでは、このラベルを使用して junction を参照できます。 **必須**。 552 ページの『リモート・タイプ仮想ホスト junction の作成』を参照してください。	

仮想ホスト junction へのサーバーの追加

操作: 既存の仮想ホスト junction にサーバーを追加します。

構文:

```
virtualhost add -h host-name options vhost-label
```

ホスト名	
<code>-h host-name</code>	追加するターゲット・バックエンド・サーバーの DNS ホスト名または IP アドレス。 **必須**。 422 ページの『標準 WebSEAL junction の構成』を参照してください。
一般オプション	
TCP および SSL junction タイプ	

<code>-a address</code>	ターゲット・バックエンド・サーバーとの通信時に WebSEAL が使用するローカル IP アドレスを指定します。このオプションを指定しない場合、WebSEAL ではオペレーティング・システムにより指定されるデフォルトのアドレスを使用します。 特定の junction のアドレスを指定すると、WebSEAL は、その junction サーバーとのすべての通信の場合にこのローカル・アドレスにバインドします。
<code>-i</code>	WebSEAL サーバーは、URL を大文字小文字を区別せずに処理します。 467 ページの『大文字小文字を区別しない URL のサポート』を参照してください。
<code>-p port</code>	バックエンドのサード・パーティー・サーバーの TCP ポート。デフォルトは、TCP junction の場合は 80 で、SSL junction の場合は 443 です。 423 ページの『TCP タイプの標準 junction の作成』および 423 ページの『SSL タイプの標準 junction の作成』を参照してください。
<code>-q url</code>	query_contents スクリプトの相対パス。デフォルトでは、WebSEAL は <code>/cgi_bin/</code> 内を検索して query_contents を見つけようとしています。これとは別のディレクトリーに入っている場合、または query_contents のファイル名が異なる場合は、このオプションを使用して、該当のファイルへの新しい URL を WebSEAL に指示してください。バックエンド Windows サーバーの場合は必須です。 438 ページの『Windows ベースの Web サーバーでの query_contents のインストールおよび構成』を参照してください。
<code>-w</code>	Windows ファイル・システム・サポート。 468 ページの『Windows ファイル・システムへの junction』を参照してください。
ステートフル junction 448 ページの『ステートフル junction』を参照してください。	
<code>-u UUID</code>	ステートフル仮想ホスト junction (-s) を介して WebSEAL に接続されたバックエンド・サーバーの UUID を指定します。
SSL を介した相互認証 443 ページの『相互認証 SSL junction』を参照してください。	

-D "DN"	バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張するものです。
プロキシ junction (-t tcpproxy または -t sslproxy が必要) 446 ページの『TCP および SSL のプロキシ junction の作成』を参照してください。	
-H <i>host-name</i>	プロキシ・サーバーの DNS ホスト名または IP アドレス。
-P <i>port</i>	プロキシ・サーバーの TCP ポート。
仮想ホスト junction ラベル	
バックエンド・サーバーの追加先仮想ホスト junction のラベル (<i>vhost-label</i>)。	

第 9 部 シングル・サインオン・ソリューション

第 31 章 junction を介したシングル・サインオン・ソリューション

この章では、WebSEAL プロキシ構成において junction バックエンド・サーバーにある保護 Web リソースへのシングル・サインオン・アクセスのためのソリューションについて説明します。

この章で扱うトピックは以下のとおりです。

- 『Tivoli Federated Identity Manager を使用したシングル・サインオン』
- 590 ページの『HTTP BA ヘッダーを使用したシングル・サインオン』
- 596 ページの『HTTP ヘッダーで提供される ID 情報』
- 600 ページの『グローバル・サインオン (GSO)』
- 604 ページの『IBM WebSphere へのシングル・サインオン (LTPA)』
- 607 ページの『フォーム・シングル・サインオン認証』

Tivoli Federated Identity Manager を使用したシングル・サインオン

WebSEAL は、Junction へのシングル・サインオンに使用できるトークンを、Tivoli Federated Identity Manager から取得することができます。シングル・サインオンに使用できる Tivoli Federated Identity Manager トークン・タイプの例は、以下のとおりです。

- Kerberos 資格情報
- LTPA トークン
- SAML トークン

Tivoli Federated Identity Manager には Tivoli Federated Identity Manager で使用可能な STS モジュールを使用して SSO トークンを生成する機能があります。

WebSEAL は以下の方法でこのモジュールにトークン要求を委任して、Tivoli Federated Identity Manager SSO トークンを取得します。

1. クライアントは HTTPS または HTTP を介して WebSEAL に対する認証を行い、Junction サーバー上のオブジェクトを要求します。ただし、Junction サーバーに対するアクセス権を取得するには、Tivoli Federated Identity Manager SSO 資格情報が必要です。
2. WebSEAL は STS モジュールに Simple Object Access Protocol (SOAP) 要求を送信して、SSO トークンを要求します。
3. Tivoli Federated Identity Manager は STS モジュールの要件に基づいてトークンを生成します。
4. Tivoli Federated Identity Manager は WebSEAL にトークンを戻し、WebSEAL はそのトークンを junction に転送します。

セキュリティー・トークンの生成を処理するには、Tivoli Federated Identity Manager 内で trust チェーンを作成する必要があります。以下の表では、trust チェーンの構成要件が強調表示されています。

表 42. Tivoli Federated Identity Manager trust チェーンの構成要件

trust チェーンのエレメント	要件	
要求タイプ	Oasis URI を発行する	
ルックアップ・タイプ	従来の WS-Trust エレメント (適用先、発行者、およびトークン・タイプ) を使用する	
適用先	アドレス	WebSEAL 構成ファイルの [tfimssso:<jct id>] スタンザ内の applies-to オプションに対応する
	サービス名	WebSEAL 構成ファイルの [tfimssso:<jct id>] スタンザ内の service-name オプションに対応する 注: このエントリー内のフィールドは、両方ともアスタリスク (*) に設定してすべてのサービス名にマッチングさせるか、2 番目のフィールドを [tfimssso:<jct id>] service-name で定義された値に設定する必要があります。trust チェーンの構成について詳しくは、Tivoli Federated Identity Manager の資料を参照してください。
	ポート・タイプ	設定しない
発行者	アドレス	amwebrte-sts-client
	サービス名	設定しない
	ポート・タイプ	設定しない
トークン・タイプ	サポートされている Tivoli Federated Identity Manager SSO トークン・タイプのいずれか	
Trust Service チェーン・モジュール	1. com.tivoli.am.fim.trustserver.sts.modules.STSTokenIVCred: -mode = validate 2. com.tivoli.am.fim.trustserver.sts.modules. any_STS_module: -mode = issue	

Tivoli Federated Identity Manager シングル・サインオン用の Junction を作成するには、オプション -Y を指定して Junction 作成コマンド (server task create) を使用します。詳しくは、717 ページの『server task create』または 752 ページの『server task virtualhost create』の『オプション』を参照してください。

注: オプション -Y を指定して junction 作成コマンドを使用する前に、Tivoli Federated Identity Manager シングル・サインオンのために特定の junction をサポートするよう WebSEAL 構成ファイルを構成する必要があります。

Tivoli Federated Identity Manager シングル・サインオン方式を使用するための構成オプションは、**[tfimssso:<jct-id>]** スタンザ内で指定します。このスタンザには、単一の junction の Tivoli Federated Identity Manager シングル・サインオン構成情報が含まれます。標準 junction の場合は、先頭のスラッシュを含む junction ポイン

ト名を使用してスタンザ名を修飾する必要があります ([tfimssso:/junction_a] など)。仮想ホスト junction の場合は、仮想ホスト・ラベルを使用してスタンザ名を修飾する必要があります ([tfimssso:www.ibm.com] など)。

[tfimssso:<jct-id>] スタンザ内の **tfim-cluster-name** オプションは、Tivoli Federated Identity Manager サービスの WebSphere クラスターの名前を定義します。クラスターのオプションを指定するには、対応する **[tfim-cluster:<cluster>]** スタンザを使用します。

[tfim-cluster:<cluster>] server スタンザ・エントリー内で、Tivoli Federated Identity Manager のプロキシとして機能する Web サーバーの優先順位および URL を指定します。**[tfim-cluster:<cluster>]** スタンザに複数の **server** エントリーを含めて、フェイルオーバーおよびロード・バランシングに使用するサーバー・エントリーを複数指定することができます。Tivoli Federated Identity Manager クラスターが構成されると、WebSEAL は Tivoli Federated Identity Manager プロキシ Web サーバーの状況を 1 分ごとにチェックします。Tivoli Federated Identity Manager

これらの構成オプションについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の **[tfimssso:<jct-id>]** スタンザおよび **[tfim-cluster:<cluster>]** スタンザを参照してください。

シングル・サインオンのこの方法を実装するには、Tivoli Federated Identity Manager の STS モジュールを使用する必要があります。詳しくは、Tivoli Federated Identity Manager の資料を参照してください。

Tivoli Federated Identity Manager との接続用の GSKit 構成

GSKit での SSL 接続の作成を制御するために使用できるよう、いくつかの GSKit 属性が用意されています。

WebSEAL は、SSL 接続を初期化するときに特定の GSKit 属性を使用するように構成できます。

[tfim-cluster:<cluster>] スタンザの **gsk-attr-name** 構成エントリーにより、Tivoli Federated Identity Manager との接続を初期化するときに WebSEAL が使用する GSKit 属性を制御します。この構成エントリーは、複数回指定することができます。必要な GSKit 属性それぞれを、新規エントリーとして含めてください。

```
[tfim-cluster:<cluster>]
gsk-attr-name = {enum | string | number}:id:value
```

注: **[ssl]** スタンザには、クライアントおよび junction Web サーバーとの接続で使用する類似の構成エントリーが存在します。

これらの構成エントリーについて詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

Kerberos 資格情報の使用

TFIM が提供できるトークン・タイプの 1 つが Kerberos 代行資格情報です。TFIM によって生成された Kerberos 資格情報を使用してシングル・サインオンを実

行する方法は、従来の Asset Manager シングル・サインオン・メカニズムを使用する方法に比べて、以下の利点があります。

- Kerberos 資格情報は、特別なコードを実装しなくても、ASP.NET Web アプリケーションを使用して簡単に利用することができます。
- Kerberos 資格情報は、暗号シグニチャーを保持しながらアプリケーション間で転送できるため、セキュリティを強化することができます。

WebSEAL で junction にシングル・サインオンするためのソリューションとして Kerberos 資格情報を使用する場合は、いくつか制限があります。Tivoli Federated Identity Manager が Windows システム上で実行中である必要があります。また、junction 先サーバーに Kerberos シングル・サインオン・ソリューションを導入すると、環境構成によってはパフォーマンスが低下します。各 Kerberos トークンの有効期間は、1 回の Kerberos 認証の間のみです。したがって、WebSEAL はトランザクションごとに新しい Kerberos トークンを要求する必要があります。WebSEAL は TFIM に対する SOAP 要求を介して間接的にトークンを要求する必要がありますが、このこともパフォーマンス低下の原因となります。junction Web サーバーがセッション状態を保持できる環境では、このソリューションによるパフォーマンス低下は最小限に抑えられます。

Kerberos トークンは 1 回限り使用するように設計されているため、WebSEAL にはパフォーマンス問題を最小限に抑えるための以下の機能が備わっています。

- バックエンド Web サーバーから 401 (許可が必要) 応答が受信された場合のみ SSO トークンが取得されるオプションを構成できます。バックエンド・サーバーにセッション状態を保持する機能がある場合、WebSEAL では不要な Kerberos トークンが取得されません。HTTP 要求があるたびにセキュリティ・トークンを送信するのか、それともトークンを要求する前に WebSEAL は 401 応答を待機する必要があるのかを指定するには、**[tfimssso:<jet-id>]** スタンザ内で **always-send-tokens** オプションを使用します。
- WS-Trust Web サービス仕様を使用すると、同じ SOAP 要求内の TFIM に対して複数の SSO トークンが要求されます。TFIM から取得するトークン数を指定するには、**[tfimssso:<jet-id>]** スタンザ内で **token-collection-size** オプションを使用します。トークンはユーザーのセッション内でキャッシュされ、以降の要求で使用されます。キャッシュされたトークンがすべて使用された場合、または有効期限が切れた場合にのみ、WebSEAL は TFIM に対して追加のトークンを要求します。

HTTP BA ヘッダーを使用したシングル・サインオン

このセクションでは、**-b** オプションを使用し、WebSEAL の複数の junction にわたって、シングル・サインオン構成を作成する場合に考えられるソリューションについて説明します。

- 591 ページの『シングル・サインオン (SSO) の概念』
- 591 ページの『HTTP BA ヘッダー内のクライアント識別』
- 592 ページの『クライアント識別と汎用パスワード』
- 593 ページの『元のクライアント BA ヘッダー情報の転送』
- 594 ページの『クライアント BA ヘッダー情報の除去』

- 595 ページの『GSO からのユーザー名とパスワード』
- 595 ページの『junction を介したクライアント識別情報』

シングル・サインオン (SSO) の概念

保護リソースが、バックエンド Web アプリケーション・サーバーに存在する場合は、そのリソースを要求するクライアントは、複数回のログインを行わなければならないことがあります。WebSEAL サーバーに 1 回と、バックエンド・サーバーに 1 回です。多くの場合、それぞれのログインごとに、異なるログイン識別が必要になります。

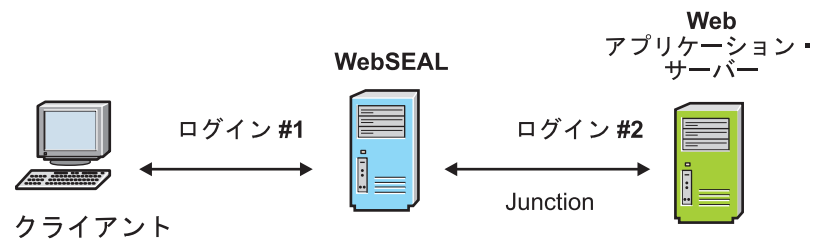


図 44. 複数回のログイン

多くの場合、複数のログイン識別の管理と保守の問題は、シングル・サインオン (SSO) ソリューションの使用により解決できます。シングル・サインオン・ソリューションによって、ユーザーは、リソースの場所には関係なく、1 回の初期ログインだけを使用して、リソースにアクセスできます。バックエンド・サーバーからの後続のログイン要件は、すべてユーザーには透過的に処理されます。

HTTP BA ヘッダー内のクライアント識別

ユーザーは、バックエンド・サーバーに対して元のクライアント識別情報または変更後のクライアント識別情報を提供するように、WebSEAL junction を構成することができます。-b オプション・セットを使用すると、特定のクライアント識別情報を HTTP 基本認証 (BA) ヘッダーに入れることができます。

アドミニストレータは、ネットワーク体系およびセキュリティー要件を分析し、以下の問題の回答を決定しなければなりません。

1. バックエンド・サーバーは、認証情報を必要とするか?

(WebSEAL は、HTTP 基本認証ヘッダーを使用して、認証情報を伝えます。)

2. バックエンド・サーバーが、認証情報を必要とするならば、この情報のソースはどこか?

(WebSEAL は HTTP ヘッダーにどんな情報を入れるのか?)

3. WebSEAL とバックエンド・サーバーの間の接続は、セキュア接続である必要があるか?

(TCP junction か SSL junction か?)

クライアントと WebSEAL の間の初期認証が行われた後に、WebSEAL は新規基本認証ヘッダーを作成します。要求は、この junction を通り、バックエンド・サーバ

ーまで行く間、この新規ヘッダーを使用します。ユーザーは この新規ヘッダーに入る特定の認証情報を指示するために、**-b** オプションを使用します。

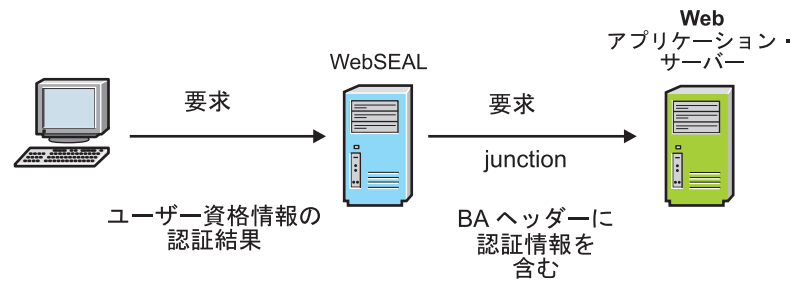


図 45. バックエンド・アプリケーション・サーバーへの認証情報の提供

クライアント識別と汎用パスワード

-b supply オプションは、認証された Security Access Manager ユーザー名 (クライアントの元の識別) を、静的な汎用 (「ダミー」) パスワードと一緒に提供するよう、WebSEAL に指示します。元のクライアント・パスワードは、このシナリオでは使用しません。

汎用パスワードによって、パスワード管理の必要がなくなり、アプリケーションはユーザー単位でサポートされます。「ダミー」パスワードを設定するには、WebSEAL 構成ファイルの **basicauth-dummy-passwd** スタンザ・エントリーを使用します。

```
[junction]
basicauth-dummy-passwd = password
```

このシナリオでは、バックエンド・サーバーが Security Access Manager ID からの認証を必要とすることを想定しています。クライアント・ユーザーを既知の Security Access Manager ユーザーにマップすることによって、WebSEAL は、バックエンド・サーバーに関する認証を管理し、ドメイン全体にわたる単純なシングル・サインオン・ソリューションを提供します。

このソリューションには、以下の条件があります。

- 元のクライアント要求に入っているユーザー名に加えて、汎用 (「ダミー」) パスワードをバックエンド・サーバーに提供するように、WebSEAL を構成します。
- 「ダミー」パスワードは WebSEAL 構成ファイル内に構成します。
- バックエンド・サーバー・レジストリーは、HTTP BA ヘッダー内に提供される Security Access Manager 識別を認識できなければなりません。
- 機密認証情報 (ユーザー名とパスワード) は、junction を介して渡されるため、junction のセキュリティは重要です。したがって、SSL junction が適切です。

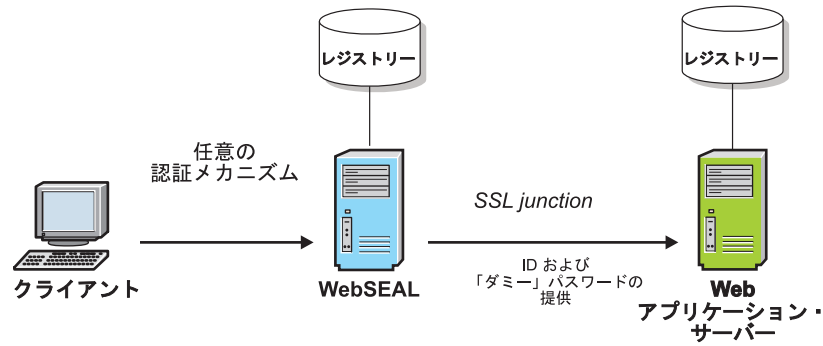


図 46. 識別と「ダミー」パスワードが含まれる BA ヘッダー

-b supply オプションの制約事項

すべての要求について同じ Security Access Manager 「ダミー」パスワードが使用されます。すべてのユーザーはバックエンド・サーバー・レジストリー内に同じパスワードをもっています。共通の「ダミー」パスワードを使用したのでは、アプリケーション・サーバーが、そのユーザー名を使用してログインする際のクライアントの正当性を証明する根拠にはなりません。

クライアントがバックエンド・サーバーにアクセスする場合に、必ず WebSEAL を通るようにすれば、このソリューションにセキュリティ上の問題が生じることはありません。ただし、考えられる他のアクセス手段から、バックエンド・サーバーを物理的に保護することも重要です。

このシナリオでは、パスワード・レベルのセキュリティがないため、バックエンド・サーバーが暗黙的に WebSEAL を信用して、クライアントの正当性を検証しなければなりません。

バックエンド・サーバー・レジストリーは、Security Access Manager 識別を受け入れるためには、その識別も認識する必要があります。

元のクライアント BA ヘッダー情報の転送

-b ignore オプションは、妨害を受けることなく、元のクライアントの基本認証 (BA) ヘッダーをバックエンド・サーバーに直接渡すよう、WebSEAL に指示します。WebSEAL は、この BA クライアント情報の認証を行うように構成することも、クライアントの提供する BA ヘッダーを無視して、ヘッダーを変更せずにバックエンド・サーバーに転送するように WebSEAL を構成することもできます。

注: これは真のシングル・サインオン・メカニズムではなく、むしろ、WebSEAL からは透過的に行われるサード・パーティー・サーバーへの直接ログインです。

このソリューションには、以下の条件があります。

- バックエンド・サーバーには、BA を介したクライアント識別情報が必要です。

バックエンド・サーバーは、基本認証試行ための質問をクライアントに返送します。クライアントは、ユーザー名およびパスワードの情報を応答として戻し、WebSEAL はこれを変更せずにそのまま通過させます。

- バックエンド・サーバーは、独自のクライアント提供のパスワードを保持します。
- 元のクライアント要求に入っているユーザー名とパスワードをバックエンド・サーバーに提供するように、WebSEAL を構成します。
- 機密認証情報 (ユーザー名とパスワード) は、junction を介して渡されるため、junction のセキュリティは重要です。SSL junction が最も適切です。

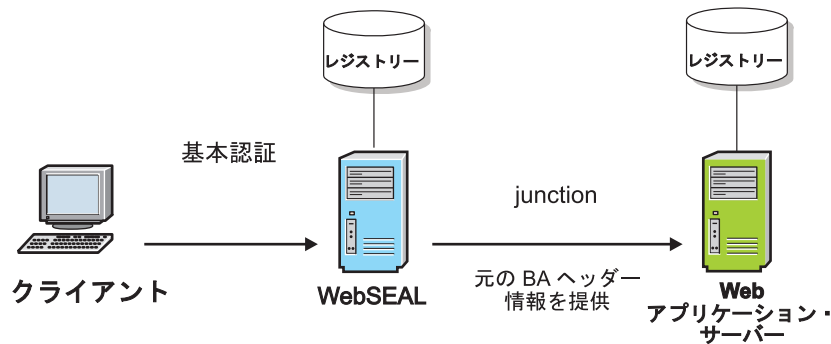


図 47. WebSEAL による元のクライアント識別情報の転送

クライアント BA ヘッダー情報の除去

-b filter オプションは、クライアント要求をバックエンド・サーバーに転送する前に、クライアント要求から基本認証ヘッダー情報をすべて除去するよう WebSEAL に指示します。このシナリオでは、WebSEAL は、単一セキュリティ・プロバイダーになります。

このソリューションには、以下の条件があります。

- クライアントと WebSEAL の間に基本認証が構成されている。
- バックエンド・サーバーは、基本認証を必要としない。
- バックエンド・サーバーには、WebSEAL を介してのみアクセスできる。
- WebSEAL が、バックエンド・サーバーに代わって認証を行う。

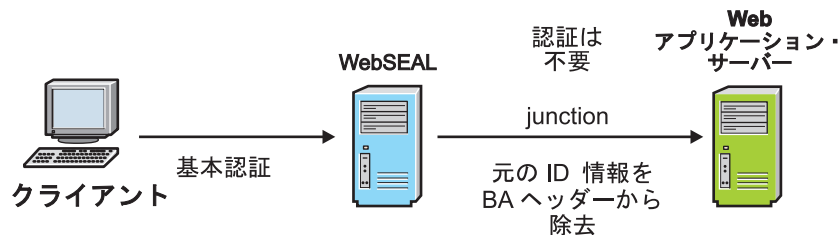


図 48. クライアント BA ヘッダー情報の除去

バックエンド・サーバーに対して何らかのクライアント情報を提供する必要がある場合は、このオプションと **-c** オプションを組み合わせ、Security Access Manager クライアント識別情報を HTTP ヘッダー・フィールドに挿入することができます。596 ページの『HTTP ヘッダー内のクライアント識別 (-c)』を参照してください。

GSO からのユーザー名とパスワード

-b gso オプションは、認証情報 (ユーザー名とパスワード) をバックエンド・サーバーに提供するように WebSEAL に指示します。この認証情報は、グローバル・サインオン (GSO) を処理するようにセットアップされているサーバーから取得されたものです。

このソリューションには、以下の条件があります。

- バックエンド・サーバー・アプリケーションは、WebSEAL レジストリーに入っていない、異なるユーザー名とパスワードを必要とします。
- WebSEAL とバックエンド・サーバーのいずれにとっても、セキュリティは重要です。

機密認証情報 (ユーザー名とパスワード) は、junction を介して渡されるため、junction のセキュリティは重要です。SSL junction が適切です。

このメカニズムについて、詳しくは、600 ページの『グローバル・サインオン (GSO)』で説明しています。

junction を介したクライアント識別情報

Junction は、BA ヘッダーにクライアント識別情報を指定するようにセットアップできます。**-b** オプションでは、4 つの引数、filter、supply、ignore、gso を使用できます。これらの引数については、590 ページの『HTTP BA ヘッダーを使用したシングル・サインオン』で詳しく説明します。

-b オプションは、相互認証の場合の junction 設定に影響を与えるので、正しい組み合わせを考慮しなければなりません。

-b supply の使用

- このオプションを使用した BA ヘッダーによる WebSEAL 認証は許可されていません。このオプションは、元のクライアント・ユーザー名と「ダミー」のパスワードについて BA ヘッダーを使用します。
- このオプションを使用したクライアント証明書による WebSEAL 認証は許可されています。

-b ignore の使用

- このオプションを使用した BA ヘッダーによる WebSEAL 認証は許可されていません。このオプションは、元のクライアント・ユーザー名とパスワードについて BA ヘッダーを使用します。
- このオプションを使用したクライアント証明書による WebSEAL 認証は許可されています。

-b gso の使用

- このオプションを使用した BA ヘッダーによる WebSEAL 認証は許可されていません。このオプションは、GSO サーバーにより指定されたユーザー名とパスワードの情報について BA ヘッダーを使用します。
- このオプションを使用したクライアント証明書による WebSEAL 認証は許可されています。

-b filter の使用

- 内部的には、**-b filter** オプションは、WebSEAL 認証が BA ヘッダー情報を使用するように設定されている場合に使用されます。

WebSEAL の BA ヘッダーは、後続のすべての HTTP トランザクションで使用されます。バックエンド・サーバーから見ると、WebSEAL に常時ログオンしているように認識されます。

- このオプションを使用したクライアント証明書による WebSEAL 認証は許可されています。
- バックエンド・サーバーが、(ブラウザからの) 実際のクライアント識別を必要とする場合には、CGI 変数の HTTP_IV_USER、HTTP_IV_GROUP、および HTTP_IV_CREDS を使用できます。スクリプトおよびサブリットの場合は、対応する Security Access Manager 固有の HTTP ヘッダー、すなわち iv-user、iv-groups、iv-creds を使用してください。

HTTP ヘッダーで提供される ID 情報

このセクションでは、以下のトピックについて説明します。

- 『HTTP ヘッダー内のクライアント識別 (-c)』
- 599 ページの『HTTP ヘッダーのクライアント IP アドレス (-r)』
- 600 ページの『WebSEAL 生成の HTTP ヘッダーのサイズの制限』

HTTP ヘッダー内のクライアント識別 (-c)

-c Junction オプションを使用すると、Security Access Manager 固有のクライアント識別、グループ・メンバーシップ、および資格情報の情報を、Junction サード・パーティー・アプリケーション・サーバー宛ての要求の HTTP ヘッダーに挿入することができます。この HTTP ヘッダー情報によって、junction 先サード・パーティー・サーバー上のアプリケーションが、クライアントの Security Access Manager 識別情報に基づいてユーザー固有のアクション (例えば、シングル・サインオン) を実行することが可能になります。

HTTP ヘッダー情報は、バックエンド・サーバー上のサービスで使用できるよう、環境変数形式に変換する必要があります。CGI プログラミングをサポートするには、ダッシュ (-) をすべて下線 (_) で置き換え、ヘッダー・ストリングの先頭に「HTTP」を付加することによって、ヘッダー情報を CGI 環境変数形式に変換します。Security Access Manager 固有の HTTP ヘッダー・エントリは、環境変数 **HTTP_IV_USER**、**HTTP_IV_USER_L**、**HTTP_IV_GROUPS**、および **HTTP_IV_CREDS** として、CGI プログラムから使用できます。

その他のアプリケーション・フレームワーク製品の場合、HTTP 要求からヘッダーを抜き出す方法については、該当する製品の資料を参照してください。

固有の HTTP ヘッダー Security Access Manager	CGI 環境変数の 変数ヘッダー	説明
iv-user	HTTP_IV_USER	クライアントのユーザー名 (ログイン ID)。クライアントが非認証 (不明) の場合は、デフォルトにより「Unauthenticated」になります。
iv-user-l	HTTP_IV_USER_L	クライアントの識別名 (DN)。
iv-groups	HTTP_IV_GROUPS	クライアントが属するグループのリスト。コンマで区切られた引用エンタリーで構成されます。
iv-creds	HTTP_IV_CREDS	Security Access Manager 資格情報を表す、エンコードされた不透明データ構造。リモート・サーバーに資格情報を提供するので、中層のアプリケーションは、許可 API を使用して許可サービス呼び出せます。 「IBM Security Access Manager for Web: Authorization C API デベロッパーズ・リファレンス」を参照してください。

`junction create` コマンド (531 ページの『第 28 章 コマンド・オプションの要約: 標準 junction』を参照してください) への **-c** オプションは、junction を介してバックエンド・アプリケーション・サーバーに送信される Security Access Manager 固有の HTTP ヘッダー・データを指定します。

-c header-types

header-types 引数には以下のものが含まれます。

引数	説明
iv_user	要求の iv-user HTTP ヘッダーに対してユーザー名 (ショート形式) を指定します。
iv_user_l	要求の iv-user-l HTTP ヘッダーに対してユーザーのフル DN (ロング形式) を指定します。
iv_groups	要求の iv-groups HTTP ヘッダーに対してグループのユーザー・リストを指定します。
iv_creds	要求の iv-creds HTTP ヘッダーに対してユーザーの資格情報を指定します。

引数	説明
all	要求の iv-user 、 iv-groups 、および iv-creds HTTP ヘッダーの識別情報を指定します。

また、**-c** オプションは、仮想ホスト **junction** でもサポートされています。547 ページの『第 29 章 仮想ホスト **junction**』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト **junction**』を参照してください。

-c junction の使用の条件

- **-c** オプションに対する複数の引数は、コンマだけで区切ります。スペースは入れないでください。
- **-c all** オプションでは、**junction** を介して、**iv-user** (**HTTP_IV_USER**)、**iv-groups** (**HTTP_IV_GROUPS**)、および **iv-creds** (**HTTP_IV_CREDS**) ヘッダーのみが渡されます。

注: **-c all** オプションでは、**iv-user-l** (**HTTP_IV_USER_L**) ヘッダーは渡されません。

- 4 種類のヘッダーをすべて **junction** を介して渡すには、以下のように 4 つのヘッダー・オプションをすべて個々に指定する必要があります。

```
-c iv_user,iv_user_l,iv_groups,iv_creds
```

- **HTTP_IV_USER** ヘッダーは、**iv-user** および **iv-user-l** のいずれかが指定されている場合にのみ、その一方に対して使用されます。**iv-user** および **iv-user-l** ヘッダーの両方が指定されている場合、**HTTP_IV_USER_L** が **iv-user-l** に対して使用されます。
- **iv-creds** ヘッダー値のセキュリティーを確実にするには、SSL **junction** を使用してください。
- コンテンツのキャッシュ・メカニズムが、**-c** および **-C** オプションを指定して構成された **junction** を介した要求に対する応答をキャッシュしないこと。

注: **-c junction** 情報をキャッシュに入れる環境変数を構成する方法については、「*IBM Security Access Manager for Web: Performance Tuning Guide*」の WebSEAL のセクションを参照してください。キャッシュ構成を適用することによって、**-c junction** の条件下で WebSEAL のパフォーマンスを向上させることができます。

-c junction の例

```
-c iv_user,iv_user_l
```

iv-user (**HTTP_IV_USER**) および **iv-user-l** (**HTTP_IV_USER_L**) ヘッダーはバックエンド・アプリケーション・サーバーに渡されます。

```
-c iv_user_l
```

iv-user-l (**HTTP_IV_USER**) ヘッダーはバックエンド・アプリケーション・サーバーに渡されます。

```
-c all
```

iv-user (HTTP_IV_USER)、**iv-groups (HTTP_IV_GROUPS)**、および **iv-creds (HTTP_IV_CREDS)** ヘッダーはバックエンド・アプリケーション・サーバーに渡されます。

```
-c iv_user,iv_user_l,iv_groups,iv_creds
```

iv-user (HTTP_IV_USER)、**iv-user-l (HTTP_IV_USER_L)**、**iv-groups (HTTP_IV_GROUPS)**、および **iv-creds (HTTP_IV_CREDS)** ヘッダーはバックエンド・アプリケーション・サーバーに渡されます。

HTTP ヘッダーのクライアント IP アドレス (-r)

-r junction オプションを使用すると、junction 先アプリケーション・サーバー宛ての要求の HTTP ヘッダーに、クライアント IP アドレスを挿入することができます。この HTTP ヘッダー情報によって、junction 先サード・パーティー・サーバー上のアプリケーションが、IP アドレスに基づいてアクションを実行することが可能になります。

HTTP ヘッダー情報は、バックエンド・サーバー上のサービスで使えるよう、環境変数形式に変換する必要があります。ダッシュ (-) をすべて下線 (_) で置き換え、ストリングの先頭に「HTTP」を付加することによって、ヘッダー情報を CGI 環境変数形式に変換します。HTTP ヘッダーの値は、新しい環境変数の値になります。

Security Access Manager に固有の HTTP ヘッダー・フィールド	CGI 環境変数の 等価変数	説明
iv-remote-address	HTTP_IV_REMOTE_ADDRESS	クライアントの IP アドレス。この値は、プロキシ・サーバーまたはネットワーク・アドレス変換機構 (NAT) の IP アドレスを表している場合もあります。
iv-remote-address-ipv6	HTTP_IV_REMOTE_ADDRESS_IPV6	IPv6 形式のクライアントの IP アドレス。

-r オプションは、着信要求の IP アドレスをバックエンド・アプリケーション・サーバーに送ることを指定します。このオプションは引数なしで指定します。

また、**-r** オプションは、仮想ホスト junction でもサポートされています。547 ページの『第 29 章 仮想ホスト junction』および 573 ページの『第 30 章 コマンド・オプションの要約: 仮想ホスト junction』を参照してください。

WebSEAL 生成の HTTP ヘッダーのサイズの制限

このタスクについて

Junction 先バックエンド・サーバーへの要求に挿入するために WebSEAL が生成する HTTP ヘッダーのサイズを、制限することができます。そのためには、WebSEAL 構成ファイルの `[junction]` スタンザ内の `max-webseal-header-size` スタンザ・エントリーに、WebSEAL 生成の HTTP ヘッダーの最大サイズ (バイト数) を指定します。「0」の値はこの機能を使用不可にします。

```
[junction]
max-webseal-header-size = 0
```

注: `max-webseal-header-size` エントリーでは、HTTP-Tag-Value ヘッダーの最大サイズに制限はありません。

WebSEAL 生成の HTTP ヘッダーが長すぎてバックエンド・アプリケーション・サーバーにより拒否されることがある場合には、このスタンザ・エントリーを使用すると便利です。例えば、多数のグループに属するユーザーの場合、`iv-creds` ヘッダーが長くなりすぎる場合があります。

このスタンザ・エントリーを構成しておくで、WebSEAL が生成したヘッダーが最大値を超えている場合は、複数のヘッダーに分割されます。次の CGI アプリケーションからの出力例は、分割ヘッダーの効果を示しています。

```
HTTP_IV_CREDS_1=Version=1, BAKs3DCCBnMMADCCBm0wggZpAgIDkDCCAYUwKzA
HTTP_IV_CREDS_2=+0+8eAgI8iAICeDYCAgCkAgFUBAaSVNCJqncMOWNuPXN1Y21==
HTTP_IV_CREDS_SEGMENTS=2
```

この機能を使用可能にした場合は、標準の WebSEAL 固有 HTTP ヘッダーの代わりに分割ヘッダーを認識するように、バックエンド・アプリケーションを変更する必要があります。

グローバル・サインオン (GSO)

グローバル・サインオン (GSO) は、バックエンド Web アプリケーション・サーバーに代替ユーザー名とパスワードを提供できるシングル・サインオン・ソリューションです。

このセクションは、以下のトピックで構成されています。

- 『グローバル・サインオンの概要』
- 601 ページの『認証情報のマッピング』
- 602 ページの『GSO 使用可能 WebSEAL junction の構成』
- 603 ページの『GSO キャッシュの構成』

グローバル・サインオンの概要

グローバル・サインオンでは、ユーザーは 1 回ログインするだけで、使用を許可されているコンピューティング・リソースへのアクセス権限を取得できます。GSO は、異機種混合の分散コンピューティング環境で複数のシステムおよびアプリケーションから構成されている大規模企業向けに設計されたもので、これにより、エンド・ユーザーは複数のユーザー名とパスワードを管理する必要がなくなります。

この統合は、WebSEAL とバックエンド Web サーバーの間に「認識」junction を作成することで達成されます。そのためには、まず、Web Portal Manager または **pdadmin** ユーティリティーを使用して、GSO リソースおよび GSO リソース・グループを作成する必要があります。

WebSEAL は、junction 先サーバー上のリソースに対する要求を受信すると、ユーザー・レジストリー・サーバーに対して適切な認証情報を要求します。ユーザー・レジストリー・サーバーには、個々の登録ユーザーのマッピングのデータベースが入っています。このデータベースは、特定のリソースおよびアプリケーション用の代替ユーザー名とパスワードを提供します。

以下の図は、GSO メカニズムを使用して、バックエンド・アプリケーション・リソース用のユーザー名とパスワードを検索する方法を示しています。

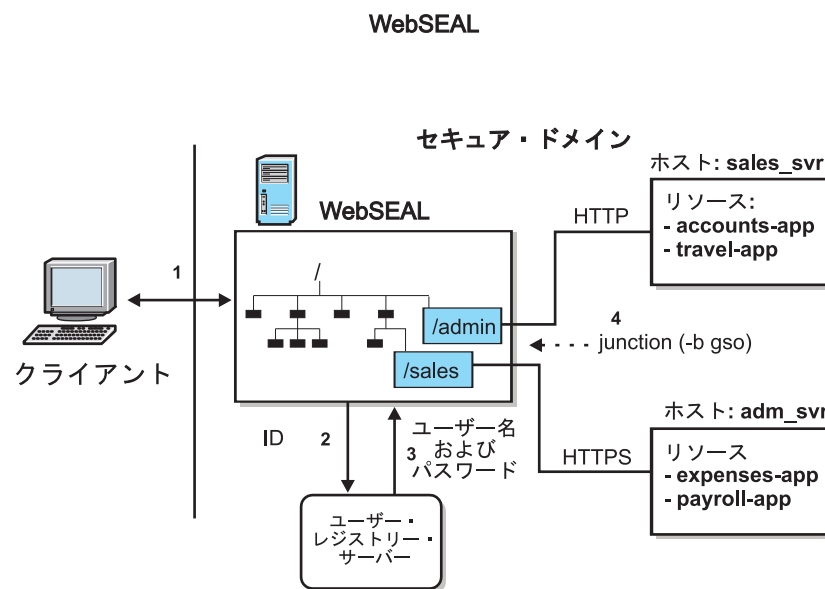


図 49. グローバル・サインオン・メカニズム

1. バックエンド・サーバー上のアプリケーション・リソースへのアクセス要求について、クライアントは WebSEAL に対して認証を行います。Security Access Manager 識別が取得されます。

注: シングル・サインオン・プロセスは、初期認証方式とは別個のものです。

2. WebSEAL は、Security Access Manager 識別をユーザー・レジストリー・サーバーに渡します。
3. レジストリー・サーバーは、ユーザーと要求されたアプリケーション・リソースに該当するユーザー名とパスワードを戻します。
4. WebSEAL は、junction を介してバックエンド・サーバーに送信される要求の HTTP 基本認証ヘッダーに、ユーザー名情報とパスワード情報を挿入します。

認証情報のマッピング

次の例は、ユーザー・レジストリーが WebSEAL に認証情報を提供する方法を示しています。

ユーザー Michael が **travel-app** アプリケーション・リソースを実行したい場合、WebSEAL はユーザー・レジストリー・サーバーに Michael の認証情報を要求します。詳細については、601 ページの図 49 のセクションを参照してください。

ユーザー・レジストリー・サーバーは、特定の認証情報へのリソース・マッピングの形式で認証情報の完全なデータベースを保守しています。認証情報は、ユーザー名とパスワードの組み合わせであり、**リソース資格情報**と呼ばれます。リソース資格情報は、登録済みユーザーについてのみ作成できます。

レジストリー・サーバーには、リソース **travel-app** を特定のリソース資格情報にマップする Michael 用のデータベースが入っています。

次の表に、GSO リソース資格情報データベースの構造を示します。

Michael	Paul
resource: travel-app username=mike password=123	resource: travel-app username=bundy password=abc
resource: payroll-app username=powell password=456	resource: payroll-app username=jensen password=xyz

この例では、レジストリーは、ユーザー名「mike」とパスワード「123」を WebSEAL に戻します。WebSEAL は、junction を介してバックエンド・サーバーに送信される要求内に基本認証ヘッダーを構成するときにこの情報を使用します。

GSO 使用可能 WebSEAL junction の構成

このタスクについて

GSO に対するサポートは、WebSEAL とバックエンド・サーバーの間の junction に構成されます。

GSO を使用可能にする junction 作成するには、**-b gso** オプションを指定した **create** コマンドを使用します。次の例に、**create** コマンドの構文を示します。

```
create -t tcp -h host-name -b gso -T resource jct-point
```

GSO junction をセットアップするためのオプションを以下にリストします。

オプション	説明
-b gso	この junction を通過するすべての要求に関して、GSO が認証情報を提供する必要があることを指定します。
-T resource/resource-group	GSO リソースまたはリソース・グループを指定します。このオプションの引数として使用されるリソース名は、GSO データベース内にリストされているリソース名に正確に一致する必要があります。GSO junction の場合は必須です。

WebSEAL/GSO ソリューションで使用される **junction** は、その **junction** の作成時に **-t ssl** オプションを追加適用することにより、SSL を通して安全を確保することができます。

SSL **junction** は必ず **GSO** と一緒に使用して、資格情報とすべてのデータを確実に暗号化するようにしてください。

GSO 使用可能 WebSEAL junction の例

ホスト **sales_svr** にあるアプリケーション・リソース **travel-app** を、**junction** ポイント **/sales** に **junction** します。

```
create -t tcp -b gso -T travel-app -h sales_svr /sales
```

ホスト **adm_svr** にあるアプリケーション・リソース **payroll-app** を、**junction** ポイント **/admin** に **junction** し、SSL を使用してその **junction** をセキュアにします。

```
create -t ssl -b gso -T payroll-app -h adm_svr /admin
```

注: この例では、**-t ssl** オプションでデフォルト・ポート 443 が指示されています。

GSO キャッシュの構成

グローバル・サインオン (GSO) キャッシュ機能を使用すると、負荷の大きい環境において **GSO junction** のパフォーマンスを高めることができます。デフォルトでは、**GSO** キャッシュは使用不可にされています。このキャッシュ拡張機能を使用しなかった場合は、**GSO** ターゲット情報 (**GSO** ユーザー名および **GSO** パスワード) を検索するたびに、ユーザー・レジストリー・サーバーを呼び出すことが必要になります。

GSO キャッシュを構成するためのスタンザ・エントリーは、**WebSEAL** 構成ファイルの **[gso-cache]** スタンザに入っています。まず、キャッシュを使用可能にする必要があります。その他のスタンザ・エントリーは、キャッシュのサイズおよびキャッシュ・エントリーのタイムアウト値を構成するために使用します。存続時間および非アクティブ・タイムアウトの値を大きくすると、パフォーマンスは向上しますが、代わりに **WebSEAL** メモリー内の情報が外部に漏れる危険性が高くなります。使用しているネットワーク・ソリューションで **GSO junction** が使用されていない場合は、**GSO** キャッシュは使用可能にしないでください。

スタンザ・エントリー	説明
gso-cache-enabled	GSO キャッシュ機能を使用可能あるいは使用不可にします。値は yes または no です。デフォルトは no です。
gso-cache-size	キャッシュ・ハッシュ・テーブル内の許容最大エントリー数を設定します。この値は、 GSO junction を介してアプリケーションにアクセスする同時ユーザー・セッションのピーク数に近い値に設定してください。この値が大きいほどメモリー使用量が増加しますが、情報アクセスの速度が高まります。1 つのキャッシュ・エントリーは約 50 バイトを消費します。

スタンザ・エントリー	説明
gso-cache-entry-lifetime	1 つのキャッシュ・エントリーが、アクティブかどうかに関係なくキャッシュ内に残ることができる最大時間 (秒)。キャッシュ・エントリーがこの時間を過ぎた場合は、そのエントリーに該当するユーザーによる次の要求では、新たにユーザー・レジストリー・サーバーを呼び出すことが必要になります。デフォルト値は 900 秒です。
gso-cache-entry-idle-timeout	非アクティブのキャッシュ・エントリーがキャッシュ内に残ることができる最大時間 (秒)。デフォルト値は 120 秒です。

IBM WebSphere へのシングル・サインオン (LTPA)

このセクションは、以下のトピックで構成されています。

- 『LTPA の概要』
- 605 ページの『LTPA junction の構成』
- 606 ページの『LTPA キャッシュの構成』
- 607 ページの『LTPA シングル・サインオンに関する技術上の注意点』

LTPA の概要

WebSEAL は、IBM WebSphere 環境に対する認証と許可サービスおよび保護を提供することができます。WebSphere は、Cookie ベースの LTPA (Lightweight Third-Party Authentication) メカニズムのサポートを提供します。

WebSEAL が WebSphere の保護フロントエンドとして配置されている場合は、ユーザーは 2 つのログイン・ポイントに出会う可能性があります。Web junction を介した 1 つ以上の IBM WebSphere サーバーへのシングル・サインオン・ソリューションを実現するには、LTPA をサポートするように WebSEAL を構成します。

ユーザーは、WebSphere のリソースを要求するときに、まず WebSEAL に対して自分の身元を証明し、認証を受ける必要があります。認証が成功すると、WebSEAL はユーザーに代わって LTPA Cookie を生成します。WebSphere 用の認証トークンとしての役割を果たすこの LTPA Cookie には、ユーザー識別、鍵およびトークン・データ、バッファ長、および有効期限に関する情報が含まれています。これらの情報は、WebSEAL と WebSphere サーバーの間で共用される、パスワードで保護された秘密鍵を使用して暗号化されます。

WebSEAL は、junction を介して WebSphere に送信される要求の HTTP ヘッダーに Cookie を挿入します。バックエンド WebSphere サーバーは、この要求を受け取り、Cookie を復号し、Cookie に含まれている識別情報に基づいてユーザーを認証します。

WebSEAL は、パフォーマンスを高めるために、LTPA Cookie をキャッシュに保管し、同じユーザー・セッション中の以後の要求にはそのキャッシュ内の LTPA

Cookie を使用することができます。キャッシュに保管される Cookie について、存続時間タイムアウトおよびアイドル (非アクティブ) タイムアウトの値を設定することができます。

WebSEAL は、LTPA バージョン 1 (LtpaToken) および LTPA バージョン 2 (LtpaToken2) の Cookie をサポートします。WebSphere サーバーが LtpaToken2 をサポートしている場合は、LTPA バージョン 2 Cookie を推奨します。

- LtpaToken

LtpaToken は、WebSphere Application Server の前のリリースと相互運用するために使用されます。このトークンは、認証 ID 属性のみを含んでいます。

LtpaToken は、WebSphere Application Server バージョン 5.1.0.2 (z/OS の場合) またはバージョン 5.1.1 (分散の場合) より前のリリースで生成されます。

- LtpaToken2

LtpaToken2 は、強い暗号化を含み、複数の属性をトークンに追加することができます。このトークンは、認証 ID および追加情報 (元のログイン・サーバーと接続するために使用される属性、固有の判別 ID 以外も考慮に入れる場合に検索に使用する固有のキャッシュ鍵など) を含みます。

LtpaToken2 は、WebSphere Application Server バージョン 5.1.0.2 (z/OS の場合) およびバージョン 5.1.1 (分散の場合) 以降用に生成されます。

ピア・サーバー環境での LTPA シングル・サインオンの使用について詳しくは、631 ページの『第 33 章 LTPA シングル・サインオン』を参照してください。

LTPA junction の構成

LTPA Cookie を使用して WebSphere へのシングル・サインオンができるようになるには、以下の構成手順が必要です。

1. LTPA メカニズムを使用可能にする。
2. 識別情報の暗号化に使用する鍵ファイルの名前を設定する。
3. この鍵格納ファイルにパスワードを提供する。
4. WebSEAL junction の LTPA Cookie 名が WebSphere LTPA Cookie 名に一致することを確認する。

LTPA トークンが含まれている WebSEAL Cookie の名前は、WebSphere アプリケーションでの LTPA Cookie の構成済みの名前と一致する必要があります。

jct-ltpa-cookie-name 構成項目は、グローバルに構成することも junction ごとに構成することもできます。この Cookie 名を構成しない場合、WebSEAL は WebSphere と同じデフォルト値を使用します。179 ページの『Junction の Cookie 名の指定』を参照してください。

最初の 3 つの構成要件は、標準 junction および仮想ホスト junction の **create** コマンドへの以下のオプションで指定します。

- **-A** オプションは、LTPA Cookie を使用可能にします。

LTPA バージョン 1 Cookie (LtpaToken) および LTPA バージョン 2 Cookie (LtpaToken2) の両方がサポートされます。LTPA バージョン 1 Cookie は、デフォルトで指定されています。LTPA バージョン 2 Cookie は、**-2** オプションを追加で使用して指定する必要があります。

-F および **-Z** オプションも必要です。

- **-2** オプションは、LTPA バージョン 2 Cookie (LtpaToken2) が使用されることを指定します。

-2 オプションを指定しない **-A** オプションは、LTPA バージョン 1 Cookie (LtpaToken) が使用されることを指定します。

- **-F "keyfile"** オプションと引数は、Cookie に格納されている識別情報を暗号化するために使用される鍵ファイルの名前を指定します。共有鍵は最初に WebSphere サーバーで作成され、安全な方法で WebSEAL サーバーにコピーされます。この操作についての詳細は、WebSphere の該当資料を参照してください。
- **-Z "keyfile-password"** は、鍵格納ファイルをオープンするために必要なパスワードを指定します。

パスワードは、junction XML ファイル内には暗号化されたテキストとして現れます。

上記のオプションは、WebSEAL とバックエンド WebSphere サーバーの間の junction を作成するときに、他の必要な junction オプションに加えて指定してください。例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task default-webseald-webseal.ibm.com create ...  
-A -F "/abc/xyz/key.file" -Z "abcdefg" ...
```

LTPA キャッシュの構成

LTPA Cookie の作成、暗号化、および復号に伴って、処理オーバーヘッドが発生します。LTPA キャッシュ機能を使用すると、負荷の大きい環境において LTPA junction のパフォーマンスを高めることができます。このキャッシュを拡張しておかないと、後続のユーザー要求のたびに、新たな LTPA Cookie が作成され暗号化されます。デフォルトでは、LTPA キャッシュは使用可能になっています。

LTPA キャッシュを構成するためのスタンザ・エントリーは、WebSEAL 構成ファイルの **[ltpa-cache]** スタンザに入っています。これらのスタンザ・エントリーは、キャッシュのサイズおよびキャッシュ・エントリーのタイムアウト値を指定します。存続時間および非アクティブ・タイムアウトの値を大きくすると、パフォーマンスは向上しますが、代わりに WebSEAL メモリー内の情報が外部に漏れる危険性が高くなります。

スタンザ・エントリー	説明
ltpa-cache-enabled	LTPA キャッシュ機能を使用可能あるいは使用不可にします。指定できる値は、「yes」および「no」です。デフォルト値は「yes」です。

スタンザ・エントリー	説明
ltpa-cache-size	キャッシュ・ハッシュ・テーブル内の許容最大エントリー数を設定します。この値は、LTPA junction を介してアプリケーションにアクセスする同時ユーザー・セッションのピーク数に近い値に設定してください。この値が大きいほどメモリー使用量が増加しますが、情報アクセスの速度が高まります。1 つのキャッシュ・エントリーは約 50 バイトを消費します。デフォルト値は 4096 エントリーです。
ltpa-cache-entry-lifetime	1 つのキャッシュ・エントリーが、アクティブかどうかに関係なくキャッシュ内に残ることができる最大時間 (秒)。キャッシュ・エントリーがこの時間を過ぎた場合は、そのエントリーに該当するユーザーによる次の要求では、新たに LTPA Cookie を作成することが必要になります。デフォルト値は 3600 秒です。
ltpa-cache-entry-idle-timeout	非アクティブのキャッシュ・エントリーがキャッシュ内に残ることができる最大時間 (秒)。デフォルト値は 600 秒です。

LTPA シングル・サインオンに関する技術上の注意点

LTPA シングル・サインオンに適用される技術上の注意事項を以下に示します。

- 鍵格納ファイルには、特定の WebSphere サーバーに関する情報が入っています。LTPA junction は、それぞれ 1 つの WebSphere サーバーに固有のものです。同じ junction ポイントに複数のサーバーを追加した場合は、すべてのサーバーが同じ鍵格納ファイルを共有します。
- シングル・サインオンが成功するためには、WebSEAL と WebSphere サーバーが同じレジストリー情報を共有する必要があります。
- LTPA のセットアップと共有秘密鍵の作成を行うのは、WebSphere サーバーの責任です。WebSEAL が担当するのは、junction とキャッシュの構成です。
- WebSphere バージョン 5.1.1 以降は、新規 LTPA バージョン 2 Cookie (LtpaToken2) をサポートします。これらの環境では、**-2** オプションを使用して LtpaToken2 サポートを指定します。
- WebSEAL は、追加の属性を LTPA Cookie の WebSphere サーバーへ渡すために、WebSphere LTPA Security Attribute Propagation を使用しません。

フォーム・シングル・サインオン認証

フォーム・シングル・サインオン認証を使用すると、WebSEAL は、認証された Security Access Manager ユーザーを、HTML フォームを使用した認証を必要とする junction 先バックエンド・アプリケーション・サーバーに、透過的にログインさせることができます。

このセクションは、以下のトピックで構成されています。

- 『フォーム・シングル・サインオンの概念』
- 『フォーム・シングル・サインオン・プロセスのフロー』
- 610 ページの『アプリケーション・サポートのための要件』
- 611 ページの『フォーム・シングル・サインオン用の構成ファイルの作成』
- 615 ページの『フォーム・シングル・サインオンを使用可能にする方法』
- 615 ページの『フォーム・シングル・サインオンの例』

フォーム・シングル・サインオンの概念

フォーム・シングル・サインオン認証は、HTML フォームを認証に使用する既存のアプリケーションをサポートするものであり、WebSEAL が行う認証を直接信任するように変更することはできません。フォーム・シングル・サインオン認証を使用可能にすると、以下のような結果が生じます。

- WebSEAL は、バックエンド・アプリケーションが開始する認証プロセスに割り込みます。
- WebSEAL は、ログイン・フォームに必要なデータを提供し、ユーザーに代わってログイン・フォームをサブミットします。
- WebSEAL は、すべての Cookie およびヘッダーを保管します。
- ユーザーは、2 番目のログインが行われていることを認識しません。
- バックエンド・アプリケーションは、ログイン・フォームがユーザーから直接送られてきたものではないことを認識しません。

WebSEAL は、次のことを行うように構成する必要があります。

- ログイン・フォームを認識してインターセプトする。
- 必要な認証データを記入する。

アドミニストレーターは、以下の方法でフォーム・シングル・サインオンを使用可能にします。

- ログイン・フォームをどのように認識、記入、および処理するかを指定する構成ファイルを作成する。
- **-S** オプション (構成ファイルの場所を指定する) を使用して `junction` を構成することにより、フォーム・シングル・サインオンを使用可能にする。

フォーム・シングル・サインオン・プロセスのフロー

以下のシナリオでは、WebSEAL が既にユーザーを認証しているものとします。

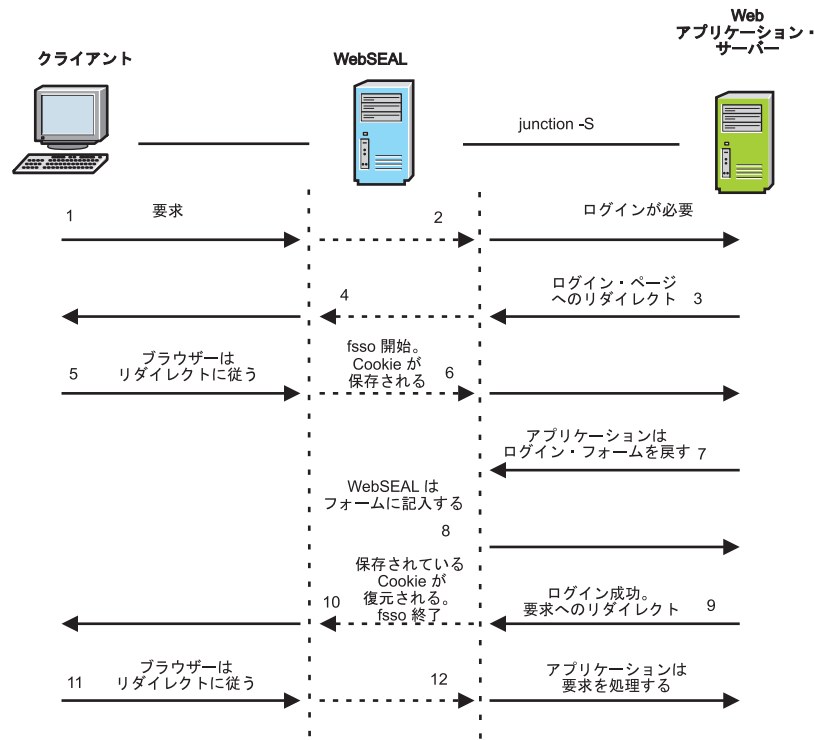


図 50. フォーム・シングル・サインオン・プロセスのフロー

1. クライアント・ブラウザは次のようにページを要求します。
`https://webseal/formssso/content.html`
2. WebSEAL は要求を junction に渡します。
3. バックエンド・アプリケーションはユーザーが認証を行うことを必要としているので、アプリケーションのログイン・ページ (`login.html`) へのリダイレクトが、junction を介して送り返されます。
4. WebSEAL はリダイレクトをブラウザに渡します。
5. ブラウザーはリダイレクトと要求に従います。
`https://webseal/formssso/login.html`

注: プロセス・フローのこの時点までは、すべて WebSEAL の標準機能です。

6. この例では、WebSEAL はフォーム・シングル・サインオン用として構成されています (junction に対する `-S` オプション)。WebSEAL は、フォーム SSO 構成ファイルに含まれている情報に基づいて、この要求がログイン・ページに対する要求であることを認識します。そして、要求は junction に渡されます。WebSEAL は、ブラウザから送信されるすべての Cookie を、ステップ 8 で使用するために保管します。
7. アプリケーションは、ログイン・ページを戻すと共に、多くの場合アプリケーション固有の Cookie も戻します。

WebSEAL は、戻された HTML の構文を解析してログイン・フォームを識別します。文書内に HTML フォームが見つかったら、WebSEAL は、そのフォーム内のアクション URI を、カスタム構成ファイル内の `login-form-action` スタンザ・エントリーの値と比較します。両者が一致した場合は、WebSEAL はそのフォームを使用します。一致しない場合は、WebSEAL はさらに他のフォーム

の検索を続けます。ページ内に、構成ファイルの中のアクション URI パターンに一致するフォームが見つからない場合は、WebSEAL はフォーム・シングル・サインオン処理を終了し、ブラウザにエラーを戻します。

WebSEAL は、HTML ページの構文を解析して、要求方式、アクション URI、およびフォーム内のその他の入力フィールドを識別し、ステップ 8 で使用するためにその情報を保管します。

8. WebSEAL は、証明書要求を生成 (ログイン・フォームに記入) し、それをバックエンド・アプリケーションに送信します。
9. アプリケーションは、WebSEAL がフォームに入れて送った認証データを使用して、ユーザーを認証します。そして、アプリケーションは `content.html` にリダイレクトを返送します。
10. WebSEAL は、ステップ 7 とステップ 9 で戻された応答から抽出して保管しておいたすべての Cookie を組み合わせて、リダイレクトと共にそれらの Cookie をブラウザに返送します。

注: これでフォーム SSO 固有の機能は完了です。

11. ブラウザーはリダイレクトと要求に従います。

`https://webseal/formssso/content.html`

12. WebSEAL は、junction を介して要求をバックエンド・アプリケーションに渡します。

上記のプロセスでは、ブラウザは WebSEAL に対する要求を 3 回実行します。しかし、ユーザーから見れば、`https://webseal/formssso/content.html` に対する要求を 1 回実行するだけです。その他の 2 回の要求は、HTTP リダイレクトにより自動的に行われます。

アプリケーション・サポートのための要件

フォーム認証のためのシングル・サインオンをサポートするためには、アプリケーションは以下の要件を満たしていることが必要です。

- アプリケーション用のログイン・ページ (1 つまたは複数) は、単一またはいくつかの正規表現により固有のものとして識別できることが必要です。
- ログイン・ページには複数の HTML フォームを含めることができます。ただし、ログイン・フォームは、各ログイン・フォームのアクション URI に正規表現を適用することにより識別する必要があります。そうでない場合は、ログイン・フォームはログイン・ページ内の最初のフォームでなければなりません。「action」属性を使用してログイン・フォームを識別する場合、「action」属性は WebSEAL の HTML フィルター操作を通過しないという点に注意してください。フィルター操作の前に、正規表現がアクション URI に一致していることが必要です。
- クライアント・サイド・スクリプトを使用して入力データを検査することができますが、このスクリプトは入力データを変更する (例えば、Javascript を使用してユーザーのブラウザ内に Cookie を設定する) ものであってはなりません。
- ログイン・データは、認証プロセスの一時点でのみサブミットされます。
- 証明書要求が渡される junction は、ログイン・ページが戻される junction と同じでなければなりません。

フォーム・シングル・サインオン用の構成ファイルの作成

フォーム・シングル・サインオン構成ファイルは、アドミニストレーターが作成するカスタム・ファイルで、任意の場所に保管できます。

junction の **-S** オプションは、フォーム・シングル・サインオン機能を有効にし、構成ファイルの名前を指定します。615 ページの『フォーム・シングル・サインオンを使用可能にする方法』を参照してください。サンプルの構成ファイル (コメント化された説明を含む) には、LMI からアクセスできます。「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「グローバル設定」>「フォーム・ベースのシングル・サインオン」に移動して FSSO 構成ファイルを作成します。

構成ファイルは、**[forms-sso-login-pages]** スタンザで始まり、以下の形式を備えている必要があります。

```
[forms-sso-login-pages]
login-page-stanza = xxxxx
#login-page-stanza = aaaaa
#login-page-stanza = bbbbbb

[xxxxx]
login-page = regular-expression-page-match
login-form-action = regular-expression-form-match
gso-resource = gso-target
argument-stanza = yyyyyy

[yyyyyy]
name = method:value
```

[forms-sso-login-pages] スタンザ

フォーム・シングル・サインオン構成ファイルは、必ず **[forms-sso-login-pages]** スタンザで始まっていなければなりません。このスタンザには、1 つまたは複数の **login-page-stanza** エントリーが含まれています。これらのエントリーは、バックエンド・アプリケーション・サーバー上のログイン・ページに関する構成情報が入っている、カスタム名を持つスタンザを指し示します。

1 つのバックエンド・サーバーが、それぞれ異なる認証方式を使用する複数のアプリケーションのホストとして機能することがあるため、1 つの junction 上で複数のログイン・ページをサポートする能力を持っていることが重要です。

例:

```
[forms-sso-login-pages]
login-page-stanza = loginpage1
login-page-stanza = loginpage2
```

カスタム・ログイン・ページ・スタンザ

カスタム・ログイン・ページ・スタンザは、特定の URL パターンをインターセプトするために使用されます。このスタンザには以下のスタンザ・エントリーを含めることができます。

スタンザ・エントリー	説明
login-page	このスタンザ・エントリーは、特定アプリケーションのログイン・ページに対する要求を一意的に識別するパターンを、正規表現を使用して指定します。WebSEAL は、該当するページをインターセプトして、フォーム・シングル・サインオン・プロセスを開始します。正規表現は、サーバーがマウントされている junction ポイントを基準とした相対的なもので (その junction ポイント自体は含まない)、要求 URI と比較されます。
login-form-action	このスタンザ・エントリーは、インターセプトされるページに含まれているどのフォームがアプリケーションのログイン・フォームかを識別するパターンを、正規表現を使用して指定します。ページ内にフォームが 1 つしかない場合、または該当のログイン・フォームが文書内の最初のフォームである場合は、この正規表現として「*」を使用できます。そうでない場合は、正規表現はログイン・フォームの「action」属性に一致している必要があります。
gso-resource	このスタンザ・エントリーは、GSO データベースから GSO ユーザー名およびパスワードを検索するときに使用する GSO リソースを指定します。GSO を使用して GSO ユーザー名およびパスワードを保管しない場合は、このスタンザ・エントリーはブランクのままにしてください。
argument-stanza	このスタンザ・エントリー・ポイントは、ログイン・フォームを完成するために必要なフィールドとデータをリストした他のカスタム・スタンザを指定します。

例:

```
[loginpage1]
login-page = /cgi-bin/getloginpage*
login-form-action = *
gso-resource =
argument-stanza = form1-data
```

login-page スタンザ・エントリーについて:

WebSEAL は、**login-page** スタンザ・エントリーの値として指定されている正規表現を使用して、着信要求が実際にログイン・ページに対する要求であるかどうかを判断します。ログイン・ページに対する要求である場合は、WebSEAL はその要求をインターセプトして、フォーム・シングル・サインオン処理を開始します。

login-page スタンザ・エントリーは、1 つのカスタム・ログイン・ページ・スタンザの中で 1 つだけ使用できます。したがって、1 つの **login-page** スタンザ・エントリーごとに 1 つずつ、カスタム・ログイン・ページ・スタンザを作成する必要があります。

login-page 正規表現は、junction を基準とした相対要求 URI と比較されます。以下の例では、「websealA」という WebSEAL サーバーに対して「junctionX」という junction 上のリソースを要求する場合の URI は、次のようになります。

<https://websealA.ibm.com/junctionX/auth/login.html>

この URL の中の下記の部分が、**login-page** の正規表現と比較されます。

/auth/login.html

login-form-action スタンザ・エントリーについて:

login-form-action スタンザ・エントリーは、インターセプトされるページ上のログイン・フォームを識別します。**login-form-action** スタンザ・エントリーは、1 つのスタンザで 1 つだけ使用できます。

login-form-action スタンザ・エントリーの値として指定した正規表現は、HTML「form」タグの「action」属性の内容と比較されます。「action」属性は、相対パス、サーバー相対パス、または絶対パスで表現された URI です。

login-form-action スタンザ・エントリーは、バックエンド・サーバーから渡された時点でのこのパスに一致していなければなりません。ただし、ほとんどの場合、WebSEAL は、クライアントにフォワードする前にこのパスを変更します。

ページ上の複数の「action」属性が正規表現に一致している場合は、最初に一致した 1 つだけがログイン・フォームとして受け入れられます。

ページ上に、正規表現に一致するフォームが 1 つもない場合は、フォームが見つからないことを示すエラーがブラウザに返送されます。

ページにログイン・フォームが 1 つしか含まれていない場合は、そのログイン・フォームに一致させるための簡便な方法として、**login-form-action = *** を設定することができます。

正規表現の使用

次の表は、フォーム・シングル・サインオン構成ファイル内で正規表現に使用できる特殊文字を示しています。

*	ゼロ個以上の任意の文字に一致します。
?	任意の 1 文字に一致します。
¥	エスケープ文字 (例えば ¥? は ? に一致します)。
[acd]	文字 a、c、または d に一致します (大文字小文字の区別あり)。
[^acd]	a、c、または d を除く任意の文字に一致します (大文字小文字の区別あり)。
[a-z]	a から z の任意の文字 (小文字) に一致します。
[^0-9]	0 から 9 を除く任意の文字 (つまり数字以外の文字) に一致します。
[a-zA-Z]	a から z (小文字) または A から Z (大文字) の任意の文字に一致します。

ログイン・ページ要求は単一の識別可能な URI なので、ほとんどの場合特殊文字は必要ありません。場合によっては、URI の末尾にある照会データが原因でログイン・ページが一致しなくなるのを防ぐために、正規表現の終わりに「*」を使用することができます。

引数スタンザ

カスタム引数スタンザには、以下の形式で 1 つまたは複数のエントリーを含めることができます。

name = method:value

name

name 変数の値は、HTML 「input」 タグの 「name」 属性と同じ値に設定されます。例えば、次のとおりです。

```
<input name=uid type=text>Username</input>
```

この変数には、HTML 「select」 または 「textarea」 タグの 「name」 属性の値も使用できます。

method:value

この組み合わせは、フォームが必要としている認証データを検索します。この認証データには以下のものが含まれます。

- リテラル・ストリング・データ

string:text

使用される入力テキスト・ストリングです。

- GSO ユーザー名およびパスワード

gso:username
gso:password

入力は、現行ユーザーの GSO ユーザー名およびパスワード (カスタム・ログイン・ページ・スタンザで指定されているターゲットからの) です。

- ユーザーの資格情報内の属性の値

cred:cred-ext-attr-name

デフォルトでは、資格情報には、ユーザーの Security Access Manager ユーザー名および DN などの情報が含まれています。ユーザーの Security Access Manager ユーザー名を入力値として使用するには、次のように値を指定します。

cred:azn_cred_principal_name

ユーザーの DN には次のようにしてアクセスできます。

cred:azn_cred_authzn_id

カスタム資格情報属性 (*tag-value* メカニズムにより追加されたもの) も使用できます。

このスタンザでは、隠し入力フィールドを指定する必要はありません。この種のフィールドは、HTML フォームから自動的に検索され、認証要求と一緒に実行依頼されます。

例:

```
[form1-data]  
uid = string:brian
```

フォーム・シングル・サインオンを使用可能にする方法

カスタム・フォーム・シングル・サインオン構成ファイルを完成し、フォーム・シングル・サインオンをサポートする `junction` を構成する必要があります。そのためには、**pdadmin create** コマンドで `-S junction` オプションを使用します。

-S config-file

config-file 引数は、カスタム・フォーム・シングル・サインオン構成ファイルのロケーションを指定します。

-S junction オプションは、junction でフォーム・シングル・サインオン機能を使用可能にします。例えば、以下のようにします (1 行で入力します)。

UNIX:

```
pdadmin> server task web1-webseald-cruz create -t tcp -h websvrA
-S fsso.conf /jctX
```

構成ファイルは、**junction** 作成時のほか、**WebSEAL** の始動のたびに読み取られます。構成ファイルにエラーがあると、始動時に **WebSEAL** が失敗することがあります。

フォーム・シングル・サインオンの例

以下のヘルプ・サイトの例は、独自のフォーム・ベースのログインを起動します。この例では、フォーム・シングル・サインオン・ソリューションにより、登録ユーザーがどのようにこのサイトにシームレスにアクセスできるかを示します。

このセクションの内容は以下のとおりです。

- ・ フォーム・セクション。これは、ヘルプ・アプリケーションが戻す HTML ログイン・ページに入れて送られるフォームと同様のものです。
- ・ このフォームを処理するために使用されるカスタム・フォーム・シングル・サインオン構成ファイル。

インターセプトされる HTML ページ内のフォーム:

[illegible]

このフォームを処理するために使用されるカスタム構成ファイル:

helpnow FSSO configuration:

```
[forms-sso-login-pages]
login-page-stanza = helpnow
```

```
[helpnow]
# The HelpNow site redirects you to this page
# you are required to log in.
login-page = /bluebase/bin/files/wcls_hnb_welcomePage1.cgi

# The login form is the first in the page, so we can just call it
# '*'.
login-form-action = *

# The GSO resource, helpnow, contains the employee serial number.
gso-resource = helpnow

# Authentication arguments follow.
argument-stanza = auth-data

[auth-data]
# The 'data' field contains the employee serial number.
data = gso:username

# The Cntselect field contains a number corresponding to the employee's
# country of origin. The string "897" corresponds to the USA.
Cntselect = string:897
```

第 32 章 クロスドメイン・シングル・サインオン

クロスドメイン・シングル・サインオン (CDSSO と呼ばれる機能) では、Web ユーザーは、リソースを要求するときに、シングル・サインオンを実行し、2 つの別個のセキュア・ドメイン間をシームレスに移動することができます。

この章では、クロスドメイン・シングル・サインオン・ソリューションをインプリメントするために必要な構成ステップと概念について説明します。

この章で扱うトピックは以下のとおりです。

- 『クロスドメイン・シングル・サインオンの概念』
- 621 ページの『クロスドメイン・シングル・サインオンの構成』
- 627 ページの『CDSSO 用の拡張属性』
- 629 ページの『クロスドメイン・シングル・サインオン用のトークンの UTF-8 エンコード』

クロスドメイン・シングル・サインオンの概念

このセクションは、以下のトピックで構成されています。

- 『クロスドメイン・シングル・サインオンの概要』
- 618 ページの『デフォルト認証トークンとカスタム認証トークン』
- 618 ページの『拡張ユーザー属性および ID のマッピング』
- 618 ページの『属性転送およびユーザー・マッピングの CDSSO プロセス・フロー』

クロスドメイン・シングル・サインオンの概要

クロスドメイン・シングル・サインオン (CDSSO と呼ばれる機能) は、固有のサーバーおよびドメイン間でユーザー資格情報を転送するためのデフォルトのメカニズムを提供します。CDSSO では、Web ユーザーは、リソースを要求するときに、シングル・サインオンを行い、2 つの別個のセキュア・ドメイン間をシームレスに移動することができます。CDSSO 認証メカニズムは、マスター認証サーバー (MAS と呼ばれる機能) に依存しません (635 ページの『第 34 章 e-community シングル・サインオン』を参照)。

CDSSO は、複数のセキュア・ドメインの統合を実現することで、スケーラブルなネットワーク体系というゴールの達成を支援します。例えば、2 つ以上の固有のドメイン (それぞれが独自のユーザーとオブジェクト・スペースを持つ) を含む、大規模な企業エクストラネットをセットアップすることができます。CDSSO により、シングル・サインオンを使用したドメイン間でのユーザーの移動が可能になります。

ユーザーが、他のドメインにあるリソースへの要求を出すと、CDSSO メカニズムは、第 1 のドメインから第 2 のドメインに、暗号化されたユーザー ID トークンを転送します。このトークンの中の識別情報は、第 1 ドメイン内でこのユーザーが正しく認証されていることを、受信側ドメインに知らせます。この識別情報にはパ

スワード情報は含まれていません。受信側サーバーはこのトークンを使用して、キャッシュ内にこのユーザー用の資格情報を作成します。ユーザーは、新たにログインする必要はありません。

デフォルト認証トークンとカスタム認証トークン

クロスドメイン・シングル・サインオン・ソリューションは、認証トークンを利用して、エンコードしたユーザー識別を宛先サーバーに送ります。第 1 のサーバーでこれらのトークンを構築する処理は、「トークン作成」と呼ばれます。宛先サーバーでトークンをデコードして使用することは、「トークン消費」と呼ばれます。WebSEAL では、組み込みのトークン作成モジュールおよびトークン消費モジュールを使用してデフォルト CDSSO 操作を行います。

あるいは、ネットワークと Security Access Manager のインプリメンテーションの特定の要件に合わせて、カスタム・トークン作成/消費モジュールを作成することができます。クロスドメイン外部認証に関する詳細情報および API 参照資料については、「*IBM Security Access Manager for Web: Web Security Developer Reference*」を参照してください。

拡張ユーザー属性および ID のマッピング

CDSSO プロセスでは、拡張属性の組み込みを可能にし、ユーザー ID を詳細に記述する目的で、クロスドメイン・マッピング・フレームワーク (場合により CDMF と呼ばれる) がサポートされています。CDMF は、トークン作成中に拡張ユーザー属性を処理し、トークン消費中にユーザー ID のマッピング・サービスを提供するプログラミング・インターフェースです。

CDSSO 実行中にデフォルトの組み込み CDMF 操作により以下が戻されます。

- CDSSO トークン作成モジュールに「SUCCESS」が戻されるが、拡張属性は戻されない。
- CDSSO トークン消費モジュールに「SUCCESS」が戻されるが、ID マッピングは戻されない。

クロスドメイン・マッピング・フレームワーク C API を使用して、ユーザー属性の処理とユーザー ID のマッピングをカスタマイズできます。クロスドメイン・マッピング・フレームワークの詳細およびその API 参照資料については、「*IBM Security Access Manager for Web: Web Security Developer Reference*」を参照してください。

あるいは、ソース・サーバーから宛先サーバーへの転送のために属性を WebSEAL 構成ファイルに指定できます。

属性転送およびユーザー・マッピングの CDSSO プロセス・フロー

以下のプロセス・フローの説明を図で表すと、620 ページの図 51 のようになります。

1. 複数のドメインに参加したいユーザーはいずれも、初期ドメイン内に有効なユーザー・アカウント、および参加リモート・ドメインのそれぞれの中の有効なアカウントにマップできる識別を持っていなければなりません。

ユーザーは、ユーザーのアカウントがある最初のセキュア・ドメイン (A) に対して最初に認証を行わなければ、CDSSO 機能を起動することはできません。

2. ユーザーは、サーバー **websealA** の Web ページ上のカスタム・リンクを使用してドメイン B 内のリソースにアクセスする要求を発行します。

このリンクは、以下のような、特別の CDSSO 管理ページ式で構成されています。

`/pkmscdsso?destination-URL`

例:

`http://websealA/pkmscdsso?https://websealB/resource.html`

注: `pkmscdsso` 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

3. この要求は、まずドメイン A 内の **websealA** サーバーにより処理されます。**websealA** サーバーは、**ssocreate** モジュールを呼び出してユーザーの資格情報を含む認証トークンを作成します。ユーザーの資格情報には、Security Access Manager 識別 (ショート・ネーム)、現行ドメイン (「A」)、追加のユーザー情報、およびタイム・スタンプが含まれています。
4. トークン作成モジュールは、ユーザー・マッピング・プロセス中にドメイン B が使用できる拡張ユーザー属性情報を取得できます。

属性情報のソースは 2 つあります。最初に WebSEAL 構成ファイルの **[cdsso-token-attributes]** スタンザに構成済みスタンザ・エントリーがあるかどうか検査されます。次に、追加属性を取得するために CDMF ライブラリーが呼び出されます (`cdmf_get_usr_attributes`)。CDMF ライブラリーの属性は、**[cdsso-token-attributes]** スタンザの設定をすべてオーバーライドします。

5. WebSEAL の triple-DES は、LMIにより生成される対称鍵を使って、このトークン・データを暗号化します。この鍵格納ファイルは、ドメイン A とドメイン B の両方の WebSEAL サーバーにより共用され、両方のサーバーの WebSEAL 構成ファイルの **[cdsso-peers]** スタンザに指定されています。
6. トークンには、トークンの存続時間を定義する構成可能なタイム・スタンプ (**authtoken-lifetime**) が入っています。タイム・スタンプが正しく構成されていると、これにより、リプレイ・アタックを防ぐことができます。
7. トークンは、**pkmscdsso** リンクに含まれている URL を使用して、宛先サーバーへリダイレクトされた要求に挿入されます。以下に例を示します。

`http://websealB/resource.html?PD-ID=encoded-authn-token&PD-REFERER=websealA`

照会ストリングの PD-ID 引数名は、WebSEAL 構成ファイルの **cdsso-argument** スタンザ・エントリーから取り込まれます。

ssocreate モジュールは、PD-REFERER 引数に起点サーバー (**websealA**) の名前を指定します。これにより、宛先サーバー (**websealB**) がブラウザの Referer ヘッダー情報を使用せずに、リダイレクトされた要求の起点を正確に認識できます。

8. **websealA** サーバーが、**websealB** のリソースに関するリダイレクト応答をブラウザに送信します。この応答には、暗号化トークンが含まれています。
9. **websealB** サーバーは、WebSEAL 構成ファイルの **cdsso-argument** スタンザ・エントリーの値に基づき、要求が CDSO トークンを含むものであると認識します。
10. **websealB** サーバーは、参照ドメインから着信するトークンをデコードし、検証します。このプロセスは「トークン消費」認証モジュール (**ssoconsume**) により実行されます。
11. 新規 ID の属性リストの作成時に、トークン消費モジュールはまず、WebSEAL 構成ファイルの **[cdsso-incoming-attributes]** スタンザの設定に基づいて属性を処理します。次に、このモジュールは CDMF ライブラリー (**cdmf_map_usr**) を呼び出します。実際のユーザー・マッピングはこのライブラリーにより実行されます。

CDMF ライブラリーはマップされたユーザー ID と拡張属性情報をトークン消費モジュールに戻します。トークン消費モジュールはこの ID を WebSEAL サーバーに渡し、WebSEAL サーバーは資格情報を作成します。

12. **websealB** 許可サービスは、ユーザーの資格情報と、要求されたオブジェクトに関連した ACL 許可および POP 許可に基づいて、保護オブジェクトへのアクセスを許可するか拒否します。

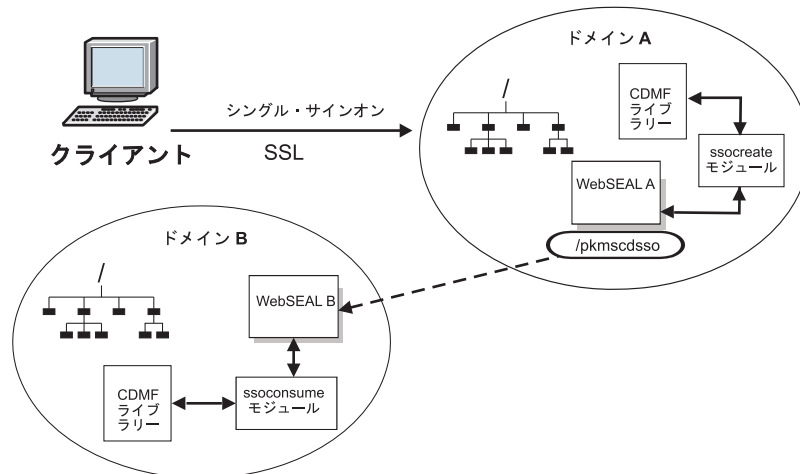


図 51. CDMF を使用したクロスドメイン・シングル・サインオン・プロセス

クロスドメイン・シングル・サインオンの構成

このセクションは、以下の基本的なトピックで構成されています。

- 『CDSSO 構成の要約』
- 622 ページの『CDSSO 条件と要件』

主な構成タスクは、以下のトピックで説明しています。

1. 623 ページの『CDSSO 認証を使用可能および使用不可にする』
2. 623 ページの『認証トークン・データの暗号化』
3. 624 ページの『トークン・タイム・スタンプの構成』
4. 625 ページの『トークン・ラベル名の構成』
5. 625 ページの『CDSSO HTML リンクの作成』

それ以外のトピックは補足的な情報です。

- 626 ページの『トークン作成中の CDMF エラーの処理』
- 626 ページの『認証トークンの保護』
- 627 ページの『仮想ホストを使用したクロスドメイン・シングル・サインオンの使用』

CDSSO 構成の要約

以下に示す個々の CDSSO 構成ステップについては、その後のセクションで詳しく説明します。

CDSSO トークン作成機能の構成

このタスクについて

初期 WebSEAL サーバーの場合の適切な手順を以下に示します。

手順

1. CDSSO トークンを生成するように WebSEAL を使用可能にする (`cdsso-create`)。
2. 組み込みトークン作成モジュールを構成する (`sso-create`)。
3. トークンをエンコードおよびデコードするために使用する鍵格納ファイルを作成する。該当するすべての参加サーバーに鍵格納ファイルをコピーする (`[cdsso-peers]` スタンザ)。
4. トークン・タイム・スタンプを構成する (`authtoken-lifetime`)。
5. トークン・ラベルを構成する (`cdsso-argument`)。
6. CDSSO HTML リンクを作成する (`/pkmscdsso`)。

CDSSO トークン消費機能の構成

このタスクについて

宛先 WebSEAL サーバーの場合の適切な手順を以下に示します。

手順

1. 認証のために CDSSO トークン (**cdsso-auth**) を消費するように WebSEAL を使用可能にする。
2. 組み込みトークン消費モジュール (**sso-consume**) を構成する。
3. 該当の鍵格納ファイルを割り当てる (**[cdsso-peers]** スタンザ)。
4. トークン・タイム・スタンプを構成する (**authtoken-lifetime**)。
5. トークン・ラベルを構成する (**cdsso-argument**)。

CDSSO 条件と要件

- CDSSO を利用するすべての WebSEAL サーバー間で、マシン時刻が同期していることが必要です。マシン間の時差が大きいと、サーバー間の認証が失敗することがあります。
- CDSSO が正しく機能するためには、この機能に参加する各 WebSEAL サーバーが、自身の完全修飾ホスト名をクロスドメイン環境内の他の参加サーバーに開示する必要があります。ドメインが含まれていないホスト名が 1 つでもあると、CDSSO は使用可能にされず、エラー・メッセージが `msg_webseald.log` に記録されます。CDSSO 環境をセットアップするときは、個々の参加サーバーに関するマシン固有ネットワーキング・セットアップにおいて、必ず完全修飾ホスト名を使用してサーバーを識別するように構成してください。
- ある種の WebSEAL 構成では、マシン・ホスト名を完全修飾ホスト名で記述することが必要とされるため、ご使用のシステムおよびネットワークがマシン名を完全修飾ホスト名に解決できることを、確認する必要があります。完全修飾ホスト名を使用すれば、例えば、複数 WebSEAL インスタンスの場合のように、1 つのマシンにつき多数のホスト名 (IP アドレス) を使用することができます。
- 特定の CDSSO 環境用に生成された鍵ペア (トークン・データの暗号化および暗号化解除に使用) を他の CDSSO 環境で再利用しないでください。常に各 CDSSO 環境に固有の鍵ペアを生成してください。

マシン名の解決

CDSSO は、マシン自体がマシン名を解決するように十分に構成されていないという理由で、WebSEAL の始動時に意図せずに使用不可にされることがあります。

このタスクについて

WebSEAL があるマシンは、IP アドレスを完全に解決できなければなりません。LMI を使用して DNS を構成することができます。

以下に説明する の一般情報は、1 つの例として使用してください。

ゴール: DNS 情報の有無についてローカルの `/etc/hosts` ファイルをチェックする前に、最初に DNS を調べるようにマシンを構成します。

手順

1. `/etc/resolv.conf` に有効な DNS サーバー・エントリーが入っていることを確認します。

2. /etc/nsswitch.conf を編集し、次のように、ホストの行が DNS 情報をチェックする正しい順序を示すようにします。

```
hosts dns files
```

代替ゴール: DNS をチェックする前に、まず、ローカル DNS 情報 (/etc/hosts) を使用するようにマシンを構成します。

タスクの結果

1. DNS を探す前に、/etc/hosts をチェックするようにマシンを構成します。/etc/nsswitch.conf を編集し、次のように、ホストの行が DNS 情報をチェックする正しい順序を示すようにします。

```
hosts files dns
```

2. 該当する DNS 情報を /etc/hosts に入力します。

```
webseal1.fully.qualified.com 1.11.111.111
```

```
webseal2.fully.qualified.com 2.22.222.222
```

CDSSO 認証を使用可能および使用不可にする

このタスクについて

初期 WebSEAL サーバーの構成

CDSSO トークン作成モジュールを使用可能にするには、WebSEAL 構成ファイルで次のスタンザ・エントリーを構成します。

```
[cdsso]
cdsso-create = {none|http|https|both}
```

値 both は、HTTP プロトコルと HTTPS プロトコルの両方を指定します。値 none は、トークン作成モジュールを使用不可にします。

宛先 WebSEAL サーバーの構成

CDSSO トークン消費モジュールを使用可能にするには、WebSEAL 構成ファイルで次のスタンザ・エントリーを構成します。

```
[cdsso]
cdsso-auth = {none|http|https|both}
```

値 both は、HTTP プロトコルと HTTPS プロトコルの両方を指定します。値 none は、トークン消費モジュールを使用不可にします。

注: WebSEAL 構成ファイルへの変更内容を活動化するには、WebSEAL サーバーを停止し、再始動する必要があります。このセクションの、該当する構成ステップをすべて完了してから WebSEAL を再始動してください。

認証トークン・データの暗号化

このタスクについて

WebSEAL は、LMI を介して生成された鍵を使用して、トークン内の認証データを暗号化しなければなりません。各参加ドメイン内の各参加 WebSEAL サーバー間で

鍵格納ファイルを共用して、この鍵を「同期化」しなければなりません。各ドメインに参加している各 WebSEAL サーバーは、同じ鍵を使用する必要があります。

注: 特定の CDSO 環境用に生成された鍵ペア (トークン・データの暗号化および暗号化解除に使用) を他の CDSO 環境で再利用しないでください。常に各 CDSO 環境に固有の鍵ペアを生成してください。

生成される鍵は、Triple-DES 192 ビット鍵です。この鍵の存続時間を指定することはできません。

注: 鍵格納ファイルの配布は、Security Access Manager の CDSO プロセスの中では行われません。

各ドメイン内の参加 WebSEAL サーバーの WebSEAL 構成ファイルの **[cdsso-peers]** スタンザで、この鍵ファイルを指定してください。形式は、以下のよう、WebSEAL サーバーの完全修飾ホスト名と、鍵ファイルの名前を含めたものです。

```
[cdsso-peers]
fully-qualified-host-name = keyfile-name
```

ドメイン A 内のサーバー websealA の場合の構成例 A:

```
[cdsso-peers]
websealB.domainB.com = A-B.key
```

この設定は、websealA が、ドメイン B 内の websealB を宛先とするトークンを暗号化するとき、どの鍵を使用するかを指定します。

ドメイン B 内のサーバー websealB の場合の構成例:

```
[cdsso-peers]
websealA.domainA.com = A-B.key
```

この設定は、websealB (ドメイン B 内の) がドメイン A 内の websealA から受信したトークンを復号するとき、どの鍵を使用するかを指定します。

この例は、1 つのマシン (例えば websealA) で A-B.key ファイルが生成され、そのファイルが手動で (そして安全に) もう一方のマシン (例えば websealB) にコピーされます。

トークン・タイム・スタンプの構成

このタスクについて

トークンには、識別トークンの存続時間を定義する構成可能なタイム・スタンプが入っています。タイム・スタンプの有効期限が切れると、そのトークンは無効であると判断され、使用されなくなります。トークンが盗まれて、その存続時間内に再生されることを防止するために、タイム・スタンプには、十分に短い値を設定して、リプレイ・アタックを防ぐようにしてください。

トークン存続時間の値を設定するには、WebSEAL 構成ファイルの **[cdsso]** スタンザ内にある **authtoken-lifetime** スタンザ・エントリーを使用します。値は、秒単位で表されます。デフォルト値は 180 秒です。

```
[cdsso]
authtoken-lifetime = 180
```

参加ドメイン間の時間のずれ (クロック・スキュー) を考慮しなければなりません。クロック・スキューとは、各ドメイン内の関係のあるサーバー上のシステム時刻に差があることを意味します。

この差が、`authtoken-lifetime` の値に近づくと、トークンの有効存続時間が大幅に減ります。この差が、`authtoken-lifetime` の値を超えると、あるドメインにあるトークンは、その他のドメインにとって有効ではなくなります。

`authtoken-lifetime` を適切に調整してください。ただし、クロック・スキューにより、`authtoken-lifetime` に大きな値を設定する必要があるときは、リプレイ・アタックのリスクが増大します。そのような場合は、各ドメインの関連するサーバー上のシステム時刻を同期してください。

詳しくは、「*IBM Security Web Gateway Appliance: Web* リバース・プロキシ・スタンザ・リファレンス」を参照してください。

トークン・ラベル名の構成

このタスクについて

リダイレクトされた要求には、シングル・サインオン・トランザクションに使用する認証情報が、その要求に対する暗号化されたトークン照会ストリング引数として組み込まれます。このトークン・ストリングには、要求を識別するための名前 (ラベル) が必要です。ラベル名は、受信側 WebSEAL サーバーに対してこの要求を一意的に識別するもので、この要求が CDSO トークン消費メカニズム (モジュール) により処理されるシングル・サインオン要求であることを示します。

このトークン・ラベルは、シングル・サインオン機能に参加する両方の WebSEAL サーバーで、まったく同じに構成する必要があります。トークン・ラベルを構成するには、WebSEAL 構成ファイルの `[cdsso]` スタンザ内にある `cdsso-argument` スタンザ・エントリーを使用します。例えば次のように指定します (デフォルトの場合)。

```
[cdsso]
cdsso-argument = PD-ID
```

詳しくは、「*IBM Security Web Gateway Appliance: Web* リバース・プロキシ・スタンザ・リファレンス」を参照してください。

CDSO HTML リンクの作成

このタスクについて

ユーザーを宛先 WebSEAL サーバー上のリソースに接続する HTML リンク (オリジナル WebSEAL サーバー上にある) では、要求を CDSO 管理ページ `pkmscdsso` にダイレクトするための特殊な CDSO 式を使用する必要があります。

```
/pkmscdsso?destination-URL
```

以下に例を示します。

<http://websealA/pkmscdsso?https://websealB/resource.html>

注: pkmscdsso 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

トークン作成モジュールは、認証トークン (ユーザーの識別情報を含む) を作成およびエンコードし、CDSSO リンク内の宛先 URL 情報を使用して、リソースにリダイレクトする応答にこのトークンを組み込みます。例:

<http://websealB/resource.html?PD-ID=encoded-authn-token&PD-REFERER=websealA>

トークン作成中の CDMF エラーの処理

このタスクについて

CDSSO トークン作成時に **ssocreate** モジュールは CDMF ライブラリーを呼び出し、トークンに組み込む拡張属性を取得します。拡張属性 (ユーザーを詳しく記述する属性) は、宛先サーバーでのユーザーの ID マッピングを正常に行うために必要となることがあります。CDMF API は **cdmf_get_usr_attributes** 呼び出しを使用して拡張属性を取得します。

cdmf_get_usr_attributes 呼び出しで必要な情報を取得できず、エラーが戻されることがあります。このような場合、トークン作成プロセスの後続の動作を制御するには、**[cdsso]** スタンザの **propagate-cdmf-errors** スタンザ・エントリーを使用します。このスタンザ・エントリーに指定できる値は「yes」および「no」です。

値「no」(デフォルト) を指定すると、CDMF が属性を取得できずエラーを戻す場合でも、トークン作成プロセスを続行できます。

値「yes」を指定すると、CDMF が属性を取得できずエラーを戻す場合に、トークン作成プロセスが強制的に終了します。

例:

```
[cdsso]
propagate-cdmf-errors = no
```

認証トークンの保護

認証トークンには、認証情報 (ユーザー名とパスワードなど) は入っていませんが、受信ドメイン内で信頼できるユーザー識別は入っています。したがって、トークン自体を、盗難とリプレイから保護する必要があります。

SSL を使用して WebSEAL サーバーとユーザーの間の通信を保護することで、トークンが保護されます。トークンは、ユーザーのブラウザー・ヒストリーから盗まれることも考えられます。トークンがトークンの存続時間内に盗まれてリプレイされる可能性がないようにするために、トークンのタイム・スタンプは、十分に短くしておく必要があります。

しかし、そのタイム・スタンプに関して有効期限が切れたトークンは、暗号アタックに対してはまだ不安定です。トークンを暗号化するために使用された鍵が発見されたり、機能が弱められたりすると、悪意を持つユーザーが独自にトークンを作成するおそれがあります。

また、トークンは偽 CDSO フローに挿入される可能性もあります。これらのトークンは、CDSO ドメインに参加している WebSEAL サーバーに対する本物の認証トークンと見分けが付きません。このため、トークンを保護するために使用した鍵を厳重に管理し、定期的に変更することも必要です。

仮想ホストを使用したクロスドメイン・シングル・サインオンの使用

565 ページの『仮想ホストを使用したクロスドメイン・シングル・サインオン』を参照してください。

CDSO 用の拡張属性

CDSO は以下の 2 とおりの方法で拡張属性を処理します。

拡張属性は 2 つの方法で処理されます。

- WebSEAL 構成ファイルの `[cdsso-token-attributes]` スタンザに構成済みスタンザ・エントリーがあるかどうかを確認されます。
- 追加属性を取得するために CDMF ライブラリーが呼び出されます (`cdmf_get_usr_attributes`)。

CDMF ライブラリーの属性は、`[cdsso-token-attributes]` スタンザの設定をすべてオーバーライドします。

CDMF ライブラリーの作成について詳しくは、「*IBM Security Access Manager for Web: Web Security Developer Reference*」を参照してください。

このセクションでは、`[cdsso-token-attributes]` スタンザの使用法を説明し、以下のトピックを扱います。

- 『トークンに追加する拡張属性』
- 628 ページの『トークンから抽出する拡張属性』

トークンに追加する拡張属性

WebSEAL 構成ファイルで、ユーザー資格情報にある拡張属性を、クロスドメイン・シングル・サインオン・トークンに追加するように指定できます。拡張属性は、ユーザー資格情報が作成されるときに拡張属性リストに追加されるユーザー ID に関する情報で構成されています。拡張属性は、カスタム認証モジュールを含む、多くの認証メカニズムによって追加できます。カスタム認証モジュールは、例えば、Security Access Manager にとって外部であるレジストリーからユーザー情報を入手するために使用できます。

ユーザーはこの設定を使用して、クロスドメイン・シングル・サインオン・トークンの内容をカスタマイズすることができます。この機能を使用すると、トークンの内容を、宛先ドメインのニーズに一致させるように調整できます。この機能を使用して属性をトークンに追加するときは、宛先ドメインのサーバーの WebSEAL 構成ファイルも構成する必要があります。宛先サーバーの場合は、各属性の処理 (抽出または無視) を指定するには、`[cdsso-incoming-attributes]` スタンザを使用します。

名前を使用して拡張属性を指定することも、複数の属性名に一致するパターンを宣言することもできます。標準の Security Access Manager ワイルドカード・マッチング文字を使用できます。サポートされるワイルドカード・パターン・マッチング文字のリストについては、77 ページの『サポートされているワイルドカード・パターン・マッチング文字』を参照してください。

各エントリーには、トークンが対象とするドメインの名前が割り当てられます。各ドメインの名前またはパターンを指定するエントリーを複数含めることができます。

構文は次のようになります。

```
[cdsso-token-attributes]
domain_name = pattern1
domain_name = pattern2
...
domain_name = patternN
<default> = pattern1
<default> = pattern2
...
<default> = patternN
```

<default> エントリーはオプションです。WebSEAL がドメイン名に一致するエントリーを見つけることができないときは、WebSEAL は <default> エントリーを探します。構成ファイルに <default> エントリーがある場合は、WebSEAL は、現在のドメインの、割り当てられた属性パターンを使用します。ストリング <default> はキーワードであるので、「<」文字と「>」文字を含めて、上に示したとおりに指定しなければなりません。

例えば、2 つのドメイン、example1.com と example2.com との間に、クロスドメイン・シングル・サインオン・ソリューションを作成するとします。ユーザーは example1.com にログインしますが、ユーザー・セッション中に、example2.com にリダイレクトすることができます。ご使用のデプロイメントには、各ユーザー資格情報に情報を挿入するカスタマイズされた外部認証 C API モジュールが組み込まれています。この情報には、固定の名前属性「job_category」と可変数の属性が入っており、それぞれに、「my_ext_attr_」という文字が先頭に付いています。この情報は、クロスドメイン・トークンに追加する必要があります。構成ファイル・エントリーは、次のようになります。

```
example2.com = job_category
example2.com = my_ext_attr_*
```

トークンから抽出する拡張属性

WebSEAL 構成ファイルで、クロスドメイン・シングル・サインオン・トークンに追加された拡張属性を、トークン消費モジュールがどのように処理するかを指定することができます。

属性は、抽出または無視できます。ある場合は、宛先ドメインの中のサーバーに属性を作成する方法がないために、属性を抽出しなければなりません。また、他の場合では、宛先ドメインの中のサーバーが独立処理を使用して同じ拡張属性を収集できるので、トークンを抽出する必要がありません。例えば、属性は、宛先サーバーのシステム時刻を反映しなければならないタイム・スタンプを反映できます。

トークン消費モジュールでは、トークンから抽出された属性は、クロスドメイン・マッピング・フレームワーク・モジュールにパススルーされます。デフォルトのクロスドメイン・マッピング・フレームワーク・モジュールは、属性を、ユーザー資格情報に直接パススルーします。カスタマイズされたクロスドメイン・マッピング・フレームワーク・モジュールは、必要に応じて属性を操作してからユーザー資格情報に渡すことができます。

エントリーの構文は、次のようになります。

```
[cdsso-incoming-attributes]
attribute_pattern = {preserve|refresh}
```

通常、拡張属性 (*attribute_pattern*) の名前は、トークンを生成する WebSEAL サーバーの構成ファイルの **[cdsso-token-attributes]** スタンザに指定されている属性の名前に一致します。使用できる値は、次のキーワードのいずれかです。

- preserve

attribute_pattern に一致するすべての属性を抽出します。

- refresh

attribute_pattern に一致する属性を抽出しません。

[cdsso-incoming-attributes] 中にあるエントリーに一致しないトークン内の拡張属性は保存 (抽出) されます。

スタンザの中のエントリーの順序は重要です。属性名に一致する最初のエントリーが使用されます。その他のエントリーは無視されます。例えば、**my_special_attr1** という名前の属性は抽出したいが、**my_special_attr_** という接頭部が付いたその他のすべてのエントリーは無視したい場合、スタンザのエントリーは次のようになります。

```
[cdsso-incoming-attributes]
my_special_attr1 = preserve
my_special_attr_* = refresh
```

上記の 627 ページの『トークンに追加する拡張属性』で示した例を使用すると、example2.com ドメインで操作されるサーバーの WebSEAL 構成ファイルのエントリーは、次のようになります。

```
[cdsso-incoming-attributes]
job_category = preserve
my_cdas_attr_1 = preserve
my_cdas_attr_* = refresh
```

この例では、属性 **job_category** および **my_cdas_attr_1** がトークンから抽出されます。接頭部 **my_cdas_attr_** が付いたその他の属性は無視されます。

クロスドメイン・シングル・サインオン用のトークンの UTF-8 エンコード

クロスドメイン・シングル・サインオンに使用するトークン内のストリングに UTF-8 エンコードを使用するか否かは、WebSEAL 構成ファイルで指定します。

```
[cdsso]
use-utf8 = {true|false}
```


デフォルト値は `true` です。

use-utf8 が「`false`」に設定されていると、ストリングはローカル・コード・ページを使用してエンコードされます。バージョン 5.1 より前のバージョンの WebSEAL サーバーにクロスドメイン・シングル・サインオンをインプリメントするときはこの値を使用します。バージョン 5.1 より前のバージョンの WebSEAL サーバーでは、トークンに UTF-8 エンコードを使用していません。これらの古いサーバーが導入されている環境に WebSEAL をデプロイするときには、UTF-8 エンコードを使用しないように WebSEAL を構成してください。この設定は、後方互換性を保つために必要です。

WebSEAL による UTF-8 エンコードのサポートについて詳しくは、65 ページの『UTF-8 でのマルチロケール・サポート』を参照してください。

第 33 章 LTPA シングル・サインオン

この章では、認証用の LTPA Cookie を使用した、ピア・セキュリティー・サーバーへのシングル・サインオン・アクセスについて説明します。

このセクションは、以下のトピックで構成されています。

- 『LTPA シングル・サインオンの概要』
- 632 ページの『LTPA シングル・サインオンの構成』
- 632 ページの『LTPA シングル・サインオンに関する技術上の注意点』

LTPA シングル・サインオンの概要

Junction を介したシングル・サインオン・ソリューションでは、WebSphere が LTPA (Lightweight Third-Party Authentication) メカニズムを使用して Junction サーバーにシングル・サインオンを提供する方法について説明しています。ピア・サーバーにシングル・サインオンを提供するには LTPA バージョン 2 も使用できます。

詳細については、587 ページの『第 31 章 junction を介したシングル・サインオン・ソリューション』を参照してください。

認証に対応したサーバー (DataPower など) が他にも存在する環境内に WebSEAL が配置される場合、潜在的に多くのログイン・ポイントが存在します。1 つ以上の WebSphere サーバーまたは DataPower サーバーへのシングル・サインオン・ソリューションを実現するには、LTPA Cookie を受け入れて LTPA Cookie 生成するように WebSEAL を構成します。

ユーザーは、WebSEAL で保護されているリソースを要求するとき、まず WebSEAL に対して自分の身元を証明し、認証を受ける必要があります。認証が成功すると、WebSEAL はユーザーに代わって LTPA Cookie を生成します。認証トークンとしての役割を果たすこの LTPA Cookie には、ユーザー ID、鍵およびトークン・データ、バッファ長、および有効期限に関する情報が含まれています。これらの情報は、WebSEAL と他の LTPA 対応サーバーとの間で共用される秘密鍵を使用して暗号化されます。

WebSEAL はクライアントに送信される HTTP 応答に Cookie を挿入します。LTPA 対応サーバーは、次の要求でこの Cookie を受け取り、Cookie を復号し、Cookie に含まれている識別情報に基づいてユーザーを認証します。

WebSEAL は LTPA バージョン 2 (LtpaToken2) の Cookie のみサポートします。

LTPA シングル・サインオンの構成

このタスクについて

LTPA Cookie は、LTPA 認証が WebSEAL 内で使用可能な場合に生成されます。これらの Cookie は、LTPA 対応の他の認証サーバーに対してシングル・サインオンを実現するために使用されます。詳しくは、177 ページの『LTPA 認証』を参照してください。

手順

LTPA Cookie を使用して他の LTPA 対応サーバーへのシングル・サインオンができるようにするには、以下の構成手順が必要です。

1. LTPA メカニズムを使用可能にする。
2. 識別情報の暗号化に使用する鍵ファイルの名前を設定する。
3. この鍵格納ファイルにパスワードを提供する。
4. WebSEAL junction の LTPA Cookie 名が WebSphere LTPA Cookie 名に一致することを確認する。

LTPA トークンを含む WebSEAL Cookie の名前は、WebSphere アプリケーションの LTPA Cookie の構成済みの名前と一致している必要があります。

jct-ltpa-cookie-name 構成項目は、グローバルに構成することも junction ごとに構成することもできます。この Cookie 名を構成しない場合、WebSEAL は WebSphere と同じデフォルト値を使用します。179 ページの『Junction の Cookie 名の指定』を参照してください。

最初の 3 つの構成要件は、標準 junction および仮想ホスト junction の **create** コマンドのオプションで指定します。上記のオプションは、WebSEAL とバックエンド WebSphere サーバーの間の junction を作成するときに、他の必要な junction オプションに加えて指定してください。例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task default-webseald-webseal.ibm.com create ...  
-A -F "key.file" -Z "abcdefg" ...
```

これらのオプションの詳細については、177 ページの『LTPA 認証』で説明しています。

LTPA シングル・サインオンに関する技術上の注意点

LTPA シングル・サインオンに適用される技術上の注意事項を以下に示します。

- 鍵ファイルには、特定の LTPA 対応認証サーバーに関する情報が含まれます。LTPA Cookie を生成および認証するときは WebSEAL によって単一の鍵ファイルが使用されるため、LTPA 対応のすべてのサーバーが同じ鍵格納ファイルを共用する必要があります。同じ junction ポイントに複数のサーバーを追加した場合は、すべてのサーバーが同じ鍵格納ファイルを共用します。
- シングル・サインオンが成功するためには、WebSEAL と LTPA 対応の認証サーバーが同じレジストリー情報を共用する必要があります。
- LTPA のセットアップと共有秘密鍵の作成を行うのは、WebSphere サーバーの責任です。

- WebSphere は LTPA バージョン 2 の Cookie のみサポートします。
- WebSEAL は、追加の属性を LTPA トークンに含めるために WebSphereLTPA Security Attribute Propagation を使用しません。

第 34 章 e-community シングル・サインオン

e-community シングル・サインオン (ECSSO と呼ばれる) により、認証済みユーザーが複数ドメインの複数サーバー上の保護リソースにアクセスするときに、追加ログオンが不要になります。

この章では、e-community シングル・サインオン・ソリューションをインプリメントするために必要な構成ステップと概念について説明します。

この章で扱うトピックは以下のとおりです。

- 『e-community シングル・サインオンの概念』
- 645 ページの 『e-community シングル・サインオンの構成』
- 655 ページの 『ECSSO 用の拡張属性』
- 657 ページの 『e-community シングル・サインオン用のトークンの UTF-8 エンコード』

e-community シングル・サインオンの概念

このセクションは、以下のトピックで構成されています。

- 『e-community の概要』
- 637 ページの 『e-community の機能と要件』
- 637 ページの 『e-community のプロセス・フロー』
- 643 ページの 『e-community Cookie』
- 643 ページの 『vouch-for 要求と応答』
- 644 ページの 『vouch-for トークン』

e-community の概要

e-community シングル・サインオン (ECSSO と呼ばれる機能) は、Security Access Manager 環境におけるクロスドメイン認証の別のインプリメンテーションです。クロスドメイン認証の目的は、ユーザーが、複数のログイン操作を行わずに、複数ドメイン内の複数サーバーのリソースにアクセスできるようにすることです。

"e-community" は、1 つのビジネス相互関係に参加する複数の異なるドメイン (Security Access Manager または DNS) から成るグループです。これらの参加ドメインは、1 つの企業の一部として構成することも (例えば地理的な事情により複数の DNS 名を使用する場合)、または、共用関係を持つ複数の異なる企業 (例えば、企業本部、生命保険会社、および財務管理会社など) として構成することもできます。

いずれのシナリオの場合も、常に、「ホーム」または「所有者」として指定されたドメインが 1 つあります。複数企業が参加する構成の場合は、ホーム・ドメインは e-community を規制する企業間協定を所有しています。

どちらのシナリオの場合も、ホーム・ドメインには、e-community に参加するユーザーに関する認証情報（認証に使用されるユーザー名やパスワードなど）が保持されています。この配置により、シングル・ポイント方式での各種管理事項に関する参照が可能になります。例えば、e-community 内でのヘルプ・デスク・コールは、すべてホーム・ドメインに送られます。

代わりに、Security Access Manager Web Portal Manager を使用してこの情報の管理を委任し、参加ドメインにそれぞれのユーザーの管理責任を持たせることもできます。

下の図は、ドメイン A (dA.com) およびドメイン B (dB.com) の 2 つの参加ドメインからなるサンプル e-community を示しています。この例では、ドメイン A がホーム（所有者）ドメインの役割を果たしています。ドメイン B は、参加ドメイン、つまり「リモート」ドメインです。

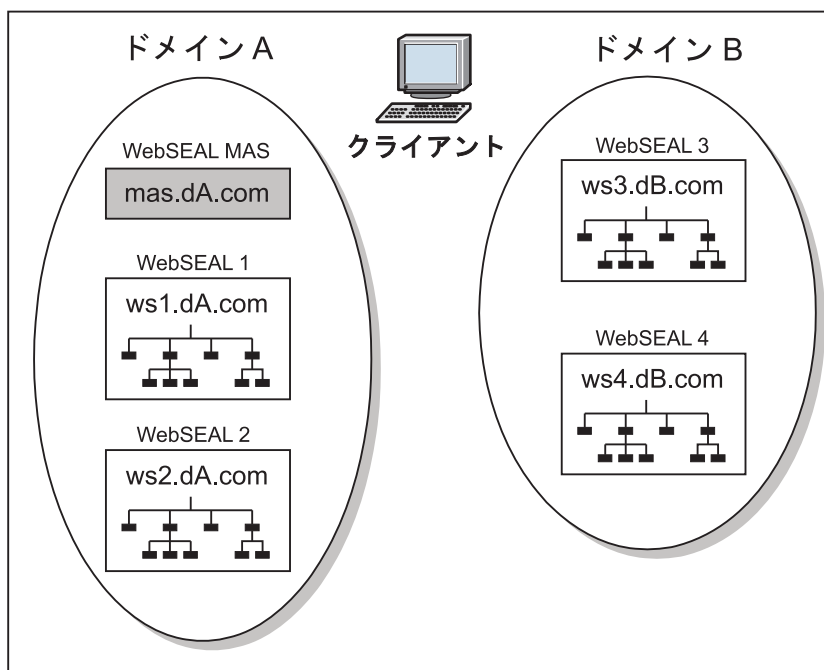


図 52. e-community のモデル

ホーム・ドメインは、ユーザーを「所有」しています。つまり、ユーザーの認証情報をコントロールします。ユーザーがリソースに対する要求をどこで出すかに関係なく、ユーザーは常にホーム・ドメインで認証を行う必要があります。

認証は、マスター認証サーバー (MAS) に照らして行われます。MAS は、ホーム・ドメイン内にあって、すべてのユーザーを認証するように構成されているサーバー（またはレプリカ・サーバー・セット）です。図では、MAS は mas.dA.com と表記されています。MAS の役割は、認証サービスを提供することのみに限定するようにしてください。つまり、MAS にはユーザーが利用できるリソースは含めないようにします。

ユーザーが MAS に対して正しく認証されると、MAS は「vouch-for」トークンを生成します。このトークンは、ユーザーが要求を出したサーバーに戻されます。サー

バーは、この vouch-for トークンを、ユーザーが MAS に対して正しく認証され、e-community に参加する資格があることを示す証明と見なします。

e-community のドメイン相互間での情報の転送については、『e-community のプロセス・フロー』のセクションで詳しく説明します。

e-community の機能と要件

- ECSO モデルは、直接 URL (ブックマーク) を使用したリソースへのアクセスをサポートしています。この機能と対照的に、クロスドメイン・シングル・サインオン (CDSSO) モデルの場合は、特別に構成された **pkmscdsso** リンクが必要です (617 ページの『クロスドメイン・シングル・サインオンの概念』を参照)。
- e-community に参加するすべてのユーザーは、「ホーム」ドメイン内に 1 つだけ配置されているマスター認証サーバー (MAS) に照らして認証されます。
- e-community インプリメンテーションでは、ユーザーが MAS に対して有効なアカウントを持っていない (例えば、ドメイン A が「ホーム」ドメインになっていて、ユーザーがドメイン B に属しているが、ドメイン A とドメイン B の e-community には参加していない) 場合に、「ローカル」認証ができるようになります。
- 認証の失敗を MAS で処理するように WebSEAL が構成されている場合を除き、ユーザーが MAS 以外の (ただし参加している) ドメイン内にあるリソースを要求し、MAS に対する認証が失敗した場合、そのユーザーには、その要求を行っているローカル・サーバーに対して認証するオプションが与えられます。
- MAS (そして最終的にはリモート・ドメイン内の他の選択されたサーバー) は、ユーザーの認証済みの識別を「vouch for」(保証) します。
- vouch-for サービスを提供できるサーバーを識別するためには、ドメイン固有の Cookie が使用されます。ドメイン Cookie を使用することにより、リモート・ドメイン内のサーバーは、ローカルで vouch-for 情報を要求することができます。e-community Cookie の暗号化されたコンテンツには、ユーザー識別またはセキュリティ情報は含まれていません。
- 暗号化された「vouched for」(保証済み) ユーザー識別を渡すためには、特殊なトークンが使用されます。vouch-for トークンには、実際のユーザー認証情報は含まれていません。保全性は、共有秘密鍵 (Triple-DES) により確保されます。このトークンには、トークンの有効性が持続する期間を制限するタイムアウト (存続時間) 値が含まれています。
- WebSEAL の構成オプションで、使用可能にすると MAS にのみ「vouch-for」トークンの生成を許可するものがあります。

e-community のプロセス・フロー

e-community は、1 つのマスター認証 WebSEAL サーバー (MAS) と、ホーム・ドメインおよびリモート・ドメインに含まれるその他の WebSEAL サーバーから成っています。MAS は、1 つの WebSEAL サーバーの単一インスタンスとして存在することも、ロード・バランサーの背後に位置する一組の WebSEAL レプリカとして存在することもできます (この場合はロード・バランサーが MAS として認識されます)。

参加するすべてのローカルおよびリモート WebSEAL サーバーを、初期クライアント認証にホーム・ドメイン MAS を使用するように構成する必要があります。この構成は、ホーム・ドメイン内のサーバーの場合は必須要件ですが、リモート・ドメイン内のサーバーの場合は必須ではありません。例えば、リモート・ドメイン内の一部のサーバーが、それぞれ自身の認証を処理するように構成することができます。これらのサーバーとそれにより保護されているリソースは、e-community 参加ドメイン内に含まれていても、e-community から独立して運用することができます。

e-community インプリメンテーションは、vouch-for システムに基づいています。あるユーザーとある WebSEAL サーバーの間にまだ有効なセッションが確立されていないときに、そのユーザーがそのサーバーにあるリソースを要求した場合、通常、WebSEAL はユーザーに認証情報を入力するようプロンプトを出します。

e-community 構成では、WebSEAL サーバーは vouch-for サーバーを識別し、この vouch-for サーバーに対して、ユーザーが認証済みであることを確認するよう要求します。

vouch-for サーバーは、そのユーザーの有効な資格情報を保持しています。ユーザーの最初の要求では、vouch-for サーバーは常に MAS になります。ホーム・ドメイン内にあるリソースについては、引き続き MAS が vouch-for サーバーとしての役割を果たします。このユーザーが e-community 内でリソース要求を続けた場合、各リモート・ドメイン内の個々のサーバーは、このユーザーについて独自の資格情報を作成し (MAS からのユーザー識別情報を使用)、自身のドメイン内にあるリソースについて vouch-for サーバーの役割を代行することができます。

vouch-for サーバーに要求される検証は、vouch-for トークンの形式をとります。vouch-for サーバーはこのトークンを作成して、要求元の WebSEAL サーバーに戻します。トークン内のユーザー識別情報は暗号化されます。トークンには存続時間の制限値が含まれています。

要求元サーバーは、vouch-for トークンを受け取ると、そのユーザー用の資格情報とローカル・セッションを作成します。これで、ユーザーは、通常の許可コントロールに基づいて、要求したリソースにアクセスできるようになります。ユーザーにとっては、再度ログインする必要がないという利点があります。これが e-community モデルの目的です。

以下の e-community プロセス・フローは、次の図を参照しながら進めてください。以下のプロセス・フローでは、まず 2 つの初回アクセス・シナリオ (1 および 2) について説明します。次に、2 または 3 に続く 2 つの次回アクセス・シナリオ (3 および 4) について説明します。シナリオ 5 は、初回アクセスの後の任意の時点で発生します。

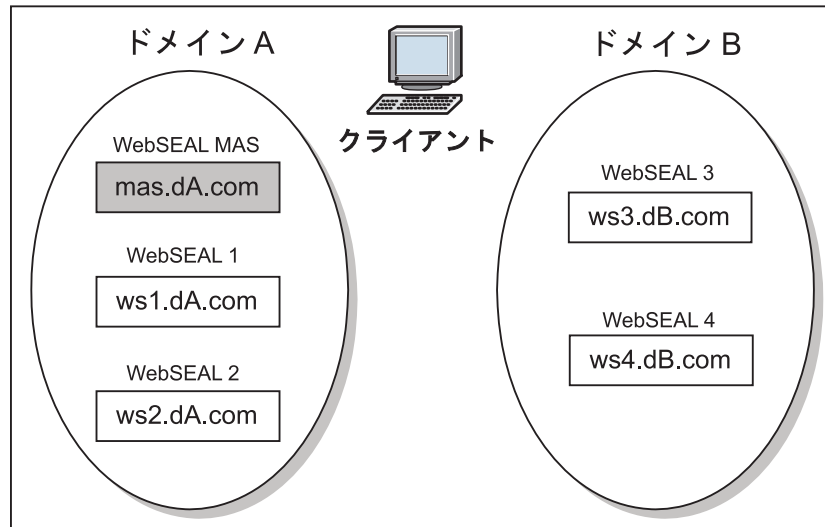


図 53. e-community のプロセス・フローの構成の例

vouch-for サーバー

- e-community のいずれかの部分に初めてアクセスしようとするユーザーを認証するには、常に MAS が使用されます。

MAS には、リソース・プロバイダーではなく認証サーバーとしての役割のみを持たせるのが、最も効果的です。MAS には、マスター認証サーバーとしての機能とリソースを保護する機能とを同時に構成しないでください。ただし、これを推奨するのはパフォーマンスを確保するためであり、セキュリティ上の要件ではありません。

- MAS は、常にホーム・ドメイン（この例ではドメイン A）の vouch-for サーバーです。
- 指定されたドメイン内の他のすべてのサーバー用の vouch-for サーバーは、ドメイン固有の e-community Cookie により識別されます。vouch-for サーバーは、ドメイン内で MAS からの vouch-for トークンを要求する最初のサーバーです。vouch-for サーバーはドメイン内ユーザーに vouch-for 情報を提供します。指定されたリモート・ドメイン内の vouch-for サービスを求める以後の要求は、ドメイン外の MAS にアクセスせずに、このサーバーがローカルで行うことができます。ホーム・ドメイン内では、e-community Cookie は MAS を vouch-for サーバーとして識別します。

(1) 初回の e-community ローカル（ドメイン A）アクセス: WebSEAL 1

1. ユーザーは、WebSEAL 1 (MAS と同じドメイン内にある) により保護されているリソースを要求します。ブラウザーには、このドメイン用の e-community Cookie は含まれていません。WebSEAL 1 のキャッシュには、このユーザーの資格情報は入っていません。
2. WebSEAL 1 の構成では、e-community 認証が使用可能にされ、MAS の場所が指定されています。WebSEAL 1 は、ブラウザーを MAS 上の特殊 vouch-for URL にリダイレクトします。
3. MAS は vouch-for 要求を受け取りますが、このユーザーの資格情報が見つからないため、ユーザーにログインするようプロンプトを出します。

4. ログインが成功すると、MAS はこのユーザー用の資格情報を作成してキャッシュに保存してから、暗号化した vouch-for トークンを添えて、ブラウザーを、最初に要求されていた WebSEAL 1 上の URL にリダイレクトします。これに加えて、ドメイン A の vouch-for サーバー (この場合は MAS) を識別するために、ドメイン A 固有の e-community Cookie もブラウザーに渡されます。
5. トークン作成モジュールは、ユーザー・マッピング・プロセスにおいて宛先サーバーが使用できる拡張ユーザー属性情報を取得できます。

属性情報のソースは 2 つあります。最初に WebSEAL 構成ファイルの **[ecssso-token-attributes]** スタンザに構成済みスタンザ・エントリーがあるかどうか調べられます。次に、追加属性を取得するために CDMF ライブラリーが呼び出されます (**cdmf_get_usr_attributes**)。CDMF ライブラリーの属性は、**[ecssso-token-attributes]** スタンザの設定をオーバーライドします。

6. ログインの試行が失敗した場合は、MAS は失敗状況を示す vouch-for トークンを戻します。このトークンは、成功状況を示す vouch-for トークンとは区別できないような形で作成されます。要求元サーバーは、失敗状況トークンに応答して、ユーザーがローカル認証に失敗した場合と同じ処置を実行します。
7. WebSEAL 1 はトークンを復号します。

注: 同一ドメイン内では、通常は識別のマッピングは必要ありません。識別のマッピングが必要な場合は、WebSEAL 1 はクロスドメイン・マッピング・フレームワーク (CDMF) を使用できます。

8. 新規 ID の属性リストが作成されると、トークン消費モジュールはまず、WebSEAL 構成ファイルの **[ecssso-incoming-attributes]** スタンザの設定に基づいて属性を処理します。次に、このモジュールは CDMF ライブラリー (**cdmf_map_usr**) を呼び出します。実際のユーザー・マッピングはこのライブラリーにより実行されます。
9. CDMF ライブラリーはマップされたユーザー ID と拡張属性情報をトークン消費モジュールに戻します。トークン消費モジュールはこの ID を WebSEAL サーバーに渡し、WebSEAL サーバーは資格情報を作成します。
10. 許可サービスは、ユーザーの資格情報と、要求されたリソースに関連した特定の ACL 許可および POP 許可に基づいて、要求を許可または拒否します。

(2) 初回の e-Community リモート (ドメイン B) アクセス: WebSEAL 3

1. ユーザーは、WebSEAL 3 (リモート・ドメイン B) により保護されているリソースを要求します。ブラウザーには、このドメイン用の e-community Cookie は含まれていません。WebSEAL 3 のキャッシュには、このユーザーの資格情報は入っていません。
2. WebSEAL 3 の構成では、e-community 認証が使用可能にされ、MAS の場所が指定されています。WebSEAL 3 は、ブラウザーを MAS 上の特殊 vouch-for URL にリダイレクトします。
3. MAS は vouch-for 要求を受け取りますが、このユーザーの資格情報が見つからないため、ユーザーにログインするようプロンプトを出します。
4. ログインが成功すると、MAS はこのユーザー用の資格情報を作成してキャッシュに保存してから、暗号化した vouch-for トークンを添えて、ブラウザーを、最初に要求されていた WebSEAL 3 上の URL にリダイレクトします。これに

加えて、ドメイン A の vouch-for サーバー (この場合は MAS) を識別するために、ドメイン A 固有の e-community Cookie もブラウザに渡されます。

5. トークン作成モジュールは、ユーザー・マッピング・プロセスにおいて宛先サーバーが使用できる拡張ユーザー属性情報を取得できます。

属性情報のソースは 2 つあります。最初に WebSEAL 構成ファイルの **[ecssso-token-attributes]** スタンザに構成済みスタンザ・エントリーがあるかどうか調べられます。次に、追加属性を取得するために CDMF ライブラリーが呼び出されます (**cdmf_get_usr_attributes**)。CDMF ライブラリーの属性は、**[ecssso-token-attributes]** スタンザの設定をオーバーライドします。

6. ログインの試行が失敗した場合は、MAS は失敗状況を示す vouch-for トークンを返します。このトークンは、成功状況を示す vouch-for トークンとは区別できないような形で作成されます。MAS でのユーザー認証が失敗した場合は、ユーザーには、WebSEAL 3 でローカル認証を受けるようプロンプトが出されます。このサーバーにこのユーザーのアカウントが存在していれば、認証は成功します。
7. WebSEAL 3 はトークンを復号します。
8. 新規 ID の属性リストが作成されると、トークン消費モジュールはまず、WebSEAL 構成ファイルの **[ecssso-incoming-attributes]** スタンザの設定に基づいて属性を処理します。次に、このモジュールは CDMF ライブラリー (**cdmf_map_usr**) を呼び出します。実際のユーザー・マッピングはこのライブラリーにより実行されます。
9. CDMF ライブラリーはマップされたユーザー ID と拡張属性情報をトークン消費モジュールに戻します。トークン消費モジュールはこの ID を WebSEAL サーバーに渡し、WebSEAL サーバーはユーザーの資格情報を作成します。
10. WebSEAL 3 は、第 2 の e-community Cookie (ドメイン B 用として有効なもの) を作成し、ブラウザ上にこの Cookie を設定します。この Cookie は、WebSEAL 3 をドメイン B の vouch-for サーバーとして識別します。
11. 許可サービスは、要求を許可または拒否します。

(3) 次回の e-Community ローカル (ドメイン A) アクセス: WebSEAL 2

1. ユーザーは、WebSEAL 2 (MAS と同じドメイン内にある) により保護されているリソースを要求します。ブラウザには、MAS を vouch-for サーバーとして識別するドメイン A e-community Cookie が含まれています。WebSEAL 2 はこの Cookie を受け取ります。WebSEAL 2 のキャッシュには、このユーザーの資格情報は入っていません。
2. WebSEAL 2 の構成では、e-community 認証が使用可能にされ、MAS の場所が指定されています。しかし、ドメイン A e-community Cookie が存在するため、MAS の場所を指定した WebSEAL 2 構成はオーバーライドされます。この Cookie を使用して、WebSEAL 2 は vouch-for サーバーを識別します。(シナリオ 2 が最初に発生しているとすれば、ブラウザにはドメイン B の Cookie も保持されていますが、これはドメイン A のサーバーには送られません。)
3. WebSEAL 2 は、Cookie で識別されているドメイン A の vouch-for サーバー上の特殊 vouch-for URL に、ブラウザをリダイレクトします (この例では、WebSEAL 2 はドメイン A 内にあるので、vouch-for サーバーは MAS です)。

4. MAS は vouch-for 要求を受け取り、キャッシュ内にこのユーザー用の資格情報を見つけます (資格情報はシナリオ 1 および 2 で作成されています)。
5. MAS は、暗号化した vouch-for トークンを添えて、ブラウザーを、最初に要求されていた WebSEAL 2 上の URL にリダイレクトします。(拡張属性の処理の説明については、シナリオ 1 と 2 を参照。)
6. WebSEAL 2 は、トークンを復号し、そのユーザー用の資格情報を独自に作成します。
7. 許可サービスは、要求を許可または拒否します。

(4) 次回の e-Community リモート (ドメイン B) アクセス: WebSEAL 4

1. ユーザーは、WebSEAL 4 (リモート・ドメイン B) により保護されているリソースを要求します。最初にシナリオ 2 が発生している場合は、ブラウザーには、WebSEAL 3 を vouch-for サーバーとして識別するドメイン B e-community Cookie が含まれています。WebSEAL 4 のキャッシュには、このユーザーの資格情報は入っていません。
2. WebSEAL 4 の構成では、e-community 認証が使用可能にされ、MAS の場所が指定されています。しかし、ドメイン B e-community Cookie が存在するため、MAS の場所を指定した WebSEAL 4 構成はオーバーライドされます。この Cookie を使用して、WebSEAL 4 は vouch-for サーバーを識別します。(最初にシナリオ 1 が発生している場合は、ブラウザーに保持されているのはドメイン A の Cookie のみであり、これはドメイン B のサーバーには送られません。代わりに、構成済みの MAS が使用されます。そして、WebSEAL 4 がドメイン B の vouch-for サーバーになります。)
3. 最初にシナリオ 2 が発生している場合、WebSEAL 4 は、ドメイン B の Cookie で識別されているドメイン B vouch-for サーバー (この場合は WebSEAL 3) の特殊 vouch-for URL に、ブラウザーをリダイレクトします。
4. WebSEAL 3 は vouch-for 要求を受け取り、キャッシュ内にこのユーザー用の資格情報を見つけます (資格情報はシナリオ 2 で作成されています)。
5. WebSEAL 3 は、暗号化した vouch-for トークンを添えて、ブラウザーを、最初に要求されていた WebSEAL 4 上の URL にリダイレクトします。(拡張属性の処理の説明については、シナリオ 1 と 2 を参照。)
6. WebSEAL 4 は、トークンを復号し、そのユーザー用の資格情報を独自に作成します。
7. 許可サービスは、要求を許可または拒否します。

(5) 以後の e-Community ローカル (ドメイン A) アクセス: WebSEAL 2

1. ユーザーは、WebSEAL 2 (ドメイン A) に接続して要求を出します。既にシナリオ 3 が発生している場合は、WebSEAL 2 のキャッシュにはこのユーザーの資格情報が入っています。
2. 許可サービスは、要求を許可または拒否します。

e-Community からのログアウト

- ユーザーがブラウザーをクローズしてログアウトした場合は、すべての SSL セッションおよびすべての e-community Cookie がクリアされます。
- ユーザーが /pkmslogout ページを使用してログアウトした場合は、そのドメイン用の SSL セッションおよび e-community Cookie がクリアされます。

e-community Cookie

- e-community Cookie は、1 つの WebSEAL サーバーにより設定されるドメイン固有の Cookie で、ユーザーのブラウザのメモリーに保存され、以後の要求に含めて他の WebSEAL サーバー (同一ドメイン内) に伝送されます。
- ドメイン固有の Cookie には、`vouch-for` サーバーの名前、e-community 識別、`vouch-for` サーバーの場所 (URL) と機能、および存続時間 (タイムアウト) 値が含まれています。Cookie には、ユーザー情報やセキュリティ情報は含まれていません。
- e-community Cookie を使用することにより、参加ドメイン内のサーバーは、ローカルで `vouch-for` 情報を要求することができます。MAS があるドメイン用の e-community Cookie の役割は、それほど重要ではありません。
- Cookie には、WebSEAL 構成ファイル内で設定されている存続時間値が含まれています。この存続時間値は、リモート・サーバーがどれだけの間該当ユーザーの `vouch-for` 情報を提供できるかを指定します。Cookie の存続時間が満了した後は、そのユーザーは認証のために MAS にリダイレクトされます。
- `[e-community-sso]` スタンザ内の `disable-ec-cookie` オプションの値が **yes** の場合、MAS は `vouch-for` トークンを生成できる唯一のサーバーになります。
- ブラウザーがクローズされると、Cookie はメモリーからクリアされます。ユーザーが特定のドメインからログアウトすると、e-community Cookie は空にされます。これにより、Cookie は実質的にはブラウザから削除されることとなります。

vouch-for 要求と応答

e-community `vouch-for` 操作には、特別に構成された 2 つの URL (`vouch-for` 要求および `vouch-for` 応答) を介してアクセスされる専用の機能が必要です。

vouch-for 要求

`vouch-for` 要求が起動されるのは、ユーザーが、自分の資格情報を含んでいないターゲット・サーバー (e-community 用として構成されているもの) のリソースを要求したときです。

このサーバーは、`vouch-for` サーバー (MAS、または e-community Cookie で識別されている代行の `vouch-for` サーバー) にリダイレクトを送ります。

`vouch-for` 要求には以下の情報が含まれています。

`https://vouch-for-server/pkmsvouchfor?ecommunity-name&target-URL`

受信側サーバーは `ecommunity-name` をチェックして、e-community 識別を確認します。そして、受信側サーバーは、`vouch-for` 応答の中の `target-URL` を使用して、最初に要求されたページにブラウザをリダイレクトします。

`pkmsvouchfor` `vouch-for` URL は構成可能です。

以下に例を示します。

`https://mas.dA.com/pkmsvouchfor?companyABC&https://ws5.dB.com/index.html`

注: pkmsvoucher 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

vouch-for 応答

vouch-for 応答は、vouch-for サーバーからターゲット・サーバーへの応答です。

vouch-for サーバーは MAS であるか、MAS ドメインからリモートに位置するドメイン内の代行 vouch-for サーバーです。

vouch-for 応答には以下の情報が含まれています。

`https://target-URL?PD-VFHOST=vouch-for-server&PD-VF=encrypted-token`

PD-VFHOST ラベルは、vouch-for 操作を実行したサーバーを識別します。受信側 (ターゲット) サーバーはこの情報を使用して、vouch-for トークンを復号するために必要な正しい鍵を選択します。PD-VF ラベルは、vouch-for 応答 URL 内の暗号化されたトークンを識別します。

例:

`https://w5.db.com/index.html?PD-VFHOST=mas.dA.com&PD-VF=3qhe9fjkgp...ge56wgb`

vouch-for トークン

クロスドメイン・シングル・サインオンを達成するには、ある種のユーザー識別情報をサーバー間で伝送する必要があります。この機密情報を取り扱う方法として、リダイレクトが使用されます。このリダイレクトに、暗号化された識別情報が URL の一部として組み込まれています。この暗号化されたデータを **vouch-for トークン** と呼びます。

- トークンには、vouch-for の成功または失敗の状況、ユーザーの識別 (成功の場合)、トークンを作成したサーバーの完全修飾名、e-community 識別、および作成日時の値が含まれています。
- 有効な vouch-for トークンの所有者は、このトークンを使用することにより、相手サーバーに対して明示的に認証しなくても、そのサーバーとのセッションおよび資格情報のセットを確立することができます。
- トークンは、認証性を確認できるように、共用 Triple-DES 秘密鍵を使用して暗号化されます。
- 暗号化されたトークン情報は、ブラウザーには保管されません。
- トークンは 1 回だけ渡されます。受信側サーバーはトークンの情報を使用して、自身のキャッシュ内にユーザー資格情報を作成します。サーバーは、このユーザーが同一セッション内で行う以後の要求に、この資格情報を使用します。
- トークンには、WebSEAL 構成ファイルに設定されている存続時間 (タイムアウト) 値があります。この値は、リプレイ・アタックの危険性を回避するために、きわめて短い時間 (秒単位) に設定できます。

e-community シングル・サインオンの構成

このセクションは、以下の基本的なトピックで構成されています。

- 『e-community 構成の要約』
- 646 ページの『e-community の条件と要件』

主な構成タスクは、以下のトピックで説明しています。

1. 647 ページの『e-community 認証を使用可能および使用不可にする』
2. 648 ページの『e-community 名の指定』
3. 648 ページの『「保証」トークンの暗号化』
4. 649 ページの『vouch-for トークン・ラベル名の構成』
5. 650 ページの『マスター認証サーバー (MAS) の指定』
6. 651 ページの『vouch-for URL の指定』
7. 652 ページの『トークンおよび ec-Cookie の存続時間値の構成』

それ以外のトピックは補足的な情報です。

- 652 ページの『トークン作成中の CDMF エラーの処理』
- 653 ページの『非認証アクセスの使用可能化』
- 654 ページの『仮想ホストでの e-community の使用』

e-community 構成の要約

e-community は、以下の条件とガイドラインに従って構成します。

- vouch-for サーバー (MAS または代行 vouch-for サーバー) は、常にトークン作成に責任を持ちます。
- 受信側サーバー (要求されたリソースを持っているサーバー) は、常にトークン消費に責任を持ちます。
- 代行 vouch-for サーバー (MAS ドメインから見てリモートの位置にあるすべてのドメイン用) は、トークン作成およびトークン消費の両方の機能を備えている必要があります。

以下に示す個々の e-community 構成ステップについては、この章の以後のセクションで詳しく説明します。

vouch-for サーバーでのトークン作成機能の構成

このタスクについて

以下に示す個々の e-community 構成ステップについては、この章の以後のセクションで詳しく説明します。

手順

1. e-community 認証を使用可能にして、通信タイプ別にシングル・サインオン要求を処理できるようにする (**e-community-sso-auth**)。
2. すべての参加サーバーに共通する統一 e-community 名を指定する (**e-community-name**)。

3. 組み込みシングル・サインオン・トークン作成モジュールを構成する (**sso-create**)。
4. vouch-for トークンをエンコードおよびデコードするために使用する鍵ファイルを作成する。該当するすべての参加サーバーに鍵格納ファイルをコピーする (**[e-community-domain-keys]** スタンザ)。
5. vouch-for 応答の中で使用するトークン・ラベルを構成する (**vf-argument**)。
6. このサーバーが MAS か MAS でないかを指定する (**is-master-authn-server**)。
7. vouch-for 要求の中で使用する vouch-for URL を指定する (**vf-url**)。
8. トークンおよび e-community Cookie の存続時間値を構成する (**vf-token-lifetime** および **ec-Cookie-lifetime**)。

受信側サーバーでのトークン消費機能の構成

このタスクについて

以下に示す個々の e-community 構成ステップについては、この章の以後のセクションで詳しく説明します。

手順

1. e-community 認証を使用可能にして、通信タイプ別にシングル・サインオン要求を処理できるようにする (**e-community-sso-auth**)。
2. すべての参加サーバーに共通する統一 e-community 名を指定する (**e-community-name**)。
3. 組み込みシングル・サインオン・トークン消費モジュールを構成する (**sso-consume**)。
4. 該当の鍵格納ファイルを割り当てる (**[e-community-domain-keys]** スタンザ)。
5. vouch-for 応答の中で使用するトークン・ラベルを構成する (**vf-argument**)。
6. このサーバーが MAS ではないことを指定する (**is-master-authn-server**)。
7. vouch-for 要求の中で使用する vouch-for URL を指定する (**vf-url**)。
8. トークンおよび e-community Cookie の存続時間値を構成する (**vf-token-lifetime** および **ec-Cookie-lifetime**)。

e-community の条件と要件

- e-community インプリメンテーションでは、e-community に参加するすべてのドメイン内のすべての WebSEAL サーバーに一貫性のある構成が必要です。
- e-community が正しく機能するためには、参加する各 WebSEAL サーバーは、自身の完全修飾ホスト名をクロスドメイン環境内の他の参加サーバーに開示する必要があります。ドメインが含まれていないホスト名が 1 つでもあると、e-community は使用可能にされず、エラー・メッセージが `msg_webseald.log` に記録されます。e-community 環境をセットアップするときは、個々の参加サーバーに関するマシン固有ネットワーク・セットアップにおいて、必ず完全修飾ホスト名を使用してサーバーを識別するように構成してください。
- e-community に参加するすべての WebSEAL サーバー間で、マシン時刻が同期していることが必要です。マシン間の時差が大きいと、サーバー間の認証が失敗することがあります。

- e-community インプリメンテーションは、HTTP と HTTPS の両方でサポートされます。
- 個々の e-community ドメインは、それぞれのユーザー識別およびそれに関連した特権を管理します。クロスドメイン・マッピング機能 (CDMF) API を使用すると、リモート・ドメインにあるユーザーをローカル・ドメイン内の有効なユーザーにマップすることができます。

e-community 内のドメインがグローバル・ユーザー識別を共用している場合は、異なるドメイン内では異なるパスワードを使用することで、ユーザーを区別することができます。ドメイン A とドメイン B でそれぞれ異なるパスワードを使用すれば、両方のドメイン内にユーザー「abc」が存在していて構いません。

- e-community の構成は、参加している各 WebSEAL サーバーの WebSEAL 構成ファイルに設定します。
- 最初に要求された URL が、MAS (または vouch-for サーバー) からブラウザーにリダイレクトされて戻されない場合は、ブラウザーが Microsoft Internet Explorer である場合、ページ・キャッシングに問題がある可能性があります。そのような問題がある場合は、保管されたページの新しいバージョンの有無を必ずチェックするようにブラウザーを構成します。

Tools > Internet Options > General > Temporary Internet Files > Settings

- MAS サーバーを、別の参加 WebSEAL インスタンスの同じインターフェース (IP アドレス) 上に構成しないでください。
- ある種の WebSEAL 構成では、マシン・ホスト名を完全修飾ホスト名で記述することが必要とされるため、ご使用のシステムおよびネットワークがマシン名を完全修飾ホスト名に解決できることを、確認する必要があります。完全修飾ホスト名を使用すれば、例えば、複数 WebSEAL インスタンスの場合のように、1 つのマシンにつき多数のホスト名 (IP アドレス) を使用することができます。

e-community 環境でのマシン名の解決

e-community は、マシン自体がマシン名を解決するように十分に構成されていないという理由で、WebSEAL の始動時に使用不可にすることができます。

このタスクについて

WebSEAL があるマシンは、IP アドレスを完全に解決できなければなりません。この機能性はオペレーティング・システムに固有ですので、手順について説明することは、この資料の役割を超えています。ご使用のシステムが正しい能力を持っているか確信がない場合は、必ず、システム・アドミニストレーターに相談してください。

e-community 認証を使用可能および使用不可にする

このタスクについて

e-community 認証方式を使用可能または使用不可にする、および通信タイプ別にシングル・サインオン要求を処理できるようにするには、WebSEAL 構成ファイルの [e-community-sso] スタンザにある e-community-sso-auth スタンザ・エントリーを使用します。

手順

- e-community 認証方式を使用可能にするには、「http」、「https」、または「both」を入力します。

「http」、「https」、および「both」の値は、e-community 参加サーバーが使用する通信のタイプを指定します。

- e-community 認証方式を使用不可にするには、「none」を入力します。

「none」の値は、該当サーバーについて e-community を使用不可にします。デフォルトの設定は「none」です。

例えば、次のとおりです。

```
[e-community-sso]
e-community-sso-auth = https
```

注: WebSEAL 構成ファイルへの変更内容を活動化するには、WebSEAL サーバーを停止し、再始動する必要があります。このセクションの、該当する構成ステップをすべて完了してから WebSEAL を再始動してください。

e-community 名の指定

このタスクについて

e-community-name スタンザ・エントリーは、すべての参加ドメイン内のすべての参加サーバーに共通する e-community の統一名を指定します。例えば、次のとおりです。

```
[e-community-sso]
e-community-name = companyABC
```

e-community-name の値は、e-community に参加するすべてのドメイン内のすべての WebSEAL で同じでなければなりません。

「保証」トークンの暗号化

このタスクについて

注: 鍵格納ファイルの配布は、Security Access Manager の e-community プロセスの中では行われません。アドミニストレーターは、各参加サーバーに、手動で安全に鍵をコピーする必要があります。

e-community に参加するサーバー間で送信されるトークンを保護するために使用する鍵ファイルの名前は、**[e-community-domain-keys]** スタンザで指定します。

```
[e-community-domain-keys]
domain-name = keyfile-name
domain-name = keyfile-name
```

e-community ドメイン鍵

e-community に参加するサーバー間で交換されるトークンを暗号化および復号するために必要な鍵格納ファイルは、**[e-community-domain-keys]** スタンザで指定します。

サーバーごとの完全修飾ドメイン名と関連付けられている鍵ファイルの名前を指定する必要があります。

次の例では、MAS (ドメイン A) に、2 つのリモート・ドメイン (**dB** および **dC**) と通信するための鍵格納ファイルと、ドメイン A の中の他のサーバーと通信するための鍵が与えられます。

```
[e-community-domain-keys]
dA.com = key.fileA-A
dB.com = key.fileA-B
dC.com = key.fileA-C
```

この例では、key.fileA-A は、ドメイン A 内のすべてのサーバー間で使用する鍵格納ファイルを示しています。

key.fileA-B は、ドメイン A とドメイン B の間で使用する鍵格納ファイルを示しています。

key.fileA-C は、ドメイン A とドメイン C の間で使用する鍵格納ファイルを示しています。

各リモート・サーバーは、MAS が使用する該当鍵格納ファイルのコピーを持っている必要があります。MAS (ドメイン A) との間でトークンを交換するためには、ドメイン B の中のすべてのサーバーが key.fileA-B のコピーを持っている必要があります。

```
[e-community-domain-keys]
dA.com = key.fileA-B
```

MAS (ドメイン A) との間でトークンを交換するためには、ドメイン C の中のすべてのサーバーが key.fileA-C のコピーを持っている必要があります。

```
[e-community-domain-keys]
dA.com = key.fileA-C
```

ドメイン A 内にあって、MAS が提供する認証サービスを使用するすべてのサーバーは、key.fileA-A ファイルのコピーを持っている必要があります。

```
[e-community-domain-keys]
dA.com = key.fileA-A
```

次の例では、key.fileB-B は、domainB のすべてのサーバー間で使用される鍵格納ファイルを示しています。また、key.fileC-C は、domainC のすべてのサーバー間で使用される鍵格納ファイルを示しています。

```
[e-community-domain-keys]
dB.com = key.fileB-B
dC.com = key.fileC-C
```

vouch-for トークン・ラベル名の構成

このタスクについて

リダイレクトされた要求には、シングル・サインオン・トランザクションに使用する認証情報が、その要求に対する暗号化されたトークン照会ストリング引数として組み込まれます。このトークン・ストリングには、要求を識別するための名前 (ラベル) が必要です。ラベル名は、受信側 WebSEAL サーバーに対してこの要求を一

意的に識別するもので、この要求がシングル・サインオン・トークン消費モジュールにより処理されるシングル・サインオン要求であることを示します。

このトークン・ラベルは、シングル・サインオン機能に参加する両方の WebSEAL サーバーで構成する必要があります。

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

手順

トークン・ラベルを構成するには、WebSEAL 構成ファイルの **[e-community-sso]** スタンザにある **vf-argument** スタンザ・エントリーを使用します。例えば次のように指定します (デフォルトの場合)。

```
[e-community-sso]
vf-argument = PD-VF
```

マスター認証サーバー (MAS) の指定

このタスクについて

e-community 内のどのサーバー・マシンにマスター認証サーバー (MAS) の機能を持たせるかを指定する必要があります。また、個々のサーバー・マシンが MAS ではないことも指定する必要があります。

is-master-authn-server

サーバーが MAS であるかどうかを指定するには、**is-master-authn-server** スタンザ・エントリーを使用します。指定できる値は、「yes」または「no」です。

以下に例を示します。

```
[e-community-sso]
is-master-authn-server = yes
```

複数の WebSEAL をマスター認証サーバーとして構成して、ロード・バランサーの背後に配置することができます。この構成では、e-community 内の他のすべてのサーバーは、ロード・バランサーを MAS として「認識」します。

構成しようとしているサーバーが MAS ではない場合は、**master-authn-server** を使用して、そのサーバーに対して MAS の場所を指定します。

master-authn-server

is-master-authn-server スタンザ・エントリーを「no」に設定した場合は、このスタンザ・エントリーをアンコメントして指定する必要があります。このスタンザ・エントリーには、MAS の完全修飾ドメイン名を指定します。

以下に例を示します。

```
[e-community-sso]
master-authn-server = mas.dA.com
```


また、MAS が使用する HTTP および HTTPS listen ポートの値が、デフォルト値 (HTTP の場合はポート 80 で、HTTPS の場合はポート 443) と違う場合は、これらのポートも指定することができます。

master-http-port

e-community-sso-auth により HTTP e-community 認証が使用可能にされ、マスター認証サーバーが標準 HTTP ポート (ポート 80) 以外のポートで HTTP 要求を listen する場合は、**master-http-port** スタンザ・エントリーにその標準外ポートを指定します。このサーバーがマスター認証サーバーである場合は、このスタンザ・エントリーは無視されます。デフォルトでは、このスタンザ・エントリーは使用不可にされています。

```
[e-community-sso]
master-http-port = port-number
```

master-https-port

e-community-sso-auth により HTTPS e-community 認証が使用可能にされ、マスター認証サーバーが標準 HTTPS ポート (ポート 443) 以外のポートで HTTPS 要求を listen する場合は、**master-https-port** スタンザ・エントリーにその標準外ポートを指定します。このサーバーがマスター認証サーバーである場合は、このスタンザ・エントリーは無視されます。デフォルトでは、このスタンザ・エントリーは使用不可にされています。

```
[e-community-sso]
master-https-port = port-number
```

vouch-for URL の指定

このタスクについて

vf-url

vf-url スタンザ・エントリーは vouch for URL を指定します。値はスラッシュ (/) で始まっていなければなりません。デフォルト値は **/pkmsvouchfor** です。

例:

```
[e-community-sso]
vf-url = /pkmsvouchfor
```

次のように拡張 URL を指定することもできます。

```
vf-url = /ecommA/pkmsvouchfor
```

拡張 URL は、クライアントが WebSEAL サーバーではない MAS と通信するとき使用されます。**vf-url** を使用すると、クライアントは、カスタマイズされたトークン消費モジュールなどの専門化された認証ライブラリーを持っている MAS へのアクセスを指定できます。

注: pkmsvouchfor 管理ページは WebSEAL サーバーに対する管理コマンドです。オブジェクト・スペースには表示されず、ポリシーを付加できません。

トークンおよび ec-Cookie の存続時間値の構成

このタスクについて

vf-token-lifetime

vf-token-lifetime スタンザ・エントリーは、vouch-for トークンの存続時間タイムアウト値 (秒数) を設定します。この値は、Cookie に刻時されている作成時刻に照らしてチェックされます。デフォルト値は 180 秒です。参加サーバー間の時間のずれ (クロック・スキュー) を考慮に入れる必要があります。

以下に例を示します。

```
[e-community-sso]
vf-token-lifetime = 180
```

ec-Cookie-lifetime

ec-Cookie-lifetime スタンザ・エントリーは、e-community ドメイン Cookie の最大存続時間 (分) を指定します。デフォルト値は 300 分です。

以下に例を示します。

```
[e-community-sso]
ec-cookie-lifetime = 300
```

参加ドメイン間の時間のずれ (クロック・スキュー) を考慮しなければなりません。クロック・スキューとは、各ドメイン内の関係のあるサーバー上のシステム時刻に差があることを意味します。この差が、**vf-token-lifetime** の値に近づくと、トークンの有効存続時間が大幅に減ります。この差が **vf-token-lifetime** の値を超えると、あるドメインにあるトークンは、その他のドメインにとって有効ではなくなります。アドミニストレーターは、**vf-token-lifetime** を適切に調整しなければなりません。ただし、クロック・スキューにより、**vf-token-lifetime** に大きな値を設定する必要があるときは、リプレイ・アタックのリスクが増大します。そのような場合は、アドミニストレーターは、各ドメインの関連するサーバー上のシステム時刻を同期しなければなりません。

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」を参照してください。

トークン作成中の CDMF エラーの処理

このタスクについて

e-community トークン作成中に **ssocreate** モジュールは CDMF ライブラリーを呼び出し、トークンに組み込む拡張属性を取得します。拡張属性 (ユーザーを詳しく記述する属性) は、宛先サーバーでのユーザーの ID マッピングを正常に行うために必要となることがあります。CDMF API は **cdmf_get_usr_attributes** 呼び出しを使用して拡張属性を取得します。

cdmf_get_usr_attributes 呼び出しで必要な情報を取得できず、エラーが戻されることがあります。このような場合、トークン作成プロセスの後続の動作を制御するに

は、[cdsso] スタンザの **propagate-cdmf-errors** スタンザ・エントリーを使用します。このスタンザ・エントリーに指定できる値は「yes」および「no」です。

値「no」(デフォルト) を指定すると、CDMF が属性を取得できずエラーを戻す場合でも、トークン作成プロセスを続行できます。

値「yes」を指定すると、CDMF が属性を取得できずエラーを戻す場合に、トークン作成プロセスが強制的に終了します。

例:

```
[cdsso]
propagate-cdmf-errors = no
```

非認証アクセスの使用可能化

このタスクについて

アドミニストレータは、非認証ユーザーに対し、e-community SSO 参加サーバー上の無保護リソースへのアクセスを許可するかどうかを制御できます。認証ユーザーに対しこのアクセスを許可すると、参加サーバーは、ユーザーがマスター認証サーバーを使用して認証を受けることを必要とせずに、無保護リソースを使用できるようにします。このポリシーが構成されていると、参加サーバーは、クライアントが保護リソースにアクセスを要求したときにのみ、マスター認証サーバーにリダイレクトします。

このポリシーは WebSEAL 構成ファイルに以下のように設定されます。

```
[e-community-ssso]
ecssso-allow-unauth = {yes|no}
```

ecssso-allow-unauth が「yes」に設定されていると、非認証アクセスが使用可能になります。デフォルト設定は「yes」です。

ecssso-allow-unauth が「no」に設定されていると、非認証アクセスが使用不可になります。この場合、クライアントは、e-community SSO 参加サーバー上のリソース(保護リソースまたは無保護リソースであるかに関係なく) へのアクセスを要求するときに、マスター認証サーバーを通して認証を受けなければなりません。

注: WebSEAL バージョン 5.1 では、デフォルトの振る舞いに変更されました。これより前のバージョンでは、非認証アクセスは使用不可でした。WebSEAL の古いバージョンとの後方互換性のある振る舞いを続けるには、ecssso-allow-unauth = no を設定します。

vouch-for トークンの生成機能の制限

このタスクについて

WebSEAL 構成ファイルには、vouch-for トークンの生成機能を MAS に限定できるオプションのパラメーターが含まれています。**e-community-ssso** スタンザ内の **disable-ec-cookie** オプションは、デフォルトで no に設定されています。このオプションの値を yes に変更すると、e-community Cookie が使用不可になり、vouch-for トークンの生成が MAS にのみ許可されます。この場合、シングル・サインオン・

プロセスでは常に MAS が使用されるため、MAS は e-community 内でサインオンしているすべてのホストを検出できます。このオプションは、カスタマイズされた ECSSO サインオフ・プロセスを作成しようとしているお客様に役立ちます。カスタマイズされたシングル・サインオフ・プロセスについては、『pkmslogout-nomas を使用したログアウト』を参照してください。

認証失敗時の動作の構成

認証されていないユーザーが MAS へのログインに失敗すると (誤ったパスワードを入力した場合など)、MAS はエラーを含む vouch-for トークンを生成します。さらにデフォルト構成の場合は、Web ブラウザーを要求元ホストにリダイレクトして戻します。要求元ホストは、vouch-for トークン内にエラーを検出すると、通常はローカル・ログインを要求します。WebSEAL 構成ファイルの **e-community-sso** スタンザにある **handle-auth-failure-at-mas** オプションを使用すると、管理者は認証失敗時の動作を構成できます。 **handle-auth-failure-at-mas** が **yes** に設定されている場合、MAS は Web ブラウザーを要求元ホストにリダイレクトして戻さずに、ログイン失敗を直接処理します。この場合 MAS では、認証が成功するまで vouch-for トークンを生成しません。

pkmslogout-nomas を使用したログアウト

すべてのホストが MAS でサインオフされるシングル・サインオフ・プロセスを構築および実装しようとしているお客様を支援するために、WebSEAL は pkmslogout-nomas 管理ページを使用するように構成できます。ECSSO が構成されている場合、pkmslogout-nomas 管理ページを **pkmslogout** コマンドの代わりとして使用して、現在のホスト上のセッションからログアウトできます。以下に例を示します。

`https://www.example.com/pkmslogout-nomas`

`/pkmslogout` 管理ページを使用すると、ローカル・システムからログアウトしたクライアントはリダイレクトされて、MAS 上で別のログアウトを実行します。

`/pkmslogout-nomas` 管理ページは `/pkmslogout` 管理ページとまったく同様に作動します。例外は、ユーザーのセッションをログアウトするために、Web ブラウザーを MAS ホストの `/pkmslogout` 管理ページにリダイレクトしない点です。このため、MAS からログアウトを連鎖的に実行できます。例えば、MAS システムにカスタム・ログアウト・ページを配置し、そのページに MAS 以外のすべてのシステムにある `/pkmslogout` ページへの非表示リンクを格納して、コミュニティ内の全システムからユーザーをログアウトすることができます。

仮想ホストでの e-community の使用

564 ページの『仮想ホストを使用した e-community シングル・サインオン』を参照してください。

ECSSO 用の拡張属性

このセクションは、以下のトピックで構成されています。

- 『トークンに追加する拡張属性』
- 656 ページの『トークンから抽出する拡張属性』

トークンに追加する拡張属性

WebSEAL 構成ファイルで、ユーザー資格情報にある拡張属性を、クロスドメイン・シングル・サインオン・トークンに追加するように指定できます。拡張属性は、ユーザー資格情報が作成されるときに拡張属性リストに追加されるユーザー ID に関する情報で構成されています。拡張属性は、外部認証 C API サービスを含む、多くの認証メカニズムによって追加できます。外部認証 C API モジュールは、例えば、Security Access Manager にとって外部であるレジストリーからユーザー情報を入手するために使用できます。

ユーザーはこの設定を使用して、e-community シングル・サインオン・トークンの内容をカスタマイズすることができます。この機能を使用すると、トークンの内容を、宛先ドメインのニーズに一致させるように調整できます。この機能を使用して属性をトークンに追加するときは、宛先ドメインのサーバーの WebSEAL 構成ファイルも構成する必要があります。宛先サーバーの場合、各属性の処理（抽出または無視）を指定するには、スタンザ `[ecssso-incoming-attributes]` を使用します。

名前を使用して拡張属性を指定することも、複数の属性名に一致するパターンを宣言することもできます。標準の Security Access Manager ワイルドカード・マッチング文字を使用できます。サポートされるワイルドカード・パターン・マッチング文字のリストについては、77 ページの『サポートされているワイルドカード・パターン・マッチング文字』を参照してください。

各エントリーには、トークンが対象とするドメインの名前が割り当てられます。各ドメインの名前またはパターンを指定するエントリーを複数含めることができます。

構文は次のようになります。

```
[ecssso-token-attributes]
domain_name = pattern1
domain_name = pattern2
...
domain_name = patternN
<default> = pattern1
<default> = pattern2
...
<default> = patternN
```

<default> エントリーはオプションです。WebSEAL がドメイン名に一致するエントリーを見つけないときは、WebSEAL は <default> エントリーを探します。構成ファイルに <default> エントリーがある場合は、WebSEAL は、現在のドメインの、割り当てられた属性パターンを使用します。ストリング <default> はキーワードであるので、< 文字と > 文字を含めて、上に示したとおりに指定しなければなりません。

例えば、2 つのドメイン、example1.com と example2.com との間に、e-community シングル・サインオン・ソリューションを作成するとします。ユーザーは example1.com にログインしますが、ユーザー・セッション中に、example2.com にリダイレクトすることができます。ご使用のデプロイメントには、各ユーザー資格情報に情報を挿入するカスタマイズされた外部認証 C API モジュールが組み込まれています。この情報には、固定の名前属性「job_category」と可変数の属性が入っており、それぞれに、「my_ext_attr_」という文字が先頭に付いています。この情報は、クロスドメイン・トークンに追加する必要があります。構成ファイル・エントリーは、次のようになります。

```
example2.com = job_category
example2.com = my_ext_attr_*
```

トークンから抽出する拡張属性

WebSEAL 構成ファイルで、e-community シングル・サインオン・トークンに追加された拡張属性を、トークン消費モジュールがどのように処理するかを指定することができます。

属性は、抽出または無視できます。ある場合は、宛先ドメインの中のサーバーに属性を作成する方法がないために、属性を抽出しなければなりません。また、他の場合では、宛先ドメインの中のサーバーが独立処理を使用して同じ拡張属性を収集できるので、トークンを抽出する必要がありません。例えば、属性は、宛先サーバーのシステム時刻を反映しなければならないタイム・スタンプを反映できます。

組み込みトークン消費モジュールでは、トークンから抽出された属性は、クロスドメイン・マッピング・フレームワーク・ライブラリーにパススルーされます。デフォルトのクロスドメイン・マッピング・フレームワーク・ライブラリーは、属性を、ユーザー資格情報に直接パススルーします。カスタマイズされたクロスドメイン・マッピング・フレームワーク・ライブラリーは、必要に応じて属性を操作してからユーザー資格情報に渡すことができます。

エントリーの構文は、次のようになります。

```
[ecssso-incoming-attributes]
attribute_pattern = {preserve|refresh}
```

通常、拡張属性 (*attribute_pattern*) の名前は、トークンを生成する WebSEAL サーバーの構成ファイルの **[ecssso-token-attributes]** スタンザに指定されている属性の名前に一致します。使用できる値は、次のキーワードのいずれかです。

- preserve

attribute_pattern に一致するすべての属性を抽出します。

- refresh

attribute_pattern に一致する属性を抽出しません。

[ecssso-incoming-attributes] の中にあるエントリーに一致しないトークン内の拡張属性は保存 (抽出) されます。

スタンザの中のエントリーの順序は重要です。属性名に一致する最初のエントリーが使用されます。その他のエントリーは無視されます。例えば、my_special_attr1

という名前の属性は抽出したいが、`my_special_attr_` という接頭部が付いたその他のすべてのエントリは無視したい場合、スタンザのエントリは次のようになります。

```
[ecssso-incoming-attributes]
my_special_attr1 = preserve
my_special_attr_* = refresh
```

上記の 655 ページの『トークンに追加する拡張属性』で示した例を使用すると、`example2.com` ドメインで操作されるサーバーの WebSEAL 構成ファイルのエントリは、次のようになります。

```
[ecssso-incoming-attributes]
job_category = preserve
my_cdas_attr_1 = preserve
my_cdas_attr_* = refresh
```

この例では、属性 `job_category` および `my_cdas_attr_1` がトークンから抽出されます。接頭部 `my_cdas_attr_` が付いたその他の属性は無視されます。

e-community シングル・サインオン用のトークンの UTF-8 エンコード

e-community シングル・サインオンに使用するトークン内のストリングに UTF-8 エンコードを使用するか否かは、WebSEAL 構成ファイルで指定します。

```
[e-community-sso]
use-utf8 = {yes|no}
```

デフォルト値は「yes」です。

use-utf8 が「no」に設定されていると、ストリングはローカル・コード・ページを使用してエンコードされます。古い (バージョン 5.1 より前の) WebSEAL サーバーを使用して e-community シングル・サインオンをインプリメントするときはこの値を使用します。5.1 より前のバージョンの WebSEAL サーバーでは、トークンに UTF-8 エンコードを使用していません。これらの古いサーバーが組み込まれている環境をデプロイするときは、UTF-8 エンコードを使用しないように WebSEAL サーバーを構成してください。この設定は、後方互換性を保つために必要です。

WebSEAL による UTF-8 エンコードのサポートについて詳しくは、65 ページの『UTF-8 でのマルチロケール・サポート』を参照してください。

第 35 章 シングル・サインオフ

Junction バックエンド・サーバー上にある複数の保護された Web リソースからシングル・サインオフを開始するように、WebSEAL を構成できます。

シングル・サインオフ機能は、以下のセクションで詳しく説明します。

- 『シングル・サインオフ機能の概要』
- 660 ページの『シングル・サインオフの構成』
- 660 ページの『シングル・サインオフ要求および応答の仕様』

シングル・サインオフ機能の概要

セッションの終了時に HTTP 要求を事前定義アプリケーションに送信するように WebSEAL を構成できます。これらの要求を受信するアプリケーションは、Junction バックエンド・サーバーにある任意の関連セッションを終了できます。

セッションが終了すると、WebSEAL は、自身が管理するセッションおよびセッション・データを削除します。WebSEAL は、バックエンド・アプリケーションによって作成および管理されるセッションを制御することはできません。そのため、対応する WebSEAL セッションが終了した後でも、バックエンド・サーバー・セッションはアクティブ状態のままです。WebSEAL には、WebSEAL 内でセッションが終了したときにバックエンド・サーバー上のセッションを除去するメカニズムがあります。

シングル・サインオフを実現するため、WebSEAL セッションが破棄されるたびに、WebSEAL は構成済みのシングル・サインオフ URI に要求を送信します。バックエンド・サーバー上のアプリケーションは、要求内に示されている情報を使用して、失効したセッションを終了できます。

WebSEAL セッションを終了できるメカニズムは、以下の 4 つです。

- **pkmslogout** にアクセスすることによるユーザー要求。
- セッション・タイムアウト。
- EAI セッション終了コマンド。
- **pdadmin** ツールからのセッション終了コマンド。

注: Session Management Server (SMS) 環境でこの機能を使用すると、終了したセッションを含む各 WebSEAL サーバーからの個別のサインオフ要求が生成されます。したがって、SMS 環境内のシングル・サインオフ・アプリケーションは、WebSEAL サーバーごとに 1 つの単一セッションに対して複数のサインオフ要求を処理する必要があります。

シングル・サインオフの構成

このタスクについて

シングル・サインオフ要求を受信する URI を指定することにより、WebSEAL でシングル・サインオフを使用可能にすることができます。**[acnt-mgt]** スタンザの **single-signoff-uri** エントリーを、各シングル・サインオフ・アプリケーションを参照するように構成します。このリソースを仮想ホスト **junction** に配置することはできず、サーバー相対 URI を指定する必要があります。以下に例を示します。

```
[acnt-mgt]
single-signoff-uri = /applications/signoff
```

WebSEAL は、WebSEAL セッションが終了するたびに、指定された各 URI に要求を送信します。それぞれの要求には、指定されたリソースの **junction** に対する構成済みのヘッダーと **Cookie** が含まれます。シングル・サインオフ・リソースは、この情報を使用してバックエンド・サーバーのすべてのセッションを終了する責任を負います。

WebSEAL は、HTTP 状況コード **200 OK** を含む応答を受信することを想定しています。それ以外の状況コードが応答に含まれる場合、WebSEAL はエラーをログに記録します。

注: シングル・サインオフは複数の **junction** サーバーで実行できます。複数の **single-sign-off-uri** エントリーを構成すると、要求が複数の URI に送信されます。

シングル・サインオフ要求および応答の仕様

シングル・サインオフ機能を構成する場合、シングル・サインオフ要求および応答は、以下の仕様に従った形式になります。

シングル・サインオフ要求

WebSEAL は、**single-signoff-uri** 構成エントリーで指定されたシングル・サインオフ・リソースに要求を送信します。シングル・サインオフ要求には以下が含まれます。

- HTTP GET メソッド。
- シングル・サインオフ・リソースがある **Junction** ポイント用に構成された任意の **Cookie** およびヘッダー。

シングル・サインオフ応答

WebSEAL は、HTTP 状況コード **200 OK** の応答を予期します。その他の状況コードはすべて、エラーとして記録されます。WebSEAL は、応答内の本体およびその他すべてのヘッダーを無視します。

第 10 部 デプロイメント

第 36 章 WebSEAL インスタンスのデプロイメント

この章では、1 つ以上の WebSEAL のデプロイ手法について説明します。

- 『WebSEAL インスタンスの構成の概要』
- 666 ページの『WebSEAL インスタンスの構成タスク』
- 668 ページの『ロード・バランシング環境』

WebSEAL インスタンスの構成の概要

このセクションは、以下のトピックで構成されています。

- 『WebSEAL インスタンスの構成計画』
- 665 ページの『WebSEAL インスタンス構成値の例』
- 666 ページの『各 WebSEAL インスタンス用の固有の構成ファイル』

WebSEAL インスタンスの構成計画

WebSEAL インスタンスは、固有の構成ファイルと `listen` ポートを持つ固有 WebSEAL サーバー・プロセスです。WebSEAL のデプロイでは、複数の WebSEAL インスタンスがサポートされています。

WebSEAL インスタンスを構成するには、Security Access Manager デプロイメントについていくつかの情報を収集する必要があります。

以下の設定値はすべて、別の指示がないかぎり、必須です。

- アドミニストレーター名およびパスワード

Security Access Manager 管理ユーザーの認証の詳細。デフォルトで、これは **sec_master** ユーザーです。WebSEAL インスタンスを構成するには、管理ユーザーのアクセス許可が必要です。

- ドメイン

Security Access Manager ドメイン。

- ホスト名

Security Access Manager ポリシー・サーバーがアプライアンスの接続に使用するホスト名。このホスト名に対応するアドレスは、アプライアンスの管理インターフェースのアドレスに一致している必要があります。有効な値には、任意の有効なホスト名または IP アドレスが含まれます。

- インスタンス名

WebSEAL インスタンスを識別する固有の名前。1 つのアプライアンスに複数の WebSEAL インスタンスをインストールできます。それぞれのインスタンスに固有の名前がなければなりません。

インスタンス名に有効な文字には、英数字 ([A-Z][a-z][0-9])、およびアンダースコア (_)、ハイフン (-)、ピリオド (.) が使用できます。 . その他の文字は無効になります。

名前の例: web1、web2、web_3、web-4、web.5

また、インスタンス名は、 **pdadmin server list** コマンド時の完全サーバー名のリスト方法に影響を与えます。このコマンドでは、完全サーバー名の形式は次のようになります。

instance_name-webseald-host_name

例えば、diamond という名前のホストにインストールされている web1 の *instance_name* の完全サーバー名は次のようになります。

web1-webseald-diamond

- **listen ポート**

このポートを介して、WebSEAL インスタンスは、Security Access Manager のポリシー・サーバーと通信します。デフォルトのポート番号は 7234 です。このポート番号は、それぞれの WebSEAL インスタンスに対して固有でなければなりません。

デフォルトのポートは、通常、デフォルトの (最初の) WebSEAL インスタンスで使用されます。インストールでは、ポート番号を次の使用可能なポートに自動的に増分します。必要であれば、このポート番号を変更できます。

1024 より大のポート番号だけが有効です。他の目的に使用されていないポートを選択してください。

- **プライマリー・インターフェースの IP アドレス**

WebSEAL インスタンスの固有 IP アドレス。WebSEAL サーバーは、着信要求時にこの IP アドレスで listen します。また、各 WebSEAL インスタンスに、固有の HTTP および HTTPS ポートを割り当てる必要があります。

- **HTTP プロトコルおよび HTTP ポート**

HTTP プロトコルを使用してユーザー要求を受け入れるかどうかを指定します。HTTP を使用可能にする場合、ユーザーはポート番号を割り当てる必要があります。デフォルトのポート番号は 80 です。このポートは、デフォルトの (最初の) インスタンスで使用されます。このポートが使用不可の場合、インストールでは、次の使用可能なポートに自動的に増分されます。

- **HTTPS プロトコルおよび HTTPS ポート**

HTTPS プロトコルを使用してユーザー要求を受け入れるかどうかを指定します。HTTPS を使用可能にする場合、ユーザーはポート番号を割り当てる必要があります。デフォルトのポート番号は 443 です。このポートは、デフォルトの (最初の) インスタンスで使用されます。このポートが使用不可の場合、インストールでは、次の使用可能なポートに自動的に増分されます。

- **ユーザー・レジストリー - SSL 通信**

WebSEAL は、認証手順の実行中に LDAP サーバーと通信します。LDAP サーバーとの通信に SSL を使用することはオプションです。ただし、セキュリティ

上の理由から、すべての実動デプロイメントで SSL を使用することをお勧めします。テスト環境またはプロトタイピング環境で、一時的に SSL の使用を使用不可にすることは可能です。

WebSEAL インスタンスと LDAP レジストリー・サーバーとの間でセキュア SSL 通信を使用する場合は、この目的のための LDAP SSL 鍵格納ファイルを使用する必要があります。これは、LDAP クライアントのインストールの際に作成され、配布された鍵格納ファイルです。初期 WebSEAL インスタンスが、LDAP とのセキュア SSL 通信を使用するようにセットアップされている場合は、複数のインスタンスが同じ鍵格納ファイルを使用することができます。

WebSEAL と LDAP サーバーとの間の SSL 通信を使用可能にするときは、以下の情報を入力しなければなりません。

– 鍵ファイル名

LDAP SSL 証明書が入るファイルです。

– 証明書ラベル

LDAP クライアントの証明書ラベルです。これはオプションです。クライアント・ラベルが指定されないときは、鍵格納ファイルに入っているデフォルトの証明書が使用されます。鍵格納ファイルに複数の証明書が入っており、使用される証明書がデフォルトの証明書でない場合は、クライアント・ラベルを指定してください。

– ポート

LDAP サーバーとの通信に使用されるポート番号。デフォルトの LDAP サーバー・ポート番号は 636 です。

WebSEAL インスタンス構成値の例

次の表には、WebSEAL インスタンスの設定値の例のセットが示されています。この設定値の例は、この章にある以降の構成セクションの構成コマンド・サンプルで使用されます。

設定	値
Security Access Manager アドミニストレータ 一名	sec_master
Security Access Manager アドミニストレータ ・パスワード	mypassw0rd
Security Access Manager ドメイン	domainA
ホスト名	diamond.subnet2.ibm.com
インスタンス名	web1
listen ポート	7235
プライマリー・インターフェースの IP アド レス	1.2.3.5
HTTP を有効にする	yes
HTTP ポート	81
HTTPS を有効にする	yes

設定	値
HTTPS ポート	444
SSL を使用可能にする	yes
鍵ファイル名	client.kdb
証明書ラベル	(none)
ポート	636

各 WebSEAL インスタンス用の固有の構成ファイル

それぞれの WebSEAL インスタンスごとに、固有の WebSEAL 構成ファイルが作成されます。構成ファイルの名前にはインスタンス名が組み込まれています。

次の形式が使用されます。

`webseald-instance_name.conf`

新たに作成されたインスタンス固有構成ファイルは、新規 WebSEAL インスタンスと内部 Security Access Manager サーバー (ポリシー・サーバーなど) の間の SSL 通信用として、自動的に構成されます。

また、新たに作成されたインスタンス固有構成ファイルは、初期 WebSEAL サーバーのサーバー証明書を使用してクライアント・ブラウザを認証するように、自動的に構成されます。

WebSEAL インスタンスの構成タスク

タスク:

- 『WebSEAL インスタンスの追加』
- 667 ページの『WebSEAL インスタンスの除去』

WebSEAL インスタンスの追加

このタスクについて

WebSEAL インスタンスを追加するには、以下のステップのそれぞれを実行してください。

手順

1. 構成を計画します。次のワークシートに記入してください。各設定に適切な値を決定するには、663 ページの『WebSEAL インスタンスの構成計画』を参照してください。

表 43. WebSEAL インスタンスを追加するためのワークシート

設定	値
アドミニストレーター名	
アドミニストレーター・パスワード	
ホスト名	
インスタンス名	

表 43. WebSEAL インスタンスを追加するためのワークシート (続き)

設定	値
listen ポート	
HTTP を有効にする	yes または no
HTTP ポート	
HTTPS を有効にする	yes または no
HTTPS ポート	
SSL を使用可能にする	yes または no
証明書ラベル	
鍵ファイル名	
ポート	

- 構成済みの WebSEAL インスタンスがすべて実行していることを確認してください。これによって、ポートの使用に関して、サーバー間に起こりうる Conflict を避けることができます。
- LMI で「リバース・プロキシ」管理ページの「新規」をクリックして、新しい WebSEAL インスタンスを作成します。
- 「インスタンス」、「IBM Security Access Manager」、「トランスポート」および「ユーザー・レジストリー」の各タブに表示されるフィールドの設定を指定します。
- 「終了」をクリックしてください。新しいインスタンスが正常に構成されたことを示すメッセージが表示されます。
- 新しいインスタンスが LMI で実行されていることを確認してください。「リバース・プロキシ」管理ページにインスタンスの状態が「開始済み」であることが示されます。

WebSEAL インスタンスの除去

このタスクについて

WebSEAL インスタンスの構成を除去するには、以下のステップを実行してください。

手順

- 以下の情報を集めます。
 - インスタンス名
 - アドミニストレーター ID
 - アドミニストレーター ID パスワード
- LMI の「リバース・プロキシ」管理ページで、削除するインスタンスを選択します。
- 「削除」をクリックします。「リバース・プロキシ・インスタンスの削除」ウィンドウが表示されます。
- 選択したインスタンスの管理者の認証詳細情報を入力します。

5. インスタンスを削除するには、「リバース・プロキシ・インスタンスの削除」ウィンドウで「削除」をクリックします。インスタンスが正常に削除されたことを示すシステム通知が表示されます。

ロード・バランシング環境

このセクションは、以下のトピックで構成されています。

- 『フロントエンド WebSEAL サーバーの複製』
- 669 ページの『login_success 応答のコントロール』

フロントエンド WebSEAL サーバーの複製

このタスクについて

注: 以下の情報は、旧バージョンの Security Access Manager で使用していた以前の **pdadmin server modify baseurl** コマンドに関する情報と置き換えて使用してください。

負荷の大きい環境では、フロントエンド WebSEAL サーバーの複製を作成して、ロード・バランシングとフェイルオーバーの能力を高めると効果的です。フロントエンド WebSEAL サーバーを複製する場合は、複製した各サーバーに、Web スペース、junction データベース、および dynurl データベースの正確なコピーを持たせる必要があります。

このバージョンの Security Access Manager では、手動の構成手順によりフロントエンド WebSEAL サーバーを複製できます。したがって、このタスクには **pdadmin** コマンドは使用しないことになりました。

以下の例では、「WS1」は基本 WebSEAL サーバー・マシンのホスト名です。

「WS2」は複製 WebSEAL サーバー・マシンのホスト名です。

1. WS1 サーバー・マシンと WS2 サーバー・マシンの両方で、WebSEAL をインストールして構成します。
2. **pdadmin** コマンドを使用して、両方の WebSEAL サーバー用の許可スペースのルートとなる新規オブジェクトを作成します。以下に例を示します。

```
pdadmin> object create /WebSEAL/newroot "Description" 5 ispolicyattachable yes
```
3. WS1 の WebSEAL を停止します。
4. WS1 で、WebSEAL 構成ファイル内の **server-name** スタンザ・エントリーの値を、「WS1」から「newroot」に変更します。

```
[server]
server-name = newroot
```
5. WS1 で WebSEAL を再始動します。
6. WS2 について、ステップ 3 から 5 を繰り返します。

これで、WS1 および WS2 サーバーは、許可を評価するためのベースとしてオブジェクト /WebSEAL/newroot を使用するようになります。WS1 および WS2 のどちらのサーバーも、/WebSEAL/newroot の下位にあるオブジェクトに対する **object list** および **object show** コマンドに応答することができます。

WS1 または WS2 を構成解除するときは、以下の手順を使用します。

手順

1. WebSEAL サーバーを停止します。
2. **server-name** スタンザ・エントリーの値を元の値に変更します。例えば、WS1 の場合は次のように入力します。

```
[server]
server-name = WS1
```

3. 通常の構成解除手順を進めます。

タスクの結果

条件:

- 統一化されたオブジェクト・スペース管理: アドミニストレーターに見えるのは単一のオブジェクト階層であるが、そのオブジェクト階層に適用される管理コマンドは複製されたすべての WebSEAL サーバーに影響を与え、すべてのサーバーがこれらのコマンドに応答することができる。
- 統一化された許可評価: WS1 および WS2 は、許可評価のベースとして /WebSEAL/newroot を使用する。
- 統一化された構成: フロントエンド WebSEAL の複製が正しく機能するために、各サーバーで、Web スペース、junction データベース、dynurl データベース構成が同じでなければならない。

login_success 応答のコントロール

このタスクについて

ロード・バランシング・システムによって制御される複数の WebSEAL インスタンスを含むネットワーク・トポロジーでは、要求 URL が、認証プロセスで起きる通信交換中に「失われる」ことがあります。例えば、1 つの WebSEAL インスタンスが、(保護リソースに対する) 要求 URL を受け取り、ユーザーにログイン・フォームを提示することが可能です。ユーザーが完了したログイン・フォームをサブミットするとき、非スティッキー・ロード・バランサーが、POST データを 2 番目の WebSEAL インスタンスに送信する場合があります。入力したログイン・フォームをユーザーがサブミットしたとき、非スティッキー・ロード・バランサーが POST データを 2 番目の WebSEAL インスタンスに送信する場合があります。(スティッキー・ロード・バランシングは、一連のサーバー間でユーザー要求を分散させ、特定のユーザーからの要求が常に同じサーバーに送信されるようにします。)

この 2 番目の WebSEAL インスタンスは、ログイン POST データを正常に処理できますが、元の URL 要求にリダイレクトすることはできません。この場合、2 番目の WebSEAL インスタンスは、「ログインが成功しました。」のメッセージを報告する login_success.html ページを送信します。

注: 初期の非証明書要求が POST であると、元の URI へのリダイレクトが、POST を受け取るリソースへの GET となるため、WebSeal は、Referer ヘッダーを利用しません。

考えられる解決策としては、以下があります。

- スティック・ロード・バランシング・システムを使用して、十分なスティッキー時間（例えば、20 秒から 30 秒）を構成し、1 つの WebSEAL インスタンスだけが全体のログイン交換を処理できるようにします。
- WebSEAL 構成ファイルを変更して、認証後の自動リダイレクトを使用可能にします。WebSEAL は、ログイン後のプロセスをより良く処理できるような指定応答ページにユーザーをリダイレクトします。自動リダイレクトによって、WebSEAL は、login_success.html 応答ページを使用しなくなります。自動リダイレクトには、以下のスタンザおよびスタンザ・エントリーが含まれます。

```
[enable-redirects]
redirect =

[acct-mgt]
login-redirect-page =
```

詳しくは、229 ページの『認証後の自動リダイレクト』を参照してください。

- login_success.html 応答ページを変更し、要求履歴にユーザーのブラウザーをリダイレクトするようにします。この技法により、2 番目の WebSEAL インスタンスが元の要求 URL を受け取り処理できるようにします。例:

```
<HTML>
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<TITLE>Success</TITLE>
</HEAD>
<BODY onLoad="history.back()">
...
</BODY>
</HTML>
```

第 37 章 アプリケーションの統合

この章には、WebSEAL 環境へのサード・パーティーのアプリケーション・サーバーの統合に関するさまざまな情報が収録されています。

この章で扱うトピックは以下のとおりです。

- 『バックエンド・サーバー・サイド・アプリケーションのサポート』
- 672 ページの『標準 junction の推奨構成例』
- 673 ページの『カスタム・パーソナライゼーション・サービス』
- 675 ページの『バックエンド・サーバーのユーザー・セッション管理』

バックエンド・サーバー・サイド・アプリケーションのサポート

WebSEAL では、junction バックエンド・サーバーで実行され、入力としてクライアント識別情報を必要とするサーバー・サイド・アプリケーションをサポートしています。サーバー・サイド・アプリケーションの例としては、次のものがあります。

- Java サープレット
- Oracle Web Listener 用カートリッジ
- サーバー・サイド・プラグイン

-c オプションを使用してバックエンド・サーバーへの junction を作成すると、WebSEAL は、そのサーバー宛ての要求の HTTP ヘッダーに、Security Access Manager 固有のクライアント識別情報を挿入します。Security Access Manager 固有のヘッダーには、次のものがあります。

- **iv-user**
- **iv-user-l**
- **iv-groups**
- **iv-creds**
- **iv-remote-address**
- **iv-remote-address-ipv6**

Junction 先サード・パーティー・サーバー上のアプリケーションは、この Security Access Manager 固有の HTTP ヘッダー情報を使用して、クライアントの Security Access Manager 識別に基づいてユーザー固有のアクションを実行できます。

このトピックについて詳しくは、以下を参照してください。

- 596 ページの『HTTP ヘッダー内のクライアント識別 (-c)』。
- 599 ページの『HTTP ヘッダーのクライアント IP アドレス (-r)』。

標準 junction の推奨構成例

このセクションでは、アプリケーション統合で標準 WebSEAL junction を使用するときの推奨構成例を紹介します。

- 『-v を使用した完全な Host ヘッダー情報』
- 『絶対 URL の標準フィルター操作』

-v を使用した完全な Host ヘッダー情報

仮想ホスト構成およびポータル・アプリケーションでは、適正なソケット接続を達成するための正しい IP アドレス、および正確なルーティングを行うための完全なサーバー名情報が必要です。

これらの特殊バックエンド・アプリケーション・サービスでは、ブラウザからのすべての要求に、完全なサーバー名およびポート指定情報が含まれていることが必要です。これらの情報は要求の **Host** ヘッダーに含まれており、アプリケーションはこのヘッダーを使用することができます。WebSEAL junction を使用しているときは、**-v junction** オプションを使用して、**Host** ヘッダーにこれらの情報が与えられます。

サーバー名およびポート情報が不完全であったり欠落していたりすると、仮想ホスティングおよびポータル・アプリケーションのパフォーマンスが低下します。さらに、これらのアプリケーションが設定するドメイン Cookie に十分な情報が組み込まれないこともあります。

Host ヘッダーに最も完全な情報を与えるための「推奨構成例」として、junction を作成または追加するときに、**-v** オプションの中で、必ず、junction 先サーバーの完全修飾ドメイン名と、接続ポート番号を使用することをお勧めします。

-v オプションの構文は次のとおりです。

-v *fully-qualified-host-name[:port]*

例:

-v xyz.ibm.com:7001

注: ポート指定が必要なのは、標準外ポート番号を使用する場合のみです。

絶対 URL の標準フィルター操作

フロントエンド逆プロキシとしての WebSEAL は、junction 先バックエンド・アプリケーション・サーバーにセキュリティ・サービスを提供します。ほとんどの場合、バックエンド・アプリケーションからクライアントに戻されるページには、その junction 先バックエンド・サーバーが持っているリソースへの URL リンクが含まれています。

このようなリンクには、要求をリソースの正しい場所に導くために junction 名を含めておくことが重要です。WebSEAL は、一組の標準ルールに従って静的 URL をフィルターに掛け、この junction 情報を提供します。スクリプト内の URL および動的に生成された URL をフィルターに掛けるためには、そのための追加構成が必

要です。URL のフィルター操作についての詳細は、475 ページの『第 25 章 Junction リソースへの URL の変更』を参照してください。

WebSEAL が静的 HTML ページにある絶対 URL を正しくフィルターに掛けることができるようにするには、**-h junction** オプションでサーバー名に関する情報を与える必要があります。このオプションは、junction 先バックエンド・サーバーの名前を WebSEAL に提供します。このオプションの引数には以下のものがあります。

- サーバーの完全修飾ドメイン名
- サーバーのショート・ネーム
- サーバーの IP アドレス

WebSEAL が、フィルターに掛ける絶対 URL を識別するためには、junction 先バックエンド・サーバー名を知っていることが必要です。ネットワーク環境によっては、**-h short-name** 構成では WebSEAL に十分な情報が与えられない場合があります。

以下の例では、**ibm.com**[®] ネットワーク上にあって「xyz」というショート・ネームを持つバックエンド・サーバー用として、次のオプションと引数を使用して junction が作成されます。

```
-h xyz
```

このサーバーから HTML ページへのリンクは、次のようになります。

```
http://xyz.ibm.com/doc/release-notes.html
```

要求の実行中にこのページがクライアントに渡されるときに、WebSEAL は、**-h** から得られる情報に基づいて「xyz.ibm.com」をサーバー名として認識できない場合があります。そのため、この URL のフィルター操作に失敗することがあります。つまり、パス内に junction 名がないと、release-notes 文書に対する要求は失敗します。

静的絶対 URL の適切なフィルター操作ができるようにするための「推奨構成例」としては、junction を作成または追加するときに、**-h** オプションに必ず junction 先サーバーの完全修飾ドメイン名を含めておくことをお勧めします。

カスタム・パーソナライゼーション・サービス

このセクションは、以下のトピックで構成されています。

- 『パーソナライゼーション・サービスの概念』
- 674 ページの『パーソナライゼーション・サービスのための WebSEAL の構成』
- 674 ページの『パーソナライゼーション・サービスの例』

パーソナライゼーション・サービスの概念

Web ポータル (起動) ページは、特定ユーザーが利用できる Web リソースのカスタマイズ済みリストを動的に作成する、統合 Web サイト・サービスです。この種のリソースには、企業コンテンツ、サポート・サービス、および学習ツールなどが含まれます。ポータル出力は、特定ユーザーのアクセス許可に基づいて個人別に設定されたリソースのリストを表示します。起動ページには、そのユーザーが正当なアクセス許可を持っているリソースのみが表示されます。

アドミニストレータは、WebSEAL 構成オプションと許可 API 資格サービスを使用して、Security Access Manager 環境内でのカスタム・ポータル・ソリューションを作成します。

WebSEAL カスタム・ポータル・サービスを作成するためのプロセス・フローでは、以下のことが行われます。

1. セキュア・ポリシーが作成され、保護オブジェクト・リソースの該当ポイントに付加されます。
2. これらのリソース・オブジェクトのそれぞれに、該当する明示 ACL が付加されます。
3. WebSEAL 構成ファイルが編集されて、ポータル・サービスへの URL、ポータル・リソースが入っているオブジェクト・スペースのパス、およびユーザーがこれらのリソースにアクセスするために必要な許可ビットが組み込まれます。
4. ポータル URL に対する個々のユーザー要求について、WebSEAL は、許可資格サービスを使用して該当オブジェクト・スペースを検索し、ユーザーの許可条件を満たすリソースのリストを作成します。
5. WebSEAL は、バックエンドの (junction 先) ポータル・サーバーに送る PD_PORTAL HTTP ヘッダーにこの情報を組み込みます。
6. バックエンド・サーバーに配置されているカスタム・ポータル・サービス (CGI やサブリットなど) は、PD_PORTAL ヘッダーの内容を読み、そして、例えばその内容を、Web ページとしてユーザーに表示される記述および URL にマップします。この情報は、アクセス・コントロール許可に基づいてパーソナライズされた、そのユーザーが利用できるリソースのリストを表します。

パーソナライゼーション・サービスのための WebSEAL の構成

手順

1. パーソナライゼーション・サービスに対する新規 junction を作成します。例えば、以下のようにします (1 行で入力します)。

```
pdadmin> server task server_name create -t tcp -h portalhost.abc.com /portal-jct
```
2. WebSEAL 構成ファイルを編集して、新しい **[portal-map]** スタンザを追加します。

```
[portal-map]
```
3. このスタンザ内のエントリは、ポータル・サービス・プログラムのサーバー相対 URL と、使用可能な保護ポータル・リソースが検索されるオブジェクト・スペース領域、そして、アクセスに必要な許可を指定します。このリストは PD_PORTAL ヘッダーに組み込まれます。

```
[portal-map]  
URL = object_space_region:permission
```
4. スタンザおよび適切なマッピング・エントリを追加した後は、WebSEAL (**webseald**) を再始動する必要があります。

パーソナライゼーション・サービスの例

- ポータル・サーバーへの junction を作成します。

```
pdadmin> server task web1-webseald-cruz -t ssl -h PORTAL1 /portal
```

- ・ パーソナライゼーション・サービスにより利用できるリソースが入る、WebSEAL 保護オブジェクト・スペースの領域を定義します。

```
pdadmin> objectspace create /Resources "Portal Object Hierarchy" 10
pdadmin> object create /Resources/Content "" 10 ispolicyattachable yes
pdadmin> object create /Resources/Support "" 10 ispolicyattachable yes
pdadmin> object create /Resources/Content/CGI "" 11 ispolicyattachable yes
pdadmin> object create /Resources/Support/Servlet "" 11 ispolicyattachable yes
```

注: 「ispolicyattachable」引数は、どのリソースについても「yes」に設定する必要があります。検索メカニズムは、明示的に設定された ACL を持つ、修飾されたリソース・オブジェクトのみを選択します。

- ・ WebSEAL 構成ファイル:

```
[portal-map]
/portal/servlet/PortalServlet = /Resources:r
```

- ・ ユーザーが使用するポータル URL:

```
https://WS1/portal/servlet/PortalServlet
```

バックエンド・サーバーのユーザー・セッション管理

WebSEAL は、クライアントとのセッション状態を維持することができます。

junction 先バックエンド・アプリケーション・サーバーにこのセッション情報を拡張するように、WebSEAL を構成することもできます。バックエンド・アプリケーションでは、セッション情報を使用してクライアントとのセッション状態を維持することも可能であり、またセッションを終了することも可能です。

このセクションは、以下のトピックで構成されています。

- ・ 『ユーザー・セッション管理の概念』
- ・ 676 ページの『ユーザー・セッション ID 管理の使用可能化』
- ・ 677 ページの『HTTP ヘッダーへのユーザー・セッション・データの挿入』
- ・ 679 ページの『ユーザー・セッションの終了』
- ・ 682 ページの『バックエンド・サーバー用のユーザー・イベント関連』

ユーザー・セッション管理の概念

クライアント/サーバー・セッションは、1 つのクライアントとサーバーの間で一定期間にわたり発生する一連の関連通信です。確立されたセッションで、サーバーは各要求に関連するクライアントを識別し、多数の要求にわたって特定のクライアントを認識することができます。

確立済みのセッションがない場合は、後続の要求ごとにクライアントとサーバーの間の通信が再折衝されます。セッション状態情報を利用すると、クライアント/サーバー・セッションのクローズと再オープンの繰り返しを避けることができるので、パフォーマンスが向上します。クライアントは、各要求ごとに新たなログインを行うことなく、1 回ログインするだけで多数の要求を出すことができます。

WebSEAL サーバーには、クライアントとのセッションの状態を維持し、このセッション情報を junction バックエンド・アプリケーション・サーバーに拡張することができます。

WebSEAL では、WebSEAL セッション ID と呼ばれるセッション識別キーを使用して、クライアントと WebSEAL の間のセッションの状態を維持します。WebSEAL セッション ID は、WebSEAL セッション・キャッシュに格納されているクライアントのセッション・データの索引として機能します。287 ページの『WebSEAL セッション・キャッシュ構造』および 293 ページの『セッション・キャッシュ構成の概要』を参照してください。

ユーザー・セッション ID と呼ばれる別々のセッション識別キーを使用して、クライアントと junction バックエンド・アプリケーション・サーバーの間のセッションの状態を維持できます。ユーザー・セッション ID は、認証ユーザーの特定セッションを一意的に識別します。この ID はユーザーの資格情報の一部として格納されます。

バックエンド・アプリケーションは、ユーザー・セッションの追跡とセッションの終了にユーザー・セッション ID を使用できます。『ユーザー・セッション ID 管理の使用可能化』を参照してください。

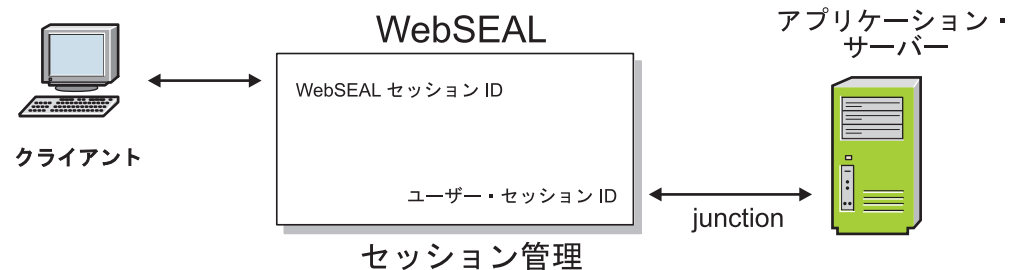


図 54. セッション管理

1 人のユーザーが異なるマシンなどから複数回ログインすると、そのユーザーに対し複数の WebSEAL セッション ID と、セッションごとに 1 つ資格情報が作成されます。ユーザー・セッション ID は WebSEAL セッション ID に基づいています（この 2 つのキーは 1 対 1 でマッピングされます）。したがって、WebSEAL セッション ID ごとにユーザー・セッション ID が存在します。

ユーザー・セッション ID を使用したセッション管理を使用可能にするには、以下の 2 つの構成ステップを実行する必要があります。

- 個々の認証クライアントの固有ユーザー・セッション ID を、各クライアントの資格情報に拡張属性として保管するように WebSEAL を構成します。
- この資格情報の拡張属性の値（ユーザー・セッション ID）を HTTP ヘッダーに挿入してバックエンド・アプリケーション・サーバーに提供できる junction で、拡張属性を構成します。

ユーザー・セッション ID 管理の使用可能化

このタスクについて

WebSEAL 構成ファイルの [session] スタンザ内の **user-session-ids** スタンザ・エントリを使用すると、要求を出す各クライアントの資格情報に拡張属性として固有のユーザー・セッション ID を作成する機能を使用可能または使用不可にできます。デフォルト値は、「no」（使用不可）です。

```
[session]
user-session-ids = no
```

手順

固有のユーザー・セッション ID を作成できるようにするには、以下のように設定します。

```
[session]
user-session-ids = yes
```

固有のユーザー・セッション ID は、名前と値を持つ拡張属性としてユーザーの資格情報に保管されます。

```
tagvalue_user_session_id = user-session-id-string
```

資格情報内の他の既存情報との競合を回避するために、この拡張属性名には常に「tagvalue_」接頭部が付加されます。

HTTP ヘッダーへのユーザー・セッション・データの挿入

クライアントのユーザー・セッション ID をバックエンド・アプリケーション・サーバーに提供するには、junction で **HTTP-Tag-Value** 拡張属性を構成します。

junction での拡張属性の設定

このタスクについて

一般に、属性により junction が特定のタイプの操作を実行できるようになります。WebSEAL 保護オブジェクト・スペース内の junction オブジェクトについて属性を設定するには、**pdadmin object modify set attribute** コマンドを使用します。

```
pdadmin> object modify object_name set attribute attr_name attr_value
```

junction の HTTP-Tag-Value 拡張属性

HTTP-Tag-Value 拡張属性は、WebSEAL に対し、ユーザー資格情報の拡張属性から値を抽出し、各要求を受けてこの値を HTTP ヘッダーに挿入し、junction を介してバックエンド・アプリケーション・サーバーに送信するように明確に指示します。

HTTP-Tag-Value 属性の内容を以下に示します。

- 資格情報の特定の拡張属性を識別します。
- この属性の値を挿入する HTTP ヘッダーのカスタム名を指定します。

HTTP-Tag-Value 属性で使用する形式は以下のとおりです。

```
credential_extended_attribute_name=http_header_name
```

属性の構成要素を以下に示します。

- *credential_extended_attribute_name*

ユーザー・セッション ID が格納されている資格情報拡張属性の名前は **user_session_id** です。(使用される形式は、拡張属性名から「tagvalue_」接頭部を除いたものです。)

この資格情報拡張属性名では大文字小文字が区別されません。

- *http_header_name*

http_header_name の値は、junction を介してユーザー・セッション ID を配信するために使用する HTTP ヘッダーのカスタム名を指定します。

このセクションの例では、**TAM-USER-SESSION-ID** というヘッダーが使用されています。

HTTP-Tag-Value junction 属性の設定

このタスクについて

HTTP-Tag-Value junction 属性を設定する **pdadmin** コマンドの例を以下に示します (1 行で入力します)。

```
pdadmin> object modify /WebSEAL/junctionA set attribute  
HTTP-Tag-Value user_session_id=TAM-USER-SESSION-ID
```

HTTP-Tag-Value junction 属性の処理

このタスクについて

WebSEAL は、バックエンド・アプリケーション・サーバーへのクライアント要求を処理するときに、junction オブジェクトに属性が構成されていないかどうかを確認します。

この例では、WebSEAL により以下の処理が行われます。

手順

1. **HTTP-Tag-Value** 属性を検出する。
2. 要求を出したユーザーの資格情報を読み取る。
3. 資格情報の **tagvalue_user_session_id** 拡張属性からユーザー・セッション ID 値を抽出する。
4. この値を **TAM-USER-SESSION-ID** HTTP ヘッダーに以下のように挿入する。

TAM-USER-SESSION-ID:user-session-id-string

タスクの結果

上記を要約すると以下ようになります。

- ユーザー資格情報でのユーザー・セッション ID の名前と値は以下のようになります。

tagvalue_user_session_id:user-session-id-string

- junction オブジェクトに設定される **HTTP-Tag-Value** 属性の値の例を以下に示します。

user_session_id=TAM-USER-SESSION-ID

- 作成される HTTP ヘッダーの名前と値の例を以下に示します。

TAM-USER-SESSION-ID:user-session-id-string

ユーザー・セッションの終了

ユーザーは、**pkmslogout** コマンドを使用して現行セッションを終了できます。アドミニストレーターまたはバックエンド・アプリケーションは、ユーザー・セッション ID スtringの情報をを使用してユーザー・セッションを終了できます。このセクションは、以下のトピックで構成されています。

- 『ユーザー・セッション ID スtringのフォーマット』
- 『古いユーザー・セッション ID フォーマットとの互換性』
- 680 ページの『単一のユーザー・セッションの終了』
- 681 ページの『すべてのユーザー・セッションの終了』

ユーザー・セッション ID スtringのフォーマット

ユーザー・セッション ID スtringのフォーマットは、MIME64 エンコードの WebSEAL サーバー名とクライアントのユーザー・セッション ID からなります。

encoded-webseal-server-name_user-session-id

WebSEAL サーバー名は、WebSEAL 構成ファイルの **[server]** スタンザの **server-name** スタンザ・エントリーに指定されている値です。

pdadmin server task terminate session コマンドにユーザー・セッション ID 値を指定するときに、*user-session-id* 値を直接使用できます。680 ページの『単一のユーザー・セッションの終了』を参照してください。

古いユーザー・セッション ID フォーマットとの互換性

WebSEAL バージョン 6.0 以降、ユーザー・セッション ID 値のフォーマットに、レプリカ・セット名を組み込むことができるようになりました。

この操作は、WebSEAL 構成ファイルの **[session]** スタンザの **user-session-ids-include-replica-set** スタンザ・エントリーにより制御されます。デフォルト値は「yes」です。例:

```
[session]
user-session-ids-include-replica-set = yes
```

値「yes」は、WebSEAL に対しユーザー・セッション ID 値にレプリカ・セット名を組み込むように指示します。この設定により、**pdadmin server task terminate session** を使用して仮想ホスト **junction** のユーザー・セッションを終了できます。

バージョン 6.0 より前のバージョンの WebSEAL とのユーザー・セッション ID の互換性が必要な場合は、この機能を使用不可にします。例:

```
[session]
user-session-ids-include-replica-set = no
```

値「no」は、WebSEAL に対しユーザー・セッション ID 値にレプリカ・セット名を組み込まないように指示します。

`user-session-ids-include-replica-set = no` が指定されていると、WebSEAL は [session] の **standard-junction-replica-set** スタンザ・エントリーにより定義されているレプリカ・セットのメンバーに **pdadmin server task terminate session** コマンドが適用されるものと解釈します。

pdadmin server task terminate session コマンドは、仮想ホスト `junction` で開始されたセッションに対しては失敗します。

レプリカ・セットに標準 `junction` を割り当てるには、366 ページの『標準 `junction` のレプリカ・セットへの割り当て』を参照してください。

レプリカ・セットに仮想ホストを割り当てるには、366 ページの『レプリカ・セットに割り当てられる仮想ホスト』を参照してください。

単一のユーザー・セッションの終了

注: SMS 環境では **pdadmin server task terminate session** コマンドがサポートされています。

アドミニストレーターまたはバックエンド・アプリケーションは、Security Access Manager の管理 API を使用して、ユーザー・セッション ID に基づき、特定のユーザー・セッションを終了することができます。679 ページの『ユーザー・セッション ID スtringのフォーマット』を参照して、ユーザー・セッション ID スtringの構造を確認してください。

ユーザー・セッション ID スtringの `user_session_id` 部分を **ivadmin_server_performtask()** 関数に渡すことができます。この関数は標準 **pdadmin server task terminate session** コマンドから入力コマンド・スStringを取り込みます。以下に例を示します。

```
pdadmin> server task instance-webseald-host terminate session user_session_id
```

各要求で渡される HTTP `iv_server_name` ヘッダーから WebSEAL インスタンス名を取得できます。

注: この **pdadmin** 操作を手動で実行できますが、`user_session_id` の長い値が原因でこの作業が煩雑になることがあります。

WebSEAL は、ユーザー・セッションを終了する前に、終了操作を開始しようとしているバックエンド・サーバーがそのために必要な許可を持っているかどうかを確認します。次に WebSEAL は、セッションが終了できるように、対応するセッション・キャッシュ・エントリーを除去します。

このコマンドをどのような条件下で使用するかについて考慮することが重要です。ユーザーを完全にセキュア・ドメインから除去することを目的としている場合は、そのユーザーのアカウントも無効に (除去) しない限り、このコマンドで単一ユーザーを終了するだけでは効果はありません。

ある種の認証方式 (基本認証、クライアント・サイド証明書、LTPA Cookie、およびフェイルオーバー Cookie など) では、ユーザーの介入を必要とせずに、キャッシュに入れられた認証情報が自動的に戻されます **pdadmin server task terminate session**

アクションでは、これらの認証方式のいずれかを使用しているユーザーの戻りログインを防ぐことはできません。したがって、レジストリー内の該当のユーザー・アカウントも無効にする必要があります。

詳細および **ivadmin_server_performtask()** 構文については、「*IBM Security Access Manager for Web: Administration C API Developer Reference*」を参照してください。

セッション終了によりユーザーが予期せずにログアウトした場合、ユーザーのブラウザに残存している元のセッションの Cookie は、古い（「失効した」）Cookie となり、WebSEAL のセッション・キャッシュの既存エントリーにマップしなくなります。ユーザーから保護オブジェクトに対する後続の要求が出されると、WebSEAL は認証を要求し、ログイン・フォームを戻します。新規のログインが必要とされる理由を説明する追加情報が含まれるように、ログイン応答をカスタマイズすることができます。この機能について詳しくは、333 ページの『古いセッション Cookie に対するカスタマイズ応答』を参照してください。

すべてのユーザー・セッションの終了

注: **pdadmin server task terminate all_sessions** コマンドは、SMS 環境ではサポートされません。代わりに **pdadmin server task sms session terminate** コマンドを使用してください。

アドミニストレーターまたはバックエンド・アプリケーションは、Security Access Manager の管理 API を使用して **pdadmin** コマンドを呼び出すことができます。このコマンドは、ユーザーのログイン ID に基づき、特定のユーザーのセッションをすべて終了します。以下に例を示します。

```
pdadmin> server task instance-webseald-host terminate all_sessions login_id
```

ユーザーのログイン ID (*login_id*) を Security Access Manager **iv-user** ヘッダーに挿入して junction バックエンド・サーバーに渡すことができます。このタスクを実行するには、**-c iv_user** オプションおよび引数を使用して junction を最初に作成する必要があります。596 ページの『HTTP ヘッダー内のクライアント識別 (-c)』を参照してください。

WebSEAL のセッション・キャッシュは、ユーザーのログイン ID、WebSEAL セッション ID、およびその他のキャッシュ・エントリー情報を相互参照するように編成されています。各ユーザーは、常に複数のセッションに共通する 1 つのログイン ID を持っています。しかし、WebSEAL セッション ID は各セッションに固有のもので、**pdadmin server task terminate all_sessions** コマンドは、固有のユーザー・ログイン ID に属しているすべてのキャッシュ・エントリーを除去します。

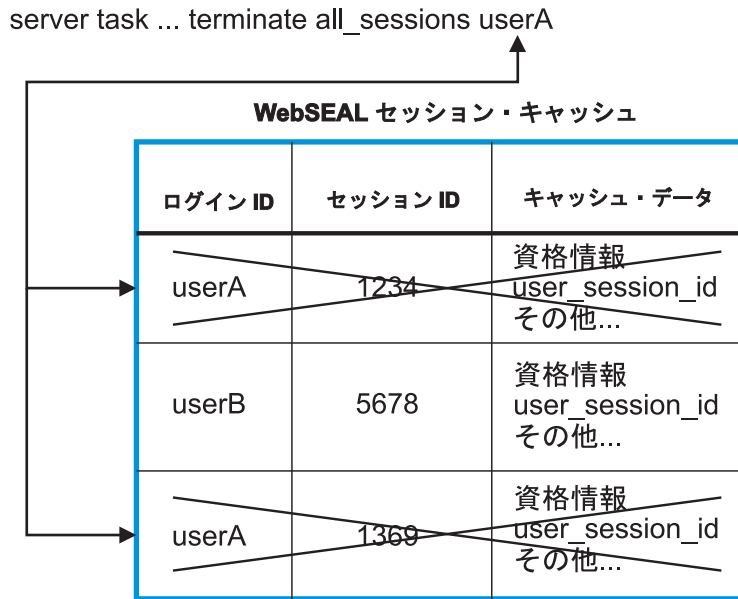


図 55. すべての userA セッションの終了

WebSEAL は、ユーザー・セッションを終了する前に、**pdadmin** コマンド起動側に正しい許可があるかどうかを確認します。

このコマンドをどのような条件下で使用するかにについて考慮することが重要です。特定のユーザー・グループをセキュア・ドメインから完全に除去することを目的としている場合は、グループのユーザーのアカウントも無効に (除去) しない限り、**pdadmin server task terminate all_sessions** コマンドだけを使用しても効果はありません。

ある種の認証方式 (基本認証、クライアント・サイド証明書、LTPA Cookie、およびフェイルオーバー Cookie など) では、ユーザーの介入を必要とせずに、キャッシュに入れられた認証情報が自動的に戻されます。**pdadmin server task terminate all_sessions** コマンドでは、これらの認証方式のどれかを使用しているユーザーの戻りログインを防ぐことはできません。したがって、レジストリー内の該当のユーザー・アカウントも無効にする必要があります。

セッション終了によりユーザーが予期せずにログアウトした場合、ユーザーのブラウザに残存している元のセッションの Cookie は、古い (「失効した」) Cookie となり、WebSEAL のセッション・キャッシュの既存エントリーにマップしなくなります。ユーザーから保護オブジェクトに対する後続の要求が出されると、WebSEAL は認証を要求し、ログイン・フォームを戻します。新規のログインが必要とされる理由を説明する追加情報が含まれるように、ログイン応答をカスタマイズすることができます。この機能について詳しくは、333 ページの『古いセッション Cookie に対するカスタマイズ応答』を参照してください。

バックエンド・サーバー用のユーザー・イベント関連

場合によっては、WebSEAL と junction 先 Web サーバーの間で個々のイベントを相関させることが必要になります。WebSEAL イベントは、セッション ID とトランザクション ID の組み合わせによって一意的に識別できます。

HTTP ヘッダーへのイベント関連データの挿入

このタスクについて

セッション ID を HTTP ストリームに挿入する方法については、675 ページの『バックエンド・サーバーのユーザー・セッション管理』を参照してください。

トランザクション ID を HTTP ストリームに挿入するには、WebSEAL の HTTP-Tag-Value 機能を使用できます。この機能を使用して、ユーザー・セッションからのエレメントを HTTP ストリームに挿入します。WebSEAL 保護オブジェクト・スペース内の junction オブジェクトの拡張属性は、転送される要求に含まれるセッション・エレメントを定義します。以下のように、**pdadmin object modify set attribute** コマンドでオブジェクトに属性を設定します。

```
<pdadmin> object modify <object_name> set attribute <attr-name> <attr-value>
```

<attr-name> の値は常に「HTTP-Tag-Value」に設定する必要があります。

<attr-value> のフォーマットは、以下のようにします。

```
<session_data_name>=<http_header_name>
```

トランザクション ID を HTTP ストリームに含めるには、<session_data_name> の値を session:tid に設定します。<http_header_name> の値は、junction を介してトランザクション ID を送信する HTTP ヘッダーのカスタム名を指定します。

以下の例は、HTTP-Tag-Value junction 属性を設定する pdadmin コマンドを示しています。コマンドは 1 行で入力します。

```
<pdadmin> object modify /WebSEAL/junctionA
set attribute HTTP-Tag-Value session:tid=TAM-TRANSACTION-ID
```

WebSEAL 要求ログへのイベント関連データの挿入

request-log-format 構成エントリーを使用して、イベント関連データを含むように WebSEAL 要求ログの内容をカスタマイズすることができます。

このタスクについて

フォーマット指定子はログ・テキストの内容を定義します。以下のフォーマット指定子によって、イベント関連データが要求ログに追加されます。

データ	フォーマット指定子
トランザクション ID	%d
セッション ID	%{tagvalue_user_session_id}C

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の **request-log-format** 構成エントリーを参照してください。

第 38 章 動的 URL

この章では、動的 URL の保護について説明します。

この章で扱うトピックは以下のとおりです。

- 『動的 URL に対するアクセス・コントロール』
- 692 ページの『動的 URL の例: Travel Kingdom 社の場合』

動的 URL に対するアクセス・コントロール

現行の Web 環境では、ユーザーは急激に変化する情報に即時にアクセスできます。多くの Web アプリケーションは、それぞれのユーザー要求に対する応答として、動的に URL を生成します。通常、このような動的 URL は、短時間しか存在しません。動的 URL は、本質的に一時的なものには違いありませんが、望ましくない使用やアクセスに対して強力な保護が必要であることに変わりはありません。

動的 URL のコンポーネント

一部の Web アプリケーション・ツールは、動的 URL と隠しフォーム・エレメントを使用して、要求されたオペレーション（および値）をアプリケーション・サーバーに伝えます。動的 URL は、特定のオペレーションとその値で標準 URL アドレスを補足します。

Web アプリケーション・インターフェースに提供された動的データ（オペレーション、パラメーター、および値）は、要求 URL の照会ストリング部分に配置されています。

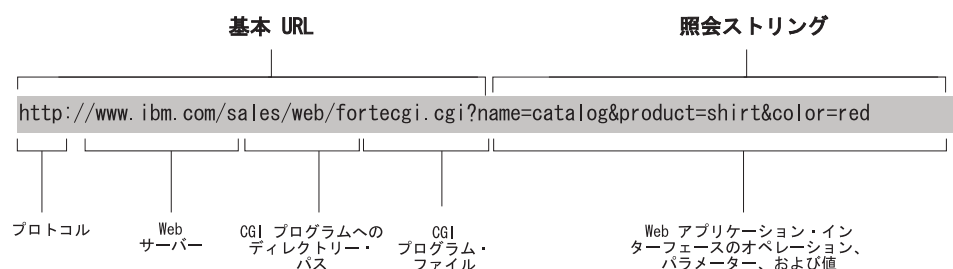


図 56. 要求 URL の照会ストリングでのデータの引き渡し

動的 URL に対するアクセス・コントロール: dynurl.conf

WebSEAL は、保護オブジェクト・スペース・モデル、アクセス・コントロール・リスト (ACL)、および保護オブジェクト・ポリシー (POP) を使用して、データベース要求により生成される URL など、動的に生成された URL を保護します。WebSEAL への各要求は、許可プロセスの最初のステップとして特定のオブジェクト

トに解決されます。オブジェクトに適用された ACL または POP は、そのオブジェクトにマップされる動的 URL に対する必要な保護を指示します。

動的 URL は一時的に存在するだけであるため、事前構成許可ポリシー・データベース・エントリーを設けておくことはできません。Security Access Manager は、多くの動的 URL を 1 つの静的保護オブジェクトにマップするメカニズムを備えることによって、この問題を解決しています。

オブジェクトからパターンへのマッピングは、dynurl.conf と呼ばれるプレーン・テキスト構成ファイルに保持されます。

LMI を使用して動的 URL 構成ファイルを管理できます。「セキュア・リバース・プロキシ設定 (Secure Reverse Proxy Settings)」>「グローバル設定」>「URL マッピング」に移動します。

このファイルの名前は、WebSEAL構成ファイルの **[server]** スタンザ内の **dynurl-map** スタンザ・エントリーで定義します。

```
[server]
dynurl-map = dynurl.conf
```

なお、このファイルは、デフォルトでは存在しないため、ユーザーが作成しなければなりません。

WebSEAL 開始中にこのファイル (エントリー付き) が存在していれば、動的 URL 機能が使用可能になります。

注: Web Portal Manager を使用してファイルを表示しようとしたが、ファイルがない場合は、次のエラー・メッセージが表示されます。

The dynurl configuration file dynurl.conf cannot be opened for reading.

このファイルを作成すると、エラー・メッセージは表示されなくなります。

照会ストリング・フォーマットへの POST 本体動的データの変換

URL 正規表現をオブジェクト・スペース・エントリーにマップするときは、照会ストリング・フォーマットには、要求方式 (POST または GET) に関係なく、GET 要求方式によって作成されるフォーマットを使用する必要があります。

GET 要求では、Web アプリケーション・インターフェースに提供された動的データ (オペレーション、パラメーター、および値) は、要求 URL の照会ストリング部分に配置されています。

POST 要求では、Web アプリケーション・インターフェースに提供される動的データ (オペレーション、パラメーター、および値) は、要求本体にあります。

アクセス・コントロール評価に必要な必須の照会ストリング構造を維持するために、WebSEAL は、POST 要求本体情報を照会ストリング・フォーマットに変換します。例えば、POST 要求は、以下の要求 URL を持っています。

```
http://server/login.form
```


動的データ (ユーザーによって提供されるログイン情報) は、この POST 要求の本体にあります。

```
name=ibm&password=secure
```

動的 URL 評価に必要な照会ストリング構造を得るためには、WebSEAL は、以下のように POST 本体情報を要求 URL に付加します。

```
http://server/login.form?name=ibm&password=secure
```

動的 URL への ACL および POP オブジェクトのマッピング

このタスクについて

動的 URL のアクセス・コントロールを指定するには、dynurl.conf 構成ファイルを作成し、そのファイルを編集してリソース・オブジェクトをパターンにマップします。ファイル内のエントリーの形式は、次のとおりです。

object template

Security Access Manager Web Portal Manager を使用して、このファイルをリモート側で編集できます。Web Portal Manager で、「WebSEAL」メニューから「動的 URL ファイル」リンクを選択します。「動的 URL」ページを使用すると、WebSEAL サーバーを選択し、そのサーバーにある dynurl.conf 構成ファイルを表示、編集、保管することができます。

Security Access Manager は、オブジェクト・スペース内に 1 つのオブジェクトを構成するパラメーターのセットを定義するために、UNIX シェル・パターン・マッチング (ワイルドカードを含む) のサブセットを使用します。このようなパラメーターに一致する動的 URL は、すべてそのオブジェクトにマップされます。サポートされるワイルドカード・パターン・マッチング文字のリストについては、77 ページの『サポートされているワイルドカード・パターン・マッチング文字』を参照してください。

以下の例は、貸方残高検索を実行する動的 URL (GET 要求から) のフォームを示しています。

```
http://server-name/home-bank/owa/acct.bal?acc=account-number
```

この動的 URL を表すオブジェクトは、次のようになります。

```
http://server-name/home-bank/owa/acct.bal?acc=*
```

この例の動的 URL を綿密に検討してみると、特定の口座番号を記述していることが分かります。**home-bank** の勘定残高を表すオブジェクトは、ACL および POP 許可がどの口座にも適用されることを示しています。どの勘定残高にも適用される理由は、エントリー (**acc=***) の最後の部分にアスタリスク・ワイルドカードが使用されており、これはすべての文字に一致するからです。

次の図では、特定の保護オブジェクトにマップされた特定の動的 URL のシナリオをそのまま示しています。

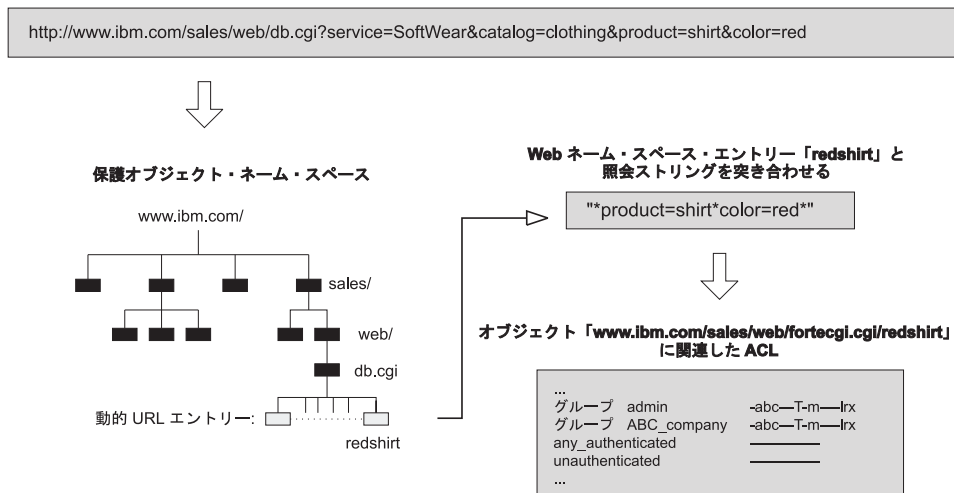


図 57. 動的 URL に対する許可

文字エンコードと照会ストリング検証

動的 URL が使用可能である場合、WebSEAL は、要求の照会ストリングの動的データを、保護 (アクセス・コントロール) を必要とするオブジェクトにマップします。UTF-8 ではなく、WebSEAL が実行されている文字セットのものでもない文字を使用した動的データを (POST 本体または照会ストリングで) WebSEAL が受け取った場合、WebSEAL は要求をリジェクトし、エラーを戻します。

照会ストリングをオブジェクトに安全にマップするには、WebSEAL およびバックエンド・アプリケーション・サーバーの両方に既知である同一の文字セットをストリングで使用する必要があります。同一の文字セットを使用しない場合、バックエンド・アプリケーションに受け入れられるが WebSEAL には受け入れられない文字を使用する要求によって、動的 URL アクセス・コントロールが迂回される可能性があります。

動的 URL 機能は、WebSEAL 構成ファイルの **[server]** スタンザの内の **decode-query** スタンザ・エントリーの値に影響を受けます。WebSEAL が、要求の照会ストリングを検証しないよう構成されている場合 (**decode-query=no**)、許可検査の動的 URL マッピングは、使用可能である場合、使用不可にする必要があります。この条件が満たされないと、WebSEAL は開始しません。

(動的 URL が使用可能であり、かつ **decode-query=yes**) に設定された WebSEAL が非 UTF-8 環境で実行していて、要求 POST 本体 (または照会ストリング) が UTF-8 文字を含んでいる場合、WebSEAL 構成ファイルの **[server]** スタンザの **utf8-form-support-enabled** スタンザ・エントリーを使用して、WebSEAL でこれらの要求の UTF-8 コーディングをデコードできるように設定できます。

動的 URL 用の WebSEAL の更新

手順

WebSEAL 保護オブジェクト・スペースを `dynurl.conf` 構成ファイル内のエントリーで更新するには、**dynurl update** コマンドを使用します。

1. `dynurl.conf` 構成ファイル内の動的 URL エントリーを作成、編集、または削除する。
2. 変更を加え終わったら、**dynurl update** ユーティリティーを使用してサーバーを更新する。

```
pdadmin> server task instance_name-webseald-host_name dynurl update
```

`server-name` 引数は、WebSEAL マシンの非修飾ホスト名を表します。

オブジェクト・スペース内の動的 URL の解決

オブジェクトへの動的 URL の解決は、`dynurl.conf` 構成ファイル内のエントリーの配列によって異なります。

オブジェクト・エントリーへの動的 URL のマップを試みると、`dynurl.conf` ファイル内のマッピングのリストがスキャンされます。ファイルのスキャンは、最初の一一致パターンが見つかるまで、上から下への順に行われます。最初の一一致が見つかったら、対応するオブジェクト・エントリーを使用した後続の許可検査が行われません。

一致が見つからない場合は、WebSEAL は、URL 自体からパスの **http://server** 部分を除いたものを使用します。

限定度が高い ACL に対応するマッピングほどリストの上位に保持します。例えば、受注処理アプリケーションの **book.sales** プロシージャが、ブック・クラブ・グループだけに制限されているが、残りの受注処理アプリケーションはすべてのユーザーによってアクセス可能であるという場合は、マッピングは次の表に示す順序で行わなければなりません。

オブジェクト・スペース・エントリー	URL テンプレート
/ows/sales/bksale	/ows/db-apps/owa/book.sales*
/ows/sales/general	/ows/db-apps/owa/*

マッピング・エントリーが逆の順序であったとすると、`/ows/db-apps/owa` ディレクトリー内のすべてのストアード・プロシージャが、`/ows/sales/general` オブジェクトにマップすることになります。この場合は、このオブジェクト・スペース解決の誤りが原因でセキュリティの侵害を招く可能性があります。

ACL および POP の評価

動的 URL がオブジェクト・スペース・エントリーに解決されると、標準 ACL または POP 継承モデルを使用して、要求を処理するか、それとも禁止するか（特権が不十分の場合）が決定されます。

POST 要求に対する制限の構成

POST 要求のコンテンツは、要求の本体に含まれています。さらに、POST 要求には、ブラウザー次第で決まるこのコンテンツの長さが含まれ、その値がバイト数で示されます。

request-body-max-read

大きい POST 要求が WebSEAL に与える影響を制限するには、WebSEAL 構成ファイルの **[server]** スタンザ内の **request-body-max-read** スタンザ・エントリーを使用して、POST 要求の本体からコンテンツとして読み込む最大バイト数を指定します。このセクションで既に述べたように、WebSEAL が読み込むコンテンツは許可検査による制約を受けます。

request-body-max-read スタンザ・エントリーの値が考慮されるのは、POST 要求が動的 URL 処理またはフォーム認証に使用されるときです。デフォルト値は 4096 バイトです。

```
[server]
request-body-max-read = 4096
```

このスタンザ・エントリーは、POST コンテンツの最大サイズを制限するものではないという点に注意してください (このサイズは無制限です)。このスタンザ・エントリーは、WebSEAL が非現実的なサイズの POST 要求を処理するのを防止するためのものです。**request-body-max-read** の変更について詳しくは、236 ページの『request-body-max-read の変更』を参照してください。

dynurl-allow-large-posts

request-body-max-read スタンザ・エントリーは、WebSEAL が読み取って処理する POST コンテンツの量を制限しますが、要求がアプリケーション・サーバーに渡されるのを完全に阻止するわけではありません。その場合、検証されていないコンテンツがアプリケーション・サーバーに渡されることになります。その結果、アプリケーション・サーバーが独自の許可機能を備えていない場合は、セキュリティ上のリスクが生じるおそれがあります。

dynurl-allow-large-posts スタンザ・エントリーを使用すると、**request-body-max-read** で指定したサイズより大きいコンテンツを含む POST 要求を、WebSEAL がどのように扱うかをコントロールすることができます。このスタンザ・エントリーの値を「no」(デフォルト) に設定すると、WebSEAL は、**request-body-max-read** に指定したサイズより大きいコンテンツを含むすべての POST 要求を完全に拒否します。

```
[server]
dynurl-allow-large-posts = no
```

このスタンザ・エントリーの値を「yes」に設定すると、WebSEAL は、POST 要求全体を受け入れますが、**request-body-max-read** の値に相当する量のコンテンツのみを検証します。

```
[server]
dynurl-allow-large-posts = yes
```

例 1:

- 大きい POST 要求 (**request-body-max-read** の値を超えるもの) が受信された。
- **dynurl-allow-large-posts = no**
- 動的 URL が使用可能にされている。
- 結果: 500 「サーバー・エラー」

例 2:

- 大きい POST 要求 (**request-body-max-read** をを超えるもの) が受信された。
- **dynurl-allow-large-posts = yes**
- 動的 URL が使用可能にされている。
- 結果: WebSEAL は、**request-body-max-read** に達するまでのコンテンツを、`dynurl.conf` 構成ファイル内の各正規表現と比較し、一致が見つかり、対応するオブジェクトに対する許可検査を行います。一致が見つからない場合は、通常どおりに、URL に対応するオブジェクトに対する許可検査が行われます。要求本体の中の **request-body-max-read** を超える部分は検証されません。
- 次のテンプレートに含まれているパターン・マッチングの文字配置は、大きい POST 要求があった場合に誤用を招くことになる例です。

`/rtpi153/webapp/examples/HitCount%?*action=reset*`

動的 URL の要約および技術上の注意点

要約

- 動的 URL を取り扱うときのセキュリティーを確保できるように WebSEAL を構成するには、動的 URL 構成ファイル (`dynurl.conf` など) を作成します。このファイルを作成するには、LMI で、「セキュア・リバース・プロキシー設定 (Secure Reverse Proxy Settings)」>「グローバル設定」>「URL マッピング」の順に移動します。

- このファイルには、次の形式の行が 1 つ以上含まれている必要があります。

object template

- WebSEAL 開始時にこのファイルが存在しないか、または空である場合は、動的 URL 機能は使用可能にされません。
- このファイルが処理された後は、オブジェクト名は WebSEAL オブジェクト・スペース内に予リソースとして表示されます。
- テンプレートには、標準パターン・マッチング文字のサブセットを含めることができます。また、テンプレートは、パターン・マッチング文字なしの、実際のストリングそのものとすることもできます。

次に示すサンプルの `dynurl.conf` ファイルは、IBM WebSphere 製品に含まれているサンプル Web アプリケーションの一部を表す 4 つのオブジェクトを定義しています。

オブジェクト・エントリ	URL テンプレート
<code>/app_showconfig</code>	<code>/rtpi153/webapp/examples/ShowConfig*</code>
<code>/app_snoop</code>	<code>/rtpi153/servlet/snoop</code>

オブジェクト・エントリー	URL テンプレート
/app_snoop	/rtpi025/servlet/snoop
/app_hitcount/ejb	/rtpi153/webapp/examples/HitCount¥?source=EJB
/app_hitcount	/rtpi153/webapp/examples/HitCount*

技術上の注意点

- 1 つのオブジェクトに複数の URL テンプレートをマップできます (例えば、app_snoop を 2 つの異なるサーバー上の URL にマップ)。
- オブジェクトはネストできます (例えば、app_hitcount および app_hitcount/ejb)。
- 着信 URL 要求は、上から下への順にテンプレートと比較されます。一致が見つかった時点で、処理は終了します。したがって、最も限定度の高いテンプレートがファイルの先頭にくるように配置してください。
- dynurl.conf ファイルの中の定義をアクティブにするには、**dynurl update** コマンドを発行します (**pdadmin server task** を使用)。

更新はただちに行われ、保護オブジェクト・スペース・ビューをリフレッシュすると、Web Portal Manager にオブジェクトが表示されます。

- オブジェクト名には大文字は使用しないようにしてください。小文字のみを使用してください。
- 保護オブジェクト・スペース内に既に存在するオブジェクト名は使用しないでください。
- dynurl.conf ファイル内のオブジェクトをどれか削除するときは、その前にそのオブジェクトに付随している ACL をすべて削除しておいてください。
- WebSEAL が、UTF-8 以外の文字、または WebSEAL によって使用される文字セット (コード・ページ) 以外の文字を含む POST 本体を受け取る場合、WebSEAL はその要求を拒否します。
- 動的 URL 評価は、すべての要求に対して作動し、文字パターンの突き合わせを使用するので、照会ストリング (GET) または要求本体 (POST) にバイナリー・データを含む要求は、500 サーバー・エラーを発して拒否されます。

動的 URL の例: Travel Kingdom 社の場合

次の架空の例には、Oracle Web Listener によって生成された URL を、どうすれば企業イントラネットで保護できるかが示されています。

この例で使用されている動的 URL Web サーバーは、Oracle Web Listener です。このテクノロジーは、他の動的 URL Web にも適用できます。

アプリケーション

Travel Kingdom は、インターネットを通して顧客に旅行予約サービスを提供する組織です。同社は自社の Web サーバー上で 2 つの Oracle データベース・アプリケーションを運用し、インターネット経由および自社ファイアウォール内からアクセスできるようにする予定です。

1. 旅行予約システム

許可された顧客は、リモートで予約し、自分の予約の現況について照会できます。Travel Kingdom の従業員も、電話による顧客に対する予約を行い、変更を処理し、その他にも多くのトランザクションを実行できます。外部の顧客は、サービスに対してクレジット・カードで決済するため、そうした情報の伝送をしっかり保護する必要があります。

2. 管理マネージャー

ほとんどの企業がそうであるように、Travel Kingdom でも、給与、地位、経験に関する情報が入っている管理データベースを保守しています。このデータには、各従業員の写真も付いています。

インターフェース

データベースに入っている次のようなストアード・プロシージャへのアクセスを提供できるよう、Oracle Web Server が構成されています。

/db-apps/owa/tr.browse	すべてのユーザーが旅行先、旅行代金などについて照会できるようにします。
/db-apps/owa/tr.book	予約を行う場合に使用します (旅行代理業部門従業員と認証顧客)。
/db-apps/owa/tr.change	現在の予約を検討し変更する場合に使用します。
/db-apps/owa/admin.browse	従業員が、内線番号、E メール・アドレス、写真などのような、制限が設けられていない従業員情報を見る場合に使用します。
/db-apps/owa/admin.resume	従業員が管理データベースに入っている自分の履歴書情報を表示したり変更したりできるようにします。
/db-apps/owa/admin.update	管理部門従業員が従業員に関する情報を更新する場合に使用します。

Web スペースの構造

WebSEAL サーバーを使用して、Travel Kingdom の統一 Web スペースへのセキュア・インターフェースを提供します。

- 旅行予約アプリケーションと管理アプリケーションの両方を実行する Oracle Web Server への junction (/ows) が作成されます。

セキュリティ・ポリシー

使いやすいシステムを保持しながらも、Web リソースに適切なセキュリティを実施するために、会社は以下のようなセキュリティ目標を設けました。

- 旅行代理業部門従業員は、すべての予約を完全にコントロールできる。
- 認証済み顧客は、その人自身の予約を行うことも変更することもできるが、その人以外の認証済み顧客の旅行データに干渉することはできない。
- 管理部門従業員は、管理情報のすべてに対して完全なアクセス権を持つ。
- Travel Kingdom の管理部門以外の従業員は、自分自身の履歴書情報を変更することができ、自分以外の従業員の部分的な情報を見ることができる。

動的 URL からオブジェクト・スペースへのマッピング

上記のセキュリティ・ポリシー目標を達成するためには、次の表に示すように、動的 URL から ACL オブジェクト・エンタリーへのマッピングを構成する必要があります。

このようなマッピングの配列 (順序付け) が、セキュリティ目標の達成に重要な役割を果たしていることに留意してください。

オブジェクト・スペース・エンタリー	URL パターン
/ows/tr/browse	/ows/db-apps/owa/tr.browse¥?dest=*&date=??/??/????
/ows/tr/auth	/ows/db-apps/owa/tr.book¥?dest=*&depart=??/??/????& return=??/??/????
/ows/tr/auth	/ows/db-apps/owa/tr.change
/ows/admin/forall	/ows/db-apps/owa/admin.resume
/ows/admin/forall	/ows/db-apps/owa/admin.browse¥?empid=[th]???
/ows/admin/auth	/ows/db-apps/owa/admin.update¥?empid=????

セキュア・クライアント

クライアントは、安全な暗号化されたチャネルを通して WebSEAL に認証されます。

Web インターフェースを使用する顧客の場合は、さらに Travel Kingdom Webmaster に登録して、アカウントを受け取る必要があります。

アカウントとグループの構造

システム上に以下の 4 つのグループが作成されます。

Staff Travel Kingdom の組織に属する従業員。

TKStaff

Travel Kingdom の旅行代理店。

AdminStaff

Travel Kingdom の管理部門従業員。なお、管理部門従業員は、Staff グループにも入っています。

Customer

インターネットによる旅行の予約を希望する Travel Kingdom の顧客。

各ユーザーには、WebSEAL サーバーがユーザーを個別に識別できるように、セキュア・ドメイン内にそれぞれアカウントが与えられます。ユーザーの識別は Oracle Web Server に渡され、Web リソースのすべてにシングル・サインオン・ソリューションが適用されます。

アクセス制御

次の表には、前記の説明を適用した結果であるアクセス・コントロールがリストされています。

/ows/tr/browse	unauthenticated	Tr
	any_authenticated	Tr
/ows/tr/auth	unauthenticated	-
	any_other	-
	group TKStaff	Tr
	group Customer	PTr
/ows/admin/forall	unauthenticated	-
	any_other	-
	group Staff	Tr
/ows/admin/auth	unauthenticated	-
	any_other	-
	group AdminStaff	Tr

Customer と TKStaff は、予約と旅行計画の保守オブジェクトに関して、同じ特権を持っています。ただし、例外として、顧客の場合は、情報を暗号化し (プライバシー許可)、非トラステッド・インターネットを通して機密データ (クレジット・カード情報など) を提供する際、さらなるセキュリティーを確保する必要があります。

結論

この例で示したのは、以下を行うことができるシステムをデプロイする概念です。

- 機密情報を保護する。
- ユーザーを認証する。
- 機密情報へのアクセスを許可する。

さらに、システム認証ユーザーの識別は、WebSEAL と Oracle Web Server の両方に認識され、監査可能な、シングル・サインオン・ソリューションを提供するために使用されます。

第 39 章 Internet Content Adaptation Protocol (ICAP) のサポート

Internet Content Adaptation Protocol (ICAP) は、インターネットベースのコンテンツの処理を専用サーバーに任せて負荷を軽減できるように設計されています。ICAP により、リソースを解放し、各機能のインプリメント方法を標準化することができます。

WebSEAL などのプロキシ・サーバーは、クライアントの要求および応答を ICAP サーバーを介して渡すように構成できます。これらの ICAP サーバーは、特定の付加価値サービスを集中的に提供するため、効率を高めることができます。例えば、1 つの ICAP サーバーが言語翻訳を専門に処理すれば、多くの追加タスクを実行する Web サーバーよりも効率的です。

ICAP は「軽量の」HTTP に似たプロトコルです。ICAP クライアントは、HTTP ベース (HTML) のメッセージまたはコンテンツを ICAP サーバーに渡して適応させることができます。適応とは、関連付けられたクライアント要求または応答に対して、コンテンツ操作などの特定の付加価値サービスを提供することを意味します。

詳しくは、Request For Comments (RFC) 3507 - Internet Content Adaptation Protocol (ICAP) (<http://www.ietf.org/rfc/rfc3507.txt>) を参照してください。

この章で扱うトピックは以下のとおりです。

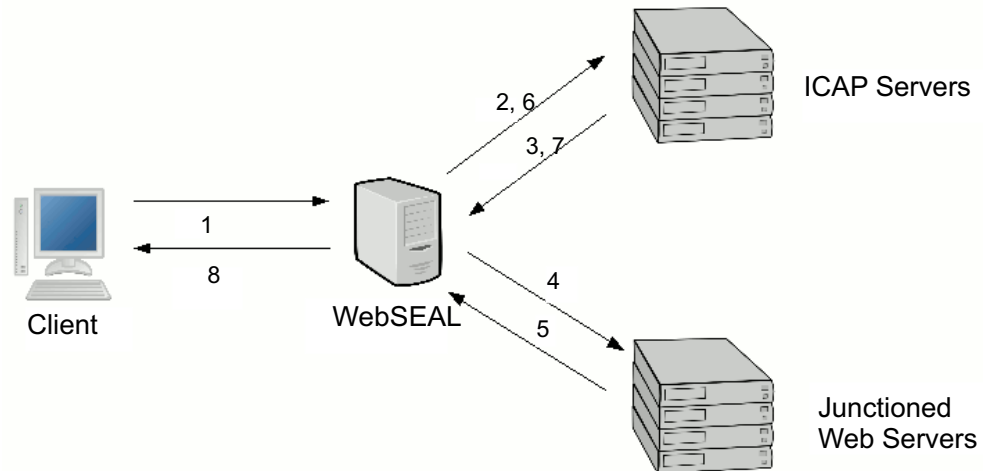
- 698 ページの『WebSEAL との ICAP の統合 - ワークフロー』
- 699 ページの『機能の有効範囲』
- 699 ページの『WebSEAL 内での ICAP サポートの構成』

例

その他の ICAP サーバー機能の例としては、ウィルス・スキャン、言語翻訳、コンテンツ・フィルタリングなどがあります。

WebSEAL との ICAP の統合 - ワークフロー

クライアント要求は、次の図に示す経路で処理されます。



1. クライアントが要求を WebSEAL に送信します。
2. WebSEAL が要求を ICAP サーバーに渡します。
3. ICAP サーバーは Internet Content Adaptation Protocol (ICAP) を使用して応答を生成し、WebSEAL に戻します。この応答は次のいずれかです。
 - a. チェーン内の他の ICAP サーバーまたはバックエンド Web サーバーに転送される、変更されたバージョンの要求。
 - b. クライアントに返信する必要がある、この要求に対する HTTP 応答。
 - c. WebSEAL によって処理されるエラー。
4. 変更された要求は、その後バックエンド Web サーバーに送信されます。
5. バックエンド Web サーバーによって応答が生成されます。
6. WebSEAL はその応答を ICAP サーバーに渡します。
7. ICAP サーバーが応答を生成し、WebSEAL に返信します。この応答は次のいずれかです。
 - a. 変更されたバージョンの応答。
 - b. エラー。
8. 応答がクライアントに送信されます。

注: 複数の ICAP サーバーが構成されている場合は、ステップ 2 から 3、および 6 から 7 が繰り返し行われます。

機能の有効範囲

以下の情報は、WebSEAL と ICAP の統合範囲を示しています。

WebSEAL は、以下をサポートします。

- 修正要求 (REQMOD)
- 応答要求 (RESPMOD)

WebSEAL は、以下に示す ICAP RFC 3507 のオプション局面をサポートしません。

- メッセージ・プレビュー
- プレビュー以外の「204 No Content」応答
- OPTIONS メソッド
- キャッシュ
- 認証
- 暗号化

注: 各局面について詳しくは、ICAP RFC 3507 (<http://www.ietf.org/rfc/rfc3507.txt>) を参照してください。

WebSEAL 内での ICAP サポートの構成

WebSEAL 内の ICAP サポートの構成には柔軟性があり、ICAP の介入を必要とするトランザクションのみが ICAP サーバーに送信されるようにすることができます。

アドミニストレータは、ICAP 処理を必要とするアプリケーションの構成と制御を行うことができます。WebSEAL 内での ICAP サポートの構成には次の 2 つの部分があります。

- 構成ファイル: ICAP サーバーの定義に使用します。
- 保護オブジェクト・ポリシー (POP): ICAP サーバーへのコールをトリガーするリソースの定義に使用します。

構成ファイル

[ICAP: <resource>] と呼ばれるスタンザ・エントリーが構成ファイルに追加されます。このスタンザ・エントリーを使用して、さまざまな ICAP リソースが定義されます。各リソースの構成要素は次のとおりです。

- ICAP サーバーの URL
- HTTP 要求または応答の処理で ICAP サーバーが使用されるかどうかを定義するトランザクション・リスト
- WebSEAL が ICAP サーバーからの応答を待つときの最大時間 (秒単位) を定義するタイムアウト値

詳しくは、「*IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス*」の **[ICAP:<resource>]** スタンザを参照してください。

注: スタンザ名に含まれる <resource> は、POP でのリソースの名前に対応します。構成ファイルでは、複数のリソースが指定されていることがあります。

例

```
[ICAP:resource_a]
URL = icap://icap_svr.tivoli.com:1344/
transaction = req
timeout = 120
[ICAP:resource_b]
URL = icap:///icap_svr.tivoli.com:1344/
transaction = rsp
timeout = 120
```

保護オブジェクト・ポリシー (POP)

保護オブジェクト・ポリシー (POP) を使用して、オブジェクト・スペースの適切な部分の定義済み ICAP リソースを使用可能にします。このメカニズムにより、ICAP 処理に追加の影響を与えるリソースを完全に制御できます。POP には、以下のものがが必要です。

- 「ICAP」という名前で作成される拡張属性。
- 設定済み ICAP リソースのいずれかの名前に一致する値。

特定のオブジェクトまたは要求を処理するために複数の ICAP サーバーが必要とされる場合は、同じ名前の属性を複数作成することができます。

以下に POP の例を示します。

```
pdadmin sec_master> pop show ICAPPop attribute ICAP
ICAP
    resource_a
    resource_b
```

注: resource_a と resource_b は、構成スタンザ [ICAP:resource_a] と [ICAP:resource_b] に対応します。

第 11 部 付録

付録 A. 構成ファイルを変更するためのガイドライン

Security Access Manager の構成ファイルに変更を加える場合に役立つガイドラインとして、以下のものが提供されています。ガイドラインは以下のカテゴリに分けられます。

- 『一般ガイドライン』
- 『デフォルト値』
- 704 ページの『ストリング』
- 704 ページの『定義済みストリング』
- 704 ページの『ファイル名』
- 705 ページの『整数』
- 705 ページの『ブール値』

一般ガイドライン

構成の設定を変更する場合は、以下の一般ガイドラインを使用します。

- どの構成ファイルにおいても、スタンザには順序依存性またはロケーション依存性はありません。
- スタンザ・エントリーは、必須またはオプションとしてマークされています。あるエントリーが必要な場合は、そのエントリーには有効なキーと値が含まれている必要があります。
- 構成ファイル内のキーの名前は変更しないでください。キーの名前を変更すると、サーバーに対して予測不能の結果を招くおそれがあります。
- スタンザ・エントリーとキー名は大文字小文字を区別します。例えば、`usessl` と `UseSSL` は、異なるエントリーとして扱われます。
- キーの名前にスペースを使用することはできません。
- キーと値の組を `キー = 値` という形式で表した場合、等号 (=) の左右のスペースは必須ではありませんが、使用をお勧めします。
- スタンザ・エントリーの末尾にある印字されない文字 (タブ、復帰、および改行など) は無視されます。印字されない文字とは、その 10 進値が 32 よりも小さい ASCII 文字をいいます。

デフォルト値

デフォルトの構成設定値を変更する場合は、以下のガイドラインを使用してください。

- 作成または変更する値が多い場合は、構成プログラムを使用します。これらのスタンザまたは値は、手動で編集しないでください。
- 一部の値は、構成中に自動的に設定されます。これらの値は、構成後のサーバーの初期化に必要になります。

- スタンザ・エントリーのデフォルト値は、サーバー構成に応じて異なる場合があります。一部のキーと値の組は、サーバーによっては適用できない場合があります、当該サーバーのデフォルトの構成ファイルから省略されています。

ストリング

一部の値は、ストリング値を受け入れます。構成ファイルを手動で編集する場合は、以下のガイドラインを使用して、ストリングを要求する構成設定値を変更してください。

- ストリング値は、ローカル・コード・セットの一部となっている文字であることが想定されています。
- 使用可能なストリング文字のセットに対しては、追加制限または異なる制限が課せられることがあります。例えば、多くのストリングは ASCII 文字に制限されています。制限事項については、それぞれのスタンザ・エントリーの説明を参照してください。
- 値にスペースまたは複数の語を使用する場合は、二重引用符が必要になる場合がありますが、常に必要になるわけではありません。不明の場合は、各スタンザ・エントリーの説明または例を参照してください。
- ユーザー・レジストリー関連のストリング値の長さに最小値と最大値の制限がある場合は、それらは基本レジストリーの制限によるものです。例えば、Active Directory の場合の最大長は、英数字で 256 文字です。

定義済みストリング

値によってはストリング値を受け入れるものもありますが、その値は定義済みのストリング・セットの中の 1 つである必要があります。構成ファイルを手動で編集する場合は、入力するストリング値は、有効な定義済みのストリング値の 1 つに一致している必要があります。

例えば、**[aznapi-configuration]** stanza セクションには、以下のエントリーが含まれています。

```
auditcfg = {azn|authn|mgmt}
```

auditcfg で予期される値は、azn、authn、または mgmt です。これ以外の値はすべて無効であり、エラーになります。

ファイル名

一部の値はファイル名です。値としてファイル名を想定している各スタンザ・エントリーの場合は、ファイル名を LMI に表示されているとおりに指定してください。ディレクトリー・パス情報は含めないでください。

整数

多くのスタンザ・エントリーでは、エントリーの値が整数として表されていることを想定しています。整数でエントリーを定義する場合、次のガイドラインを考慮してください。

- 整数値を取るスタンザ・エントリーは、その整数値が有効な範囲内にあることを想定しています。範囲は、最小 値と最大 値で示されます。

例えば、**[logging]** スタンザでは、**logflush** スタンザ・エントリーの最小値は 1 秒で、最大値は 600 秒です。

- 一部のエントリーでは、整数値は正である必要があり、最小値は 1 になります。他のエントリーの場合は、整数の最小値に 0 を使用できます。

整数値を 0 に設定する場合には注意が必要です。例えば、整数値が 0 の場合は、そのスタンザ・エントリーによって制御される機能が使用不可になる場合があります。例えば、**[ivacld]** スタンザでは、エントリーが `tcp-req-port = 0` の場合は、ポート番号が使用不可になります。または、整数値が 0 の場合は、数に制限がないことを示す場合があります。例えば、**[ldap]** スタンザでは、エントリーが `max-search-size = 0` の場合は、検索サイズの最大値に制限がないことを示しています。

- 整数値を要求するエントリーによっては、使用可能な最大数に対して Security Access Manager が上限を課していないものがあります。例えば、**[ldap]** スタンザの `timeout = number` といったタイムアウト関連の値には、通常、最大値はありません。

このタイプのエントリーの場合は、最大数は、整数のデータ型に割り振られているメモリー・サイズによってのみ制限を受けます。この最大数は、オペレーティング・システムの型に基づいて変化します。整数に 4 バイトを割り振っているシステムの場合の最大数は 2147483647 です。

ただし、アドミニストレーターとしては、設定しようとしている値にとって最も論理的な値を表す数を使用してください。

ブール値

多くのスタンザ・エントリーは、ブール値を表します。Security Access Manager は、ブール値 `yes` および `no` を認識します。

構成ファイル内の一部のエントリーは、他のサーバーおよびユーティリティによって読み取られます。例えば、**[ldap]** スタンザ内の多くのエントリーは、LDAP クライアントによって読み取られます。これら他のプログラムの中には、以下に示した追加のブール文字を認識するものもあります。

- `yes` または `true`
- `no` または `false`

ブランク値も含めた `yes|true` 以外の値は、`no|false` と解釈されます。

認識されるブール・エントリーは、スタンザ・エントリーごとにリストされています。どのような場合に `true` または `false` も認識されるのかについて判断する場合は、個々の説明を参照してください。

付録 B. コマンド・リファレンス

この付録では、WebSEAL タスク固有の **pdadmin** コマンドのサブセットについて説明します。

「*IBM Security Access Manager for Web: Command Reference*」も参照してください。

コマンド	説明
709 ページの『help』	pdadmin コマンドとオプションについてのシステム・ヘルプを取得します。
710 ページの『server list』	登録済み Security Access Manager サーバーをすべてリストします。
711 ページの『server task add』	既存の WebSEAL Junction に追加のバックエンド・アプリケーション・サーバーを追加します。
714 ページの『server task cache flush all』	HTML 文書キャッシュをフラッシュします。
716 ページの『server task cluster restart』	必要に応じて、WebSEAL クラスター全体に構成変更を適用し、再始動します。
717 ページの『server task create』	WebSEAL junction ポイントを作成します。
726 ページの『server task delete』	WebSEAL junction ポイントを削除します。
727 ページの『server task dynurl update』	動的 URL 構成ファイルを再ロードします。
728 ページの『server task help』	特定の server task コマンドについての詳細なヘルプ情報をリストします。
730 ページの『server task jmt』	junction マッピング・テーブル・データをクリアまたはロードします。
731 ページの『server task list』	WebSEAL サーバーまたは WebSEAL インスタンス上のすべての Junction ポイントをリストします。
733 ページの『server task offline』	この Junction に配置されているサーバーをオフライン操作状態にします。
735 ページの『server task online』	この Junction に配置されているサーバーをオンライン操作状態にします。
737 ページの『server task refresh all_sessions』	指定したユーザーのすべてのセッションの資格情報をリフレッシュします。
738 ページの『server task reload』	junction マッピング・テーブルをデータベースから再ロードします。
739 ページの『server task remove』	指定したインストール済みの WebSEAL サーバーまたは WebSEAL インスタンスを、WebSEAL Junction ポイントから除去します。
742 ページの『server task show』	指定した WebSEAL Junction についての詳細情報を表示します。

コマンド	説明
744 ページの『server task terminate all_sessions』	特定のユーザーのすべてのユーザー・セッションを終了します。
746 ページの『server task terminate session』	セッション ID を指定して、ユーザー・セッションを終了します。
747 ページの『server task throttle』	この Junction に配置されているサーバーをスロットル操作状態にします。
749 ページの『server task virtualhost add』	既存の仮想ホスト Junction に、追加のインストール済み WebSEAL サーバーまたは WebSEAL インスタンスを追加します。
752 ページの『server task virtualhost create』	仮想ホスト junction を作成します。
760 ページの『server task virtualhost delete』	仮想ホスト junction を削除します。
761 ページの『server task virtualhost list』	すべての構成済み仮想ホスト junction をラベル名でリストします。
763 ページの『server task virtualhost offline』	この仮想ホスト Junction に配置されているサーバーをオフライン操作状態にします。
765 ページの『server task virtualhost online』	この仮想ホスト Junction に配置されているサーバーをオンライン操作状態にします。
767 ページの『server task virtualhost remove』	仮想ホスト junction から指定したサーバーを除去します。
770 ページの『server task virtualhost show』	指定した仮想ホスト Junction についての詳細情報を表示します。
772 ページの『server task virtualhost throttle』	この仮想ホスト Junction に配置されているサーバーをスロットル操作状態にします。

構文ステートメントの読み方

リファレンス文書では、構文の定義に以下の特殊文字を使用しています。

- [] 任意指定のオプションを表します。ブラケットで囲まれていないオプションは必須です。
- ... 直前のオプションに対して複数の値が指定できることを示します。
- | 相互排他的な情報を示します。セパレーターの左方にあるオプションも、右方にあるオプションも使用できます。しかし、単一のコマンド使用中に両方のオプションを使用することはできません。
- { } オプションのいずれかが必須の場合に、相互排他的なオプションのセットを区切ります。オプションが任意指定の場合は、ブラケット ([]) で囲まれません。
- ¥ コマンド行が次の行に折り返すことを表します。継続文字です。

各コマンドのオプションは、『オプション』セクションにアルファベット順にリストされています。オプションを特定の順序で使用する必要がある場合は、この順序が構文に示されています。

help

pdadmin コマンドとオプションについてのシステム・ヘルプを取得します。

このコマンドの使用には、ログインや認証は必要ありません。

構文

help {*topic* | *command*}

オプション

topic ヘルプが必要なヘルプ・コマンド・トピックを指定します。

command

ヘルプが必要な各種のコマンドを指定します。

戻りコード

- 0 コマンドが正常に完了しました。
- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

- 以下の例に、ヘルプ・トピックおよびヘルプ・コマンドをリストします。

```
pdadmin> help
```

出力は以下のようになります。

```
Type 'help <topic>' or 'help <command>' for more information
```

```
Topics:
acl
action
admin
authzrule
config
context
domain
errtext
exit
group
help
login
logout
object
objectspace
policy
pop
quit
rsrc
rsrccred
rsrccgroup
server
user
```

```
Miscellaneous Commands:  
exit  
help  
quit
```

- 以下は、`action` または `action create` を指定した場合に使用可能なオプションおよび説明のリスト例です。

```
pdadmin> help action
```

または

```
pdadmin> help action create
```

出力は以下のようになります。

```
action create <action-name> <action-label> <action-type>  
Creates a new ACL action definition  
action create <action-name> <action-label> <action-type> <action-group-name>  
Creates a new ACL action definition in a group  
...
```

server list

登録済み Security Access Manager サーバーをすべてリストします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server list

説明

登録済み Security Access Manager サーバーをすべてリストします。**server list** コマンド以外のすべてのサーバー・コマンドでは、このコマンドの出力で表示されるのと厳密に同じ形式でサーバー名を入力する必要があります。

オプション

なし。

戻りコード

- 0 コマンドが正常に完了しました。
- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、Security Access Manager コンポーネントが許可サーバーである場合、登録されているすべてのサーバーがリストされます。

```
pdadmin> server list
```

出力は以下のようになります。

```
ivacld-topserver
ivacld-server2
ivacld-server3
ivacld-server4
```

server task add

server task add コマンドは、既存の WebSEAL junction にさらにバックエンド・アプリケーション・サーバーを追加します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name add -h host_name [options]  
junction_point
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

options **server task add** コマンドに使用できるオプションを指定します。これらのオプションには、次のようなものがあります。

-D "dn"

バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張し、SSL 上での相互認証を提供するものです。例えば、*www.example.com* の証明書の DN は次のようになります。

```
"CN=WWW.EXAMPLE.COM,OU=Software,O=example.com¥, Inc,L=Austin,  
ST=Texas,C=US"
```

このオプションは、`ssl` タイプまたは `sslproxy` タイプとして作成された Junction でのみ有効です。

-H *host_name*

プロキシ・サーバーの DNS ホスト名または IP アドレスを指定します。*host_name* の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

`www.example.com`

このオプションは、`tcpproxy` タイプまたは `sslproxy` タイプとして作成された Junction で使用されます。

-i WebSEAL サーバーが URL の大/小文字を区別しないことを表します。このオプションは、`tcp` タイプまたは `ssl` タイプとして作成された Junction で使用されます。

-P *port*

バックエンド・サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。このオプションは、`tcp` タイプまたは `ssl` タイプとして作成された Junction で使用されます。

-P *port*

`tcpproxy` タイプまたは `sslproxy` タイプとして作成されたプロキシ Junction では、このオプションは HTTP プロキシ・サーバーの TCP ポート番号を指定します。デフォルト値は 7138 です。

port の場合は、任意の有効なポート番号を使用します。有効なポート番号とは、TCP/IP で使用でき、他のアプリケーションで現在使用されていない任意の正数です。デフォルトのポート番号値を使用するか、現在使用されていない、1000 より大きいポート番号を使用します。

このオプションは、`mutual` の Junctions でバックエンドのサード・パーティー・サーバーの HTTPS ポートを指定する場合にも有効です。

-q *url* バックエンド Windows サーバーの場合は必須オプションです。
query_contents スクリプトの相対パスを指定します。デフォルトでは、Security Access Manager は `/cgi_bin` サブディレクトリーでこのスクリプトを探します。これとは別のディレクトリーに入っている場合、または **query_contents** ファイルの名前が変更されている場合は、このオプションを使用して、該当のファイルへの新しい URL を WebSEAL に指示してください。

このオプションは、`tcp` タイプまたは `ssl` タイプとして作成された Junction で使用されます。

-u *uuid*

-s オプションを使用するステートフル Junction を介してこのバックエンド・サーバーが WebSEAL に接続されたときの、UUID を指定します。このオプションは、`tcp` タイプまたは `ssl` タイプとして作成された Junction で使用されます。

-v *virtual_hostname*

バックエンド・サーバー上で表される仮想ホスト名を指定します。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。バックエンド junction サーバーの仮想インスタンスに結合するため、そのサーバーがホスト名ヘッダーを必要とする場合にこのオプションを使用します。ブラウザからのデフォルト HTTP ヘッダー要求は、バックエンド・サーバーに複数の名前と複数の仮想サーバーがあることを認識しません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

-V *virtual_hostname*

バックエンド・サーバー上で表される仮想ホスト名。このオプションは、バックエンド・サーバーにおける仮想ホストのセットアップを支援するものです。このオプションは、mutual の junction のみで使用され、HTTPS 要求に使用される仮想ホストに対応します。

バックエンド junction サーバーの仮想インスタンスの 1 つに junction しているので、そのサーバーがホスト名ヘッダーの受信を予期している場合は、**-V** を使用してください。ブラウザからのデフォルトの HTTPS ヘッダー要求は、バックエンド・サーバーが複数の名前と複数の仮想サーバーを持っていることを認識していません。仮想ホストとしてセットアップされているバックエンド・サーバー宛ての要求に追加のヘッダー情報を組み込むように、WebSEAL を構成する必要があります。

-w Microsoft Windows ファイル・システム・サポートを示します。

このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

-h *host_name*

必須オプション。ターゲット・バックエンド・アプリケーション・サーバーの DNS ホスト名または IP アドレスを指定します。host_name の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

www.example.com

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、/WebSEAL/host_name-instance_name/junction_point オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

注: このコマンドは、WebSEAL がインストールされている場合にのみ使用可能です。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

注: 既存の Junction へのサーバーの追加方法について詳しくは、「*IBM Security Access Manager for Web WebSEAL 管理ガイド*」を参照してください。

例

以下の例では、WS1 という名前の WebSEAL サーバーに APP1 という名前のバックエンド・サーバーへの新規 Junction を作成し、同じ Junction ポイントに対して、別の APP2 という名前のバックエンド・サーバーを追加します。

```
pdadmin> server task default-webseald-WS1 create -t tcp -h APP1 -s /mnt
```

```
pdadmin> server task default-webseald-WS1 add -h APP2 /mnt
```

関連項目

- 717 ページの『server task create』
- 726 ページの『server task delete』
- 739 ページの『server task remove』
- 742 ページの『server task show』

server task cache flush all

server task cache flush all コマンドは HTML 文書キャッシュをフラッシュします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task instance_name-webseald-host_name cache flush all

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
/WebSEAL/host_name-instance_name/ オブジェクトを統括する ACL 内で *s* (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

注: このコマンドは、WebSEAL がインストールされている場合にのみ使用可能です。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

注

WebSEAL コンテンツのキャッシングについて詳しくは、「*IBM Security Access Manager for Web WebSEAL 管理ガイド*」を参照してください。

例

以下の例では、すべての Web 文書のキャッシュがフラッシュされます。

```
pdadmin> server task default-webseald-abc.ibm.com cache flush all
```

関連項目

なし。

server task cluster restart

server task cluster restart コマンドはクラスター全体に構成変更を適用し、更新したサーバーを再始動します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *server_name* cluster restart [-ripple|-status]

説明

-status オプションが使用されている場合、コマンドにより、最新のクラスター再始動の状況更新が提供されます。

-status オプションが使用されていない場合、コマンドにより、クラスター内の各サーバーでマスターの構成更新が調べられます。必要に応じて、サーバーはその構成データを更新し、マスター構成情報と同期させます。更新が適用されると、サーバーは再始動されます。マスター・サーバーは、クラスター内のスレーブ・サーバーがすべて更新され、必要に応じて再始動された後に、再始動されます。

注: このコマンドは、構成済みのクラスター・マスター・サーバーでのみ使用可能です。

オプション

server_name

構成データベースのあるマスター許可サーバーの名前を指定します。

-ripple クラスター内の各 WebSEAL サーバーを、同時ではなく順番に再始動する必要があることを示します。

-status クラスター再始動の進行状況をモニターします。このオプションは、クラスターの現在の状況に基づいて、以下のいずれかのメッセージを返します。

DPWAD0444I The cluster has been restarted.

DPWAD0443I The cluster is in the process of being restarted.

許可

このコマンドにアクセスする必要があるユーザーおよびグループは、マスターの /WebSEAL/host_name-instance_name/ オブジェクトを管理する ACL で **s** (サーバー管理) 許可が付与されている必要があります。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

注:

サーバーがクラスターのマスターとして構成されていない場合は、エラーが返されます。このコマンドを実行する前に、適切な **[cluster]** スタンザ・エントリーを使用して、クラスターが正しく構成されていることを確認する必要があります。詳しくは、「IBM Security Web Gateway Appliance: Web リバース・プロキシ・スタンザ・リファレンス」の **[cluster]** スタンザを参照してください。

-ripple と **-status** の 2 つのオプションを同時に使用することはできません。**-ripple** オプションはクラスター再始動の開始時に使用することができ、一方、**-status** オプションは最新のクラスター再始動要求の進行状況をモニターします。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

デフォルト値

デフォルトでは、クラスターは同時に再始動されます。

例

以下の例では、クラスターは順番に再始動されます。このコマンドは、クラスター・マスター・サーバー (この例では **default-webseald-master.ibm.com**) で実行する必要があります。

```
server task default-webseald-master.ibm.com cluster restart -ripple
```

次の例では、要求があった後にいつでもこのコマンドを使用して、クラスター再始動の進行状況をモニターすることができます。

```
server task default-webseald-master.ibm.com cluster restart -status
```

server task create

server task create コマンドは WebSEAL Junction ポイントを作成します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

ローカル Junction の場合

```
server task instance_name-webseald-host_name create -t type [options]  
junction_point
```

ローカルでない Junction の場合

```
server task instance_name-webseald-host_name create -t type -h host_name  
[options] junction_point
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

options **server task create** コマンドに使用できるオプションを指定します。オプションには、次のようなものがあります。

-a *address*

WebSEAL がターゲットのバックエンド・サーバーと通信するときに使用するローカル IP アドレスを指定します。このオプションを指定しない場合、WebSEAL ではオペレーティング・システムにより指定されるデフォルトのアドレスを使用します。

特定の Junction に関してアドレスを指定すると、Junction サーバーとのすべての通信に関してこのローカル・アドレスにバインドするように、WebSEAL が変更されます。

-A LTPA (Lightweight Third-Party Authentication) メカニズム Junction を使用可能または使用不可にします。このオプションには **-F** オプションと **-Z** オプションが必要です。 **-A**、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。

このオプションは、*local* タイプを除くすべての Junction に対して有効です。

-2 このオプションは、**-A** オプションとともに使用でき、LTPA バージョン 2 Cookie (LtpaToken2) が使用されることを指定します。**-2** オプションを指定しない **-A** オプションは、LTPA バージョン 1 Cookie (LtpaToken) が使用されることを指定します。

-b *BA_value*

WebSEAL サーバーがバックエンド・サーバーに HTTP BA 認証情報を渡す方法を定義します。認証情報は以下の値のいずれか 1 つです。

- *filter* (デフォルト)

- ignore
- supply
- gso

このオプションは、local タイプを除くすべての Junction に対して有効です。

- B** WebSEAL が基本認証ヘッダー情報を使用してバックエンド・サーバーに対して認証し、SSL を介した相互認証を提供することを示します。このオプションには **-U** オプションと **-W** オプションが必要です。

このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-c header_type

Junction を介して HTTP ヘッダーに Security Access Manager のクライアント識別を挿入します。header_type 引数には、以下の Security Access Manager HTTP ヘッダー・タイプを任意に組み合わせて指定できます。

- {iv-user|iv-user-1}
- iv-groups
- iv-creds
- all

ヘッダー・タイプはコンマで区切る必要があり、タイプ間にスペースを入れることはできません。例: -c iv_user,iv_groups

-c all を指定することは、**-c iv-user,iv-groups,iv-creds** を指定することと同じです。

このオプションは、local タイプを除くすべての Junction に対して有効です。

- C** フロントエンド WebSEAL サーバーからバックエンド WebSEAL サーバーへのシングル・サインオンを表します。**-C** オプションは相互認証ではありません。

このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-D "dn"

バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張し、SSL 上での相互認証を提供するものです。例えば、www.example.com の証明書の DN は次のようになります。

```
"CN=WWW.EXAMPLE.COM,OU=Software,O=example.com¥,Inc,L=Austin,ST=Texas,C=US"
```

このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-e encoding_type

Junction に対して HTTP ヘッダーを生成するときに使用するエンコード方式を指定します。このエンコード方式は、**-c Junction** オプシ

ョンと tag-value の両方を指定して生成されたヘッダーに適用されます。エンコード方式として以下の値がサポートされます。

utf8_bin

WebSEAL は UTF-8 でヘッダーを送信します。

utf8_uri

WebSEAL は UTF-8 でヘッダーを送信しますが、URI でもエンコードが行われます。この動作がデフォルトの動作です。

lcp_bin

WebSEAL は、WebSEAL サーバーのローカル・コード・ページのヘッダーを送信します。

lcp_uri

WebSEAL は、WebSEAL サーバーのローカル・コード・ページのヘッダーを送信しますが、URI でもエンコードが行われます。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-f 既存の junction を強制的に置き換えます。

このオプションは、いずれの Junction タイプで作成された Junction でも使用されます。

-F *keyfile*

LTPA Cookie データの暗号化に使用される鍵ファイルの名前を指定します。

-F オプションには **-A** オプションと **-Z** オプションが必要です。

-A、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-H *host_name*

プロキシ・サーバーの DNS ホスト名または IP アドレスを指定します。**-P** オプションもプロキシ・サーバー Junction をサポートします。*host_name* の有効値には、すべての有効な IP ホスト名が含まれます。例えば、次のとおりです。

proxy.www.example.com

このオプションは、tcpproxy タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-i WebSEAL Junction が URL の大/小文字を区別しないことを表します。大/小文字の区別をしない Junction への要求を正しく認可するため、WebSEAL では、URL の小文字バージョンに対して許可検査を実行します。例えば、Windows オペレーティング・システム上で実行されている Web サーバーは、INDEX.HTM および index.htm への要求を同一ファイルへの要求として扱います。

このような Web サーバーへの Junction は、**-i** オプションまたは **-w** オプションを指定して作成する必要があります。Junction ポイント下のオブジェクトに付加する ACL または POP は、小文字のオブジェクト名を使用する必要があります。/junction/index.htm に付加された ACL は、**-i** オプションまたは **-w** オプションが使用されている場合は、以下の要求すべてに適用されます。

```
/junction/INDEX.HTM  
/junction/index.htm  
/junction/InDeX.HtM
```

このオプションは、local タイプを除くすべての Junction に対して有効です。ローカル junction は Win32 プラットフォームでのみ大文字小文字の区別がありません。他のすべてのプラットフォームでは、大文字小文字の区別があります。

- I -j** オプションを使用するときに固有の Set-Cookie ヘッダー名属性により、要求内のサーバー相対 URL が確実に変更されるようにします。

このオプションは、local タイプを除くすべての Junction に対して有効です。

- j** スクリプト生成のサーバー相対 URL を処理する Junction 識別を Cookie に入れて提供します。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-J trailer,inhead,onfocus,xhtml10

junction Cookie JavaScript ブロックをコントロールします。

-J trailer は、junction Cookie JavaScript をバックエンド・サーバーから戻される HTML ページの (前ではなく) 後ろに付加するときに使用します。

-J inhead は、HTML 4.01 準拠のために <head> タグと </head> タグの間に JavaScript ブロックを挿入するときに使用します。

-J onfocus は、JavaScript で onfocus イベント・ハンドラーを使用して、正しい junction Cookie が「複数の junction と複数のブラウザー・ウィンドウ」シナリオで使われるようにするときに使用します。

-J xhtml10 は、HTML 4.01 および XHTML 1.0 準拠である JavaScript ブロックを挿入するときに使用します。

このオプションについて詳しくは、495 ページの『junction Cookie JavaScript ブロックのコントロール』を参照してください。

- k** Junction サーバーに WebSEAL セッションの Cookie を送信します。デフォルトでは、Cookie はサーバーに送信される要求から除去されます。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-K "key_label"

WebSEAL がバックエンド・サーバーに提示する必要があるクライアント個人証明書の鍵ラベルを指定します。このオプションを使用すると、Junction サーバーがクライアント証明書を使用して WebSEAL サーバーを認証できます。

このオプションは、ssl タイプおよび sslproxy タイプとして作成された Junction でのみ有効です。

-I percent

ワーカー・スレッドの消費量のソフト限界を定義します。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-L percent

ワーカー・スレッドの消費量のハード限界を定義します。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-n

ドメイン外の Cookie の名前を変更しないことを表します。クライアント・サイド・スクリプトが Cookie の名前に依存する場合に使用します。

デフォルトでは、JMT 内に Junction がリストされるか、または -j Junction オプションが使用されている場合、Junction から戻され AMWEBJCT!junction_point の前に付加されるドメイン外の Cookie の名前は、WebSEALにより変更されます。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-P port

バックエンドのサード・パーティー・サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-P port

tcpproxy タイプまたは sslproxy タイプとして作成されたプロキシ Junction では、このオプションは HTTP プロキシ・サーバーの TCP ポート番号を指定します。-H オプションが使用される場合は、-P オプションが必須です。

このオプションは、mutual の Junctions でバックエンドのサード・パーティー・サーバーの HTTPS ポートを指定する場合にも有効です。

-q path

バックエンド Windows サーバーの場合は必須オプションです。

query_contents スクリプトの相対パスを指定します。デフォルトでは、Security Access Manager は /cgi_bin ディレクトリで query_contents スクリプトを探します。これとは別のディレクトリ

ーに入っている場合、または `query_contents` のファイルの名前が変更されている場合は、該当のファイルへの新しい URL が、このオプションにより WebSEAL に示されます。

このオプションは、`local` タイプを除くすべての Junction に対して有効です。

- r** Junction を介して HTTP ヘッダーに着信 IP アドレスを挿入します。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- R** 要求の進行は許可しますが、HTTP ヘッダー内の Junction にルール違反の理由を通知します。**-R** オプションが使用されずにルール違反が生じると、WebSEAL では、要求を進行できません。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- s** Junction がステートフル・アプリケーションをサポートすることを表します。デフォルトでは、`junction` はステートフルではありません。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- S** フォーム・シングル・サインオン構成ファイルの名前を指定します。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- T {resource | resource_group}**
リソースまたはリソース・グループの名前を指定します。このオプションは **-b gso** オプションが使用される場合にのみ必須です。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- u uuid**
ステートフル Junction (**-s** オプション) を使用して WebSEAL に接続されている、バックエンド・サーバーの UUID (Universally Unique Identifier) を指定します。このオプションは、`local` タイプを除くすべての Junction に対して有効です。
- U "user_name"**
WebSEAL サーバー・ユーザー名を指定します。このオプションには **-B** オプションと **-W** オプションが必要です。WebSEAL は、基本認証ヘッダー情報を使用してバックエンド・サーバーに対して認証し、SSL を介した相互認証を提供します。このオプションは、`ssl` タイプまたは `sslproxy` タイプとして作成された Junction でのみ有効です。
- v virtual_hostname[:HTTP-port]**
バックエンド・サーバーの仮想ホスト名を指定します。このオプションでは、同じ Web サーバーによる複数の仮想ホストの処理がサポートされます。バックエンド Junction サーバーが、そのサーバーの DNS 名とは異なるホスト名ヘッダーを予期している場合は、**-v** を使用します。このオプションは、`local` タイプを除くすべての Junction に対して有効です。`mutual` の `junction` では、この値は HTTP 要求に使用される仮想ホストに対応します。

-V *virtual_hostname[:HTTPS-port]*

バックエンド・サーバーの仮想ホスト名を指定します。このオプションでは、同じ Web サーバーによる複数の仮想ホストの処理がサポートされます。バックエンド Junction サーバーが、そのサーバーの DNS 名とは異なるホスト名ヘッダーを予期している場合は、**-V** を使用します。このオプションは、mutual の junction のみで使用され、HTTPS 要求に使用される仮想ホストに対応します。

-w Microsoft Windows ファイル・システム・サポートを示します。このオプションでは、**-i** Junction オプションにより提供される機能がすべて提供されますが、Windows のファイル名のエイリアスとして解釈される可能性のあるファイル名を含む要求は認可されません。このオプションは、local タイプを除くすべての Junction に対して有効です。ローカルの Junction では、Windows 上で Windows のファイル名の別名を含む URL は禁止されていますが、他のプラットフォームではそのような URL が許可されています。

-W *"password"*

WebSEAL サーバー・パスワードを指定します。このオプションには **-B** オプションと **-U** オプションが必要です。WebSEAL は、基本認証ヘッダー情報を使用してバックエンド・サーバーに対して認証し、SSL を介した相互認証を提供します。このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-x 透過パス Junction を作成します。

このオプションは、local タイプを除くすべての Junction に対して有効です。

-Y junction の Tivoli Federated Identity Manager シングル・サインオン (SSO) を使用可能にします。

注: このオプションを使用する前に、まず、junction 全体で Tivoli Federated Identity Manager シングル・サインオンをサポートするように WebSEAL 構成ファイルを構成する必要があります。

-Z *keyfile_pwd*

LTPA Cookie データの暗号化に使用される鍵格納ファイルのパスワードを指定します。このオプションには **-A** オプションと **-F** オプションが必要です。**-A**、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。このオプションは、local タイプを除くすべての Junction に対して有効です。

-h *host_name*

ローカルでない Junction の場合は必須オプションです。ターゲット・サーバーの DNS ホスト名または IP アドレスを指定します。このオプションは、ローカルでない Junction に対してのみ有効です。ローカルの Junction にはホスト名は必要ありません。host_name の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

www.example.com

-t type 必須オプション。Junction のタイプを指定します。以下のタイプのいずれか 1 つであることが必要です。

- tcp
- tcpproxy
- ssl
- sslproxy
- local

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/junction_point` オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

注:

統計の収集について詳しくは、「*IBM Security Access Manager for Web: Troubleshooting Guide*」を参照してください。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。ただし、コマンドが正常に送信された後でも、WebSEAL サーバーが正常にコマンドを実行できずにエラー・メッセージが戻されることがあります。
- 1 コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

- 以下の例 (1 行で入力します) では、default-webseald-cruz WebSEAL サーバー上に基本的な WebSEAL Junction /pubs が作成されます。Junction タイプは TCP、ホスト名は doc.tivoli.com です。

```
pdadmin> server task default-webseald-cruz create -t tcp  
-h doc.tivoli.com /pubs
```

出力は以下のようになります。

```
Created junction at /pubs
```

- 以下の例 (1 行で入力します) では、/myjunction Junction のワーカー・スレッドの消費を Junction 単位を基準にしてソフト・スレッド制限は 60 で、ハード・スレッド制限は 80 で制限します。

```
pdadmin> server task default-webseald-cruz create -t tcp  
-h cruz.dallas.ibm.com -l 60 -L 80 /myjunction
```

関連項目

711 ページの『server task add』
『server task delete』
739 ページの『server task remove』
742 ページの『server task show』

server task delete

server task delete コマンドは WebSEAL Junction ポイントを削除します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name delete junction_point*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/junction_point* オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー `default-webseald-abc.ibm.com` から、Junction ポイント `/pubs` を削除します。

```
pdadmin> server task default-webseald-abc.ibm.com delete /pubs
```

関連項目

711 ページの『`server task add`』

717 ページの『`server task create`』

739 ページの『`server task remove`』

742 ページの『`server task show`』

server task dynurl update

server task dynurl update コマンドは、動的 URL 構成ファイルを再ロードします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **dynurl update**

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。

webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
`/WebSEAL/host_name-instance_name/` オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、動的 URL 構成ファイルを再ロードします。

```
pdadmin> server task default-webseald-abc.ibm.com dynurl update
```

関連項目

なし。

server task help

特定の **server task** コマンドについての詳細なヘルプ情報をリストします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name help task
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が default で、WebSEAL サーバー がインストールされているホストのマシン名が abc.ibm.com である場合、完全な WebSEAL サーバー名は default-webseald-abc.ibm.com になります。

追加の WebSEAL インスタンスが構成され、web2 という名前が付けられると、完全な WebSEAL サーバー名は web2-webseald-abc.ibm.com になります。

task 指定したタスクについて、コマンド構文、説明および有効なオプションなどの詳細なヘルプをリストします。

許可

特別な認可は不要です。

戻りコード

- 0 コマンドが正常に完了しました。
- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

- 以下の例では、abc.ibm.com WebSEAL サーバーで **server task add** コマンドのヘルプを要求した後の出力を表示しています。

```
pdadmin> server task default-webseald-abc.ibm.com help add
```

出力は以下のようになります。

```
Command:
add <options> <junction point>
Description:
Adds an additional server to a junction
Usage:
TCP and SSL Junction Flags
-iServer treats URLs as case insensitive.
-h <hostname>Target host (required flag).
-p <port>TCP port of server.
Default is 80 for TCP junctions
443 for SSL junctions.
-H <hostname>Proxy hostname.
-P <port>Port of proxy server.
-D <"DN">The Distinguished Name of the server
-q <relative url> URL for query_contents script.
-u <UUID>(stateful junctions only).
-v <hostname>Virtual hostname for server.
-wWin32 file system support.
-jScripting support for junction.
Common Flags
<junction point>Where to create the junction
```

- 以下の例では、abc.ibm.com WebSEAL サーバーで **server task create** コマンドのヘルプを要求した後の出力を表示しています。

```
pdadmin> server task default-webseald-abc.ibm.com help create
```

出力は以下のようになります。

```
Command:
create -t <type> <options> <junction point>
Description:
Creates a new junction
Usage:
create -t <type> <options> <junction point>

TCP and SSL Junction Flags
...
Common Flags
-t <type>Type of junction.
One of: tcp, tcpproxy, ssl, sslproxy, local.
-fForce the creation: overwrite existing junction.
-RWebSEAL will send the Boolean Rule Header to these
junctions when a rule failure reason is provided.
<junction point>Where to create the junction
```

関連項目

709 ページの『help』

server task jmt

server task jmt コマンドは、junction マッピング・テーブル・データをクリアまたはロードします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **jmt load**

server task *instance_name-webseald-host_name* **jmt clear**

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

jmt clear

junction マッピング・テーブル・データをクリアします。

jmt load

jmt.conf ファイルにある Junction マッピング・テーブル・データをロードします。このファイルは、デフォルトでは存在しないため、ファイルを作成してデータを追加する必要があります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/` オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、WebSEAL に新規情報が反映されるように、jmt.conf ファイルから Junction マッピング・テーブル・データを読み込みます。

```
pdadmin> server task default-webseald-abc.ibm.com jmt load
```

出力は以下のようになります。

```
JMT table successfully loaded.
```

関連項目

738 ページの『server task reload』

server task list

server task list コマンドは、WebSEAL サーバーまたはインスタンス上のすべての Junction ポイントをリストします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **list**

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/per_junction_point* オブジェクトを統括する ACL 内で **I** (リスト) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および **16** 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを **10** 進数および **16** 進数のコードでリストします。

例

以下の例では、*default-webseald-cruz* WebSEAL サーバーのすべての Junction ポイントをリストします。

```
pdadmin> server task default-webseald-cruz list
```

出力は以下のようになります。

```
/
/ssljct
/tcpjct
```

関連項目

711 ページの『server task add』
717 ページの『server task create』
726 ページの『server task delete』
739 ページの『server task remove』
742 ページの『server task show』

server task offline

server task offline コマンドは、この Junction に配置されているサーバーをオフライン操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name offline [-i server_uuid] junction_point
```

説明

server task offline コマンドは、この Junction に配置されているサーバーをオフライン操作状態にします。指定されたサーバーに追加で送信される要求はありません。サーバーが指定されない場合は、この Junction のすべてのサーバーがオフライン操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

-i *server_uuid*

オフライン操作状態にするサーバーの UUID を指定します。サーバーが指

定されない場合は、この Junction のすべてのサーバーがオフライン操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task show** コマンドを使用します。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/junction_point` オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、`/pubs junction` ポイントにある `backapp1` サーバーがオフライン操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう
に、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Throttled
Throttled at: 2005-03-01-17:07:24
Hostname: backapp1.diamond.example.com
...
Current requests: 0
...
```

このサーバーをオフライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz offline
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

関連項目

『server task online』
747 ページの『server task throttle』
763 ページの『server task virtualhost offline』
765 ページの『server task virtualhost online』
772 ページの『server task virtualhost throttle』

server task online

server task online コマンドは、この Junction に配置されているサーバーをオンライン操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **online** [-i *server_uuid*] *junction_point*

説明

server task online コマンドは、この Junction に配置されているサーバーをオンライン操作状態にします。サーバーは通常操作を再開します。サーバーが指定されない場合は、この Junction のすべてのサーバーがオンライン操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

-i *server_uuid*

オンライン操作状態にするサーバーの UUID を指定します。サーバーが指定されない場合は、この Junction のすべてのサーバーがオンライン操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task show** コマンドを使用します。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/junction_point` オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、`/pubs` Junction ポイントにある `backapp1` サーバーがオンライン操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Offline
Hostname: backapp1.diamond.example.com
...
Current requests: 0
...
```

このサーバーをオンライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz online
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

関連項目

733 ページの『`server task offline`』

747 ページの『`server task throttle`』

763 ページの『server task virtualhost offline』

765 ページの『server task virtualhost online』

772 ページの『server task virtualhost throttle』

server task refresh all_sessions

server task refresh all_sessions コマンドは、指定したユーザーのすべてのセッションの資格情報をリフレッシュします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **refresh all_sessions** *user_id*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

user_id 指定したユーザーに関連するすべてのセッションの資格情報をリフレッシュします。ユーザー名の例として、`dlucas`、`sec_master` および「Mary Jones」などがあります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/` オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマン

ドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があります、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、**test_user** ユーザーのすべてのセッションがリフレッシュされます。

```
pdadmin> server task default-webseald-cruz refresh all_sessions test_user
```

関連項目

746 ページの『**server task terminate session**』

744 ページの『**server task terminate all_sessions**』

server task reload

server task reload コマンドは、データベースから **junction** マッピング・テーブルを再ロードします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task instance_name-webseald-host_name reload

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が **default** で、WebSEAL サーバー がインストールされているホストのマシン名が **abc.ibm.com** である場合、完全な WebSEAL サーバー名は **default-webseald-abc.ibm.com** になります。

追加の WebSEAL インスタンスが構成され、**web2** という名前が付けられると、完全な WebSEAL サーバー名は **web2-webseald-abc.ibm.com** になります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
`/WebSEAL/host_name-instance_name/` オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、Junction マッピング・テーブルをデータベースから再ロードします。

```
pdadmin> server task default-webseald-abc.ibm.com reload
```

関連項目

730 ページの『`server task jmt`』

server task remove

server task remove コマンドは、指定したインストール済みの WebSEAL サーバーまたはインスタンスを WebSEAL Junction ポイントから削除します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name remove -i server_uuid junction_point
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

-i *server_uuid*

Junction ポイントから除去するサーバーの UUID を指定します。UUID の取得についての詳細は、**server task show** コマンドを参照してください。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
/WebSEAL/host_name-instance_name/junction_point オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

- 0** コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1** コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、*/pubs* Junction ポイントから *backapp1* Junction サーバーを除去します。除去するサーバーの UUID を判別するには、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
...
Hostname: backappl.cruz.ibm.com
...
```

Junction からサーバーを除去します (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz remove
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

関連項目

711 ページの『server task add』
717 ページの『server task create』
726 ページの『server task delete』
742 ページの『server task show』

server task server restart

server task server restart コマンドは、Security Access Manager サーバー・タスク・フレームワークを使用して、WebSEALサーバーを再始動します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *server_name* **server restart**

オプション

server_name

再始動する WebSEAL サーバーの名前を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
/WebSEAL/*host_name-instance_name* オブジェクトを統括する ACL 内で **s** (管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提

供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

次の例では、server03 を再始動します。

```
pdadmin> server task server03 server restart
```

server task show

server task show コマンドは、指定した WebSEAL junction の詳細情報を表示します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **show** *junction_point*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/junction_point* オブジェクトを統括する ACL 内で **I** (リスト) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー abc.ibm.com 上のローカル・ルート Junction ポイント / についての情報を表示します。

```
pdadmin> server task default-webseald-abc.ibm.com show
```

出力は以下のようになります。

```
Junction point: /
Type: Local
Junction hard limit: 0 - using global value
Junction soft limit: 0 - using global value
Active worker threads: 0
Root Directory: /opt/pdweb/www-default/docs
...
Server 1:
ID: 78a1eb8c-074a-11d9-abda-00096bda9439
...
```

関連項目

- 711 ページの『server task add』
- 717 ページの『server task create』
- 726 ページの『server task delete』
- 739 ページの『server task remove』

server task server sync

server task server sync コマンドは、Security Access Manager サーバー・タスク・フレームワークを使用して、2 つの WebSEAL サーバー間で構成データを同期化します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

注: 2 つの WebSEAL サーバーのタイプは同じでなければなりません。WebSEAL サーバーのタイプは次のいずれかです。

- Web Gateway Appliance で稼働する WebSEAL。
- 標準オペレーティング・システムで稼働する WebSEAL。

構文

server task webseal_server server sync server_name

オプション

webseal_server

インストール済み WebSEAL インスタンスの完全修飾サーバー名を指定します。*webseal_server* はターゲットです。

server_name

データを抽出する WebSEAL サーバーの名前を指定します。ホスト・システム上の構成データがバックアップされ、その後このデータと同期化されます。*server_name* はソースです。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name* オブジェクトを統括する ACL 内で **s** (管理) 許可が付与されている必要があります。 **sec_master** 管理ユーザーには、デフォルトで許可が付与されます。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

次の例では、構成データをサーバー *master-webseald-abc.ibm.com* と同期します。

```
pdadmin> server task default-webseald-abc.ibm.com server sync
master-webseald-abc.ibm.com
```

server task terminate all_sessions

server task terminate all_sessions コマンドは、特定のユーザーのすべてのユーザー・セッションを終了します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **terminate all_sessions** *user_id*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

user_id ユーザーの名前を指定します。ユーザー名の例として、*dlucas*、*sec_master* および「*Mary Jones*」などがあります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/* オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、default-webseald-cruz WebSEAL サーバー上の dlucas ユーザーのすべてのセッションが終了されます。

```
pdadmin> server task default-webseald-cruz terminate all_sessions dlucas
```

関連項目

『server task terminate session』

737 ページの『server task refresh all_sessions』

server task terminate session

server task terminate session コマンドは、セッション ID を使用してユーザー・セッションを終了します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **terminate session** *session_id*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が default で、WebSEAL サーバー がインストールされているホストのマシン名が abc.ibm.com である場合、完全な WebSEAL サーバー名は default-webseald-abc.ibm.com になります。

追加の WebSEAL インスタンスが構成され、web2 という名前が付けられると、完全な WebSEAL サーバー名は web2-webseald-abc.ibm.com になります。

session_id

ユーザー・セッションの ID を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/* オブジェクトを統括する ACL 内で **s** (サーバー管理) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例 (1 行で入力します) では、default-webseald-cruz WebSEAL サーバー上の特定のセッションが終了されます。

```
pdadmin> server task default-webseald-cruz terminate  
session 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
```

関連項目

737 ページの『server task refresh all_sessions』

744 ページの『server task terminate all_sessions』

server task throttle

server task throttle コマンドは、この Junction に配置されているサーバーをスロットル操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name throttle [-i server_uuid]  
junction_point
```

説明

server task throttle コマンドは、この Junction に配置されているサーバーをスロットル操作状態にします。指定されたサーバーが継続して処理するのは、このコマンドを呼び出す前に WebSEAL とのセッションを作成したユーザーからの要求のみです。サーバーが指定されない場合は、この Junction のすべてのサーバーがスロットル操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定しま

す。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

-i server_uuid

スロットルするサーバーの UUID を指定します。サーバーが指定されない場合は、この Junction のすべてのサーバーがスロットル操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task show** コマンドを使用します。

junction_point

バックエンド・サーバーの文書スペースがマウントされる WebSEAL の保護オブジェクト・スペース内のディレクトリー名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
/WebSEAL/host_name-instance_name/junction_point オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、/pubs Junction ポイントにある backappl サーバーがスロットル操作状態になります。この Junction サーバーの UUID を判別するには、以下のよう
に、**server task show** コマンドを実行します。

```
pdadmin> server task default-webseald-cruz show /pubs
```

出力は以下のようになります。

```
Junction point: /pubs
...
Server 1:
ID: 6fc3187a-ea1c-11d7-8f4e-09267e38aa77
Server State: running
Operational State: Online
Hostname: backappl.diamond.example.com
...
Current requests: 0
...
```

このサーバーをスロットル操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz throttle
-i 6fc3187a-ea1c-11d7-8f4e-09267e38aa77 /pubs
```

関連項目

733 ページの『server task offline』
735 ページの『server task online』
763 ページの『server task virtualhost offline』
765 ページの『server task virtualhost online』
772 ページの『server task virtualhost throttle』

server task virtualhost add

server task virtualhost add コマンドは、追加のインストール済み WebSEAL サーバーまたはインスタンスを既存の仮想ホスト Junction に追加します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name virtualhost add -h host_name
[options] vhost_label
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が default で、WebSEAL サーバー がインストールされているホストのマシン名が abc.ibm.com である場合、完全な WebSEAL サーバー名は default-webseald-abc.ibm.com になります。

追加の WebSEAL インスタンスが構成され、web2 という名前が付けられると、完全な WebSEAL サーバー名は web2-webseald-abc.ibm.com になります。

options **server task virtualhost add** コマンドに使用できるオプションを指定します。これらのオプションには、次のようなものがあります。

-D "dn"

バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張し、SSL 上での相互認証を提供するものです。例えば、www.example.com の証明書の DN は次のようになります。

```
"CN=WWW.EXAMPLE.COM,OU=Software,O=example.com,Inc,L=Austin,ST=Texas,C=US"
```

このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-H host_name

プロキシ・サーバーの DNS ホスト名または IP アドレスを指定します。

host_name の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

```
proxy.www.example.com
```

このオプションは、tcpproxy タイプまたは sslproxy タイプとして作成された Junction で使用されます。

-i WebSEAL サーバーが URL の大/小文字を区別しないことを表します。

このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

-p port

バックエンド・サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

-P port

プロキシ・サーバーの TCP ポートを指定します。デフォルト値は 7138 です。

port の場合は、任意の有効なポート番号を使用します。有効なポート番号とは、TCP/IP で使用でき、他のアプリケーションで現在使用されていない任意の正数です。デフォルトのポート番号値を使用するか、現在使用されていない、1000 より大きいポート番号を使用します。

このオプションは、tcpproxy タイプまたは sslproxy タイプとして作成された Junction で使用されます。

-q path

バックエンド Windows 仮想ホストの場合は必須オプションです。

query_contents スクリプトの相対パスを指定します。デフォルトでは、Security Access Manager は /cgi_bin サブディレクトリーでこのスクリプトを探します。これとは別のディレクトリーに入っている場合、または **query_contents** ファイルの名前が変更されている場合は、このオプションを使用して、該当のファイルへの新しい URL を WebSEAL に指示してください。

このオプションは、localtcp タイプおよび localssl タイプを除くすべての Junction タイプに対して有効です。

-u uuid

-s オプションを使用するステートフル Junction を介してこのバックエンド・サーバーが WebSEAL に接続されたときの、UUID を指定します。このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

-w Microsoft Windows ファイル・システム・サポートを示します。

このオプションは、tcp タイプまたは ssl タイプとして作成された Junction で使用されます。

vhost_label

仮想ホスト Junction のラベル名を指定します。

-h host_name

必須オプション。ターゲット・サーバーの DNS ホスト名または IP アドレスを指定します。 *host_name* の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

www.example.com

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、/WebSEAL/*host_name-instance_name*/@*vhost_label* オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。「IBM Security Access Manager for Web: Error Message

Reference」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例 (1 行で入力します) では、WebSEAL サーバー `abc.ibm.com` 上にある `support-vhost-http` というラベルの既存の仮想ホスト Junction に対して、`xyz.ibm.com` というホスト名で追加のサーバーを追加します。

```
pdadmin> server task default-webseald-abc.ibm.com virtualhost add
-h xyz.ibm.com support-vhost-http
```

関連項目

『server task virtualhost create』

760 ページの『server task virtualhost delete』

761 ページの『server task virtualhost list』

767 ページの『server task virtualhost remove』

770 ページの『server task virtualhost show』

server task virtualhost create

server task virtualhost create コマンドは仮想ホスト Junction を作成します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

ローカル Junction の場合

```
server task instance_name-webseald-host_name virtualhost create -t type -v  
virtual_host_name [options] vhost_label
```

ローカルでない Junction の場合

```
server task instance_name-webseald-host_name virtualhost create -t type  
-h host_name [options] vhost_label
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。`webseald` の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、web2 という名前が付けられると、完全な WebSEAL サーバー名は web2-webseald-abc.ibm.com になります。

options **server task virtualhost create** コマンドに使用できるオプションを指定します。これらのオプションには、次のようなものがあります。

-A 仮想ホスト Junction が、LTPA (Lightweight Third-Party Authentication) メカニズムをサポートできるようにします。このオプションには **-F** オプションと **-Z** オプションが必要です。 **-A**、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。

このオプションは、localtcp タイプおよび localssl タイプを除くすべての Junction タイプに対して有効です。

-2 このオプションは、**-A** オプションとともに使用でき、LTPA バージョン 2 Cookie (LtpaToken2) が使用されることを指定します。 **-2** オプションを指定しない **-A** オプションは、LTPA バージョン 1 Cookie (LtpaToken) が使用されることを指定します。

-b BA_value

WebSEAL サーバーがバックエンド仮想ホストに HTTP 基本認証 (BA) ヘッダー内のクライアント識別情報を渡す方法を定義します。以下の値のいずれか 1 つになります。

- filter
- ignore
- supply
- gso

このオプションは、localtcp タイプおよび localssl タイプを除くすべての Junction タイプに対して有効です。

デフォルト値は filter です。

-B WebSEAL が基本認証ヘッダー情報を使用してバックエンド仮想ホストに対して認証し、SSL を介した相互認証を提供することを示します。このオプションには **-U** オプションと **-W** オプションが必要です。

このオプションは、ssl タイプまたは sslproxy タイプとして作成された Junction でのみ有効です。

-c header_type

仮想ホスト Junction を介して HTTP ヘッダーに Security Access Manager のクライアント識別を挿入します。header_type 引数には、以下の Security Access Manager HTTP ヘッダー・タイプを任意に組み合わせて指定できます。

- {iv-user|iv-user-1}
- iv-groups
- iv-creds
- all

ヘッダー・タイプはコンマで区切る必要があり、タイプ間にスペースを入れることはできません。例: -c iv_user,iv_groups

-c all を指定することは、**-c iv-user,iv-groups,iv-creds** を指定することと同じです。

このオプションは、**localtcp** タイプおよび **localssl** タイプを除くすべての **Junction** タイプに対して有効です。

- C** フロントエンド WebSEAL サーバーが、その識別情報を基本認証 (BA) ヘッダーに入れて、バックエンド WebSEAL サーバーに渡すことにより、相互認証をサポートします。さらに、**-C** オプションを使用すると、**-c** オプションによって提供されるシングル・サインオン機能性を使用可能にできます。

このオプションは、**ssl** タイプまたは **sslproxy** タイプとして作成された **Junction** でのみ有効です。

-D "dn"

バックエンド・サーバー証明書の識別名を指定します。この値は、実際の証明書 DN と突き合わされて、認証を拡張し、SSL 上での相互認証を提供するものです。例えば、**www.example.com** の証明書の DN は次のようになります。

```
"CN=WWW.EXAMPLE.COM,OU=Software,O=example.com,Inc,L=Austin,ST=Texas,C=US"
```

このオプションは、**ssl** タイプまたは **sslproxy** タイプとして作成された **Junction** でのみ有効です。

-e encoding_type

仮想ホスト **Junction** に対して HTTP ヘッダーを生成するときに使用するエンコード方式を指定します。このエンコード方式は、**-c Junction** オプションと **tag-value** の両方を指定して生成されたヘッダーに適用されます。エンコードに可能な値は以下のとおりです。

utf8_bin

WebSEAL は UTF-8 でヘッダーを送信します。

utf8_uri

WebSEAL は UTF-8 でヘッダーを送信しますが、URI でもエンコードが行われます。この動作がデフォルトの動作です。

lcp_bin

WebSEAL は、WebSEAL サーバーのローカル・コード・ページのヘッダーを送信します。

lcp_uri

WebSEAL は、WebSEAL サーバーのローカル・コード・ページのヘッダーを送信しますが、URI でもエンコードが行われます。

このオプションは、**localtcp** タイプおよび **localssl** タイプを除くすべての **Junction** タイプに対して有効です。

- f** 既存の仮想ホスト **Junction** の強制的な置換 (上書き) を実行します。

このオプションは、いずれの Junction タイプで作成された Junction でも使用されます。

-F "keyfile"

LTPA Cookie データの暗号化に使用される鍵ファイルの名前を指定します。

-F オプションには **-A** オプションと **-Z** オプションが必要です。**-A**、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-g vhost_label

-g オプションを指定すると、2 番目の追加仮想ホスト Junction が最初の仮想ホスト Junction と同じ保護オブジェクト・スペースを共有します。

このオプションは、Junction ペア (補完プロトコルを使用する 2 つの Junction) にのみ該当します。このオプションでは、2 つを超える Junction の関連付けはサポートしていません。

-H host_name

プロキシ・サーバーの DNS ホスト名または IP アドレスを指定します。**-P** オプションもプロキシ・サーバー Junction をサポートします。`host_name` の有効値には、すべての有効な IP ホスト名が含まれます。例えば、次のとおりです。

```
proxy.www.example.com
```

このオプションは、`tcpproxy` タイプまたは `sslproxy` タイプとして作成された Junction でのみ有効です。

-i

WebSEAL Junction が URL の大/小文字を区別しないことを表します。大/小文字の区別をしない Junction への要求を正しく認可するため、WebSEAL では、URL の小文字バージョンに対して許可検査を実行します。例えば、Windows オペレーティング・システム上で実行されている Web サーバーは、`INDEX.HTM` および `index.htm` への要求を同一ファイルへの要求として扱います。

このような Web サーバーへの Junction は、**-i** オプションまたは **-w** オプションを指定して作成する必要があります。Junction ポイント下のオブジェクトに付加する ACL または POP は、小文字のオブジェクト名を使用する必要があります。`/junction/index.htm` に付加された ACL は、**-i** オプションまたは **-w** オプションが使用されている場合は、以下の要求すべてに適用されます。

```
/junction/INDEX.HTM  
/junction/index.htm  
/junction/InDeX.HtM
```

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction に対して有効です。ローカル junction は Win32 プラットフォームでのみ大文字小文字の区別がありません。他のすべてのプラットフォームでは、大文字小文字の区別があります。

-k バックエンド仮想ホストに WebSEAL セッションの Cookie を送信します。デフォルトでは、Cookie はサーバーに送信される要求から除去されます。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-K "key_label"

WebSEAL がバックエンド・サーバーに提示する必要があるクライアント・サイドの証明書の鍵ラベルを指定します。このオプションを使用すると、仮想ホストがクライアント証明書を使用して WebSEAL サーバーを認証できます。

このオプションは、`ssl` タイプおよび `sslproxy` タイプとして作成された Junction でのみ有効です。

-l percent

ワーカー・スレッドの消費量のソフト限界を定義します。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-L percent

ワーカー・スレッドの消費量のハード限界を定義します。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-p port

バックエンドのサード・パーティー・サーバーの TCP ポートを指定します。デフォルト値は、TCP Junction の場合は 80、SSL Junction の場合は 443 です。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-P port

HTTP プロキシ・サーバーの TCP ポート番号を指定します。**-H** オプションが使用される場合は、**-P** オプションが必須です。

このオプションは、`tcpproxy` タイプまたは `sslproxy` タイプとして作成された Junction でのみ有効です。

-q path

バックエンド Windows 仮想ホストの場合は必須オプションです。`query_contents` スクリプトの相対パスを指定します。デフォルトでは、Security Access Manager は `/cgi_bin` ディレクトリーで `query_contents` スクリプトを探します。これとは別のディレクトリーに入っている場合、または `query_contents` のファイルの名前が変更されている場合は、該当のファイルへの新しい URL が、このオプションにより WebSEAL に示されます。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

-r Junction を介して HTTP ヘッダーに着信 IP アドレスを挿入します。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- R** 要求の進行は許可しますが、HTTP ヘッダー内の `Junction` にルール違反の理由を通知します。**-R** オプションが使用されずにルール違反が生じると、WebSEAL では、要求を進行できません。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- s** 仮想ホスト `junction` がステートフル・アプリケーションをサポートすることを表します。デフォルトでは、仮想ホスト `Junction` はステートフルではありません。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- S** フォーム・シングル・サインオン構成ファイルの名前を示します。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- T {resource | resource_group}**

GSO リソースまたはリソース・グループの名前を指定します。このオプションは **-b gso** オプションが使用される場合にのみ必須です。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- u uuid**

ステートフル仮想ホスト `Junction` (**-s** オプション) を使用して WebSEAL に接続されている、バックエンド・サーバーの UUID (Universally Unique Identifier) を指定します。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての `Junction` タイプに対して有効です。

- U "user_name"**

WebSEAL サーバー・ユーザー名を指定します。このオプションには **-B** オプションと **-W** オプションが必要です。WebSEAL は基本認証ヘッダー情報を使用してバックエンド仮想ホストに対して認証し、SSL を介した相互認証を提供します。

このオプションは、`ssl` タイプまたは `sslproxy` タイプとして作成された `Junction` でのみ有効です。

- v vhost_name[:port]**

要求の HTTP **Host** ヘッダーが、**-v** オプションで指定されている仮想ホスト名およびポート番号に一致する場合、WebSEAL は要求を処理するために仮想ホスト `Junction` を選択します。

-v オプションは、バックエンド・サーバーに送信される要求の **Host** ヘッダーの値を指定するためにも使用されます。

仮想ホストがプロトコルの標準外のポートを使用する場合、ポート番号が必要です。TCP の標準ポートは 80、SSL の標準ポートは 443 です。

tcp、**ssl**、**tcpproxy**、および **sslproxy** タイプの **junction** に **-v** が指定されていない場合は、**-h host** および **-p port** オプションに含まれている情報から **junction** が選択されます。

localtcp および **localssl** タイプの **junction** では、**-v** オプションは必須です。

- w** Microsoft Windows ファイル・システム・サポートを示します。このオプションでは、**-i Junction** オプションにより提供される機能がすべて提供されますが、Windows のファイル名のエイリアスとして解釈される可能性のあるファイル名を含む要求は認可されません。

このオプションは、**localtcp** タイプおよび **localssl** タイプを除くすべての **Junction** タイプに対して有効です。ローカルの **Junction** では、Win64 上で Windows のファイル名の別名を含む URL は禁止されていますが、他のプラットフォームではそのような URL が許可されています。

- W "password"**

WebSEAL サーバー・パスワードを指定します。このオプションには **-B** オプションと **-U** オプションが必要です。WebSEAL は基本認証ヘッダー情報を使用してバックエンド仮想ホストに対して認証し、SSL を介した相互認証を提供します。

このオプションは、**ssl** タイプまたは **sslproxy** タイプとして作成された **Junction** でのみ有効です。

- Y** **junction** の Tivoli Federated Identity Manager シングル・サインオン (SSO) を使用可能にします。

注: このオプションを使用する前に、まず **Junction** 全体で Tivoli Federated Identity Manager シングル・サインオンをサポートするように WebSEAL 構成ファイルを構成する必要があります。

- z replica_set**

次のようにして、レプリカ・セットを指定します。

SMS 環境の場合

仮想ホスト **Junction** とセッションを持つレプリカ・セットは、ログイン・セッションの下で管理され、それらのセッションを複数の仮想ホスト間でグループ化したり、分離したりする機能を提供します。

非 SMS 環境の場合

仮想ホスト **Junction** とセッションを持つレプリカ・セットは、WebSEAL セッション・キャッシュの下で管理され、WebSEAL セッション・キャッシュの区分化を制御します。この制御は、**[session]** スタンザの **standard-junction-replica-set** エントリーを使用してレプリカ・セットに割り当てられた任意の標準的な **Junction** と同一のレプリカ・セットに、仮想ホストを含められるように行われます。

-Z *keyfile_pwd*

LTPA Cookie データの暗号化に使用される鍵格納ファイルのパスワードを指定します。このオプションには **-A** オプションと **-F** オプションが必要です。 **-A**、**-F**、および **-Z** オプションはすべて一緒に使用する必要があります。

このオプションは、`localtcp` タイプおよび `localssl` タイプを除くすべての Junction タイプに対して有効です。

vhost_label

仮想ホスト Junction のラベル名を指定します。

-h *host_name*

ローカルでない Junction の場合は必須オプションです。ターゲット・サーバーの DNS ホスト名または IP アドレスを指定します。このオプションは、ローカルでない Junction に対してのみ有効です。ローカルの Junction にはホスト名は必要ありません。 *host_name* の有効値には、すべての有効な IP ホスト名が含まれます。以下に例を示します。

`www.example.com`

-t *type* 必須オプション。仮想ホスト Junction のタイプを指定します。このオプションは必須で、以下のタイプのいずれか 1 つになります。

- `tcp`
- `tcpproxy`
- `ssl`
- `sslproxy`
- `localtcp`
- `localssl`

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/@vhost_label` オブジェクトを統括する ACL 内で `s` (サーバー管理) 許可が付与されていることが必要です。例えば、`sec_master` 管理ユーザーには、デフォルトでこの許可が付与されます。

注: 統計の収集について詳しくは、「*IBM Security Access Manager for Web: Troubleshooting Guide*」を参照してください。

戻りコード

0 コマンドが正常に完了しました。WebSEAL の `server task` コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは `0` になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例 (1 行で入力します) では、vhost-xy-https ラベルを持つ SSL タイプの仮想ホスト Junction が作成されます。この Junction は、Junction サーバー cruz1.ibm.com にある仮想ホスト x.y.com の処理を実行します。WebSEAL は、この仮想ホスト Junction を介して要求を転送することにより、SSL 要求内の Host: x.y.com ヘッダー (HTTPS) に応答します。

```
pdadmin> server task default-webseald-abc.ibm.com virtualhost create
-t ssl -h cruz1.ibm.com -v x.y.com vhost-xy-https
```

関連項目

749 ページの『server task virtualhost add』
『server task virtualhost delete』
761 ページの『server task virtualhost list』
767 ページの『server task virtualhost remove』
770 ページの『server task virtualhost show』

server task virtualhost delete

server task virtualhost delete コマンドは、仮想ホスト Junction を削除します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **virtualhost delete** *vhost_label*

説明

server list virtualhost delete コマンドは、仮想ホスト Junction を削除します。仮想ホスト junction を、別の仮想ホスト junction が **-g** オプションを使用して参照している場合、参照されている仮想ホスト junction は削除できません。このような操作を試行すると、エラー・メッセージが戻されます。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、web2 という名前が付けられると、完全な WebSEAL サーバー名は web2-webseald-abc.ibm.com になります。

vhost_label

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/@vhost_label` オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例 (1 行で入力します) では、WebSEAL サーバー abc.ibm.com から仮想ホスト Junction support-vhost-https を削除します。

```
pdadmin> server task default-webseald-abc.ibm.com virtualhost delete  
support-vhost-https
```

関連項目

749 ページの『server task virtualhost add』
752 ページの『server task virtualhost create』
『server task virtualhost list』
767 ページの『server task virtualhost remove』
770 ページの『server task virtualhost show』

server task virtualhost list

server task virtualhost list コマンドは、すべての構成済み仮想ホスト Junction をラベル名でリストします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **virtualhost list**

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/@per_vhost_label* オブジェクトを統括する ACL 内で **I** (リスト) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および **16** 進形式 (例えば、**0x14c012f2**) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを **10** 進数および **16** 進数のコードでリストします。

例

以下の例では、*abc.ibm.com* WebSEAL サーバー上に構成されているすべての仮想ホスト **Junction** のラベル名をリストします。

```
pdadmin> server task default-webseald-abc.ibm.com virtualhost list
```

出力は以下のようになります。

```
pubs-vhost-http
sales-vhost-https
support-vhost-http
```

関連項目

749 ページの『`server task virtualhost add`』
752 ページの『`server task virtualhost create`』
760 ページの『`server task virtualhost delete`』
767 ページの『`server task virtualhost remove`』
770 ページの『`server task virtualhost show`』

server task virtualhost offline

server task virtualhost offline コマンドは、この仮想ホスト Junction に配置されているサーバーをオフライン操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name virtualhost offline [-i server_uid]  
vhost_label
```

説明

server task virtualhost offline コマンドは、この仮想ホスト Junction に配置されているサーバーをオフライン操作状態にします。指定されたサーバーに追加で送信される要求はありません。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがオフライン操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

`-i server_uuid`

オフライン操作状態にするサーバーの UUID を指定します。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがオフライン操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task virtualhost show** コマンドを使用します。

`vhost_label`

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/@vhost_label` オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および **16** 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを **10** 進数および **16** 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー `abc.ibm.com` 上に構成されている `support-vhost-https` というラベルの仮想ホスト Junction により、バックエンド Junction サーバー `int3.ibm.com` にある仮想ホスト `support.ibm.com` がサポートされます。

`int3.ibm.com` サーバーをオフライン操作状態にする要求があります。この Junction サーバーの UUID を判別するには、**server task virtualhost show** コマンド (1 行で入力します) を実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
```

```
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Throttled
Throttled at: 2005-03-01-17:07:24
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1
```

次のコマンドを使用して、このサーバーをオフライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost offline
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

関連項目

733 ページの『server task offline』
735 ページの『server task online』
747 ページの『server task throttle』
『server task virtualhost online』
772 ページの『server task virtualhost throttle』

server task virtualhost online

server task virtualhost online コマンドは、この仮想ホスト Junction に配置されているサーバーをオンライン操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name virtualhost online [-i server_uuid]
vhost_label
```

説明

server task virtualhost online コマンドは、この仮想ホスト Junction に配置されているサーバーをオンライン操作状態にします。サーバーは通常操作を再開します。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがオンライン操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

-i server_uuid

オンライン操作状態にするサーバーの UUID。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがオンライン操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task virtualhost show** コマンドを使用します。

vhost_label

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、*/WebSEAL/host_name-instance_name/@vhost_label* オブジェクトを統括する ACL 内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および **16** 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを **10** 進数および **16** 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー *abc.ibm.com* 上に構成されている *support-vhost-https* というラベルの仮想ホスト Junction により、バックエンド Junction サーバー *int3.ibm.com* にある仮想ホスト *support.ibm.com* がサポートされます。

int3.ibm.com サーバーをオンライン操作状態にする要求があります。この Junction サーバーの UUID を判別するには、**server task virtualhost show** コマンド (1 行で入力します) を実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Offline
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1
```

次のコマンドを使用して、このサーバーをオンライン操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost online
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

関連項目

733 ページの『server task offline』
735 ページの『server task online』
747 ページの『server task throttle』
763 ページの『server task virtualhost offline』
772 ページの『server task virtualhost throttle』

server task virtualhost remove

server task virtualhost remove コマンドは、仮想ホスト Junction から指定したサーバーを除去します。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name virtualhost remove -i server_uuid
vhost_label
```

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定しま

す。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。
webseald の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が *default* で、WebSEAL サーバー がインストールされているホストのマシン名が *abc.ibm.com* である場合、完全な WebSEAL サーバー名は *default-webseald-abc.ibm.com* になります。

追加の WebSEAL インスタンスが構成され、*web2* という名前が付けられると、完全な WebSEAL サーバー名は *web2-webseald-abc.ibm.com* になります。

-i *server_uuid*

仮想ホスト Junction から除去するサーバーの UUID を指定します。このコマンドでは、通常 URL で大/小文字の区別をするために使用される **-i** オプションは、**-u** オプションと同様に作動します。UUID の取得についての詳細は、**server task show** コマンドを参照してください。

vhost_label

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、
/WebSEAL/host_name-instance_name/@vhost_label オブジェクトを統括する ACL
内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

- 0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは 0 になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

- 1 コマンドが失敗しました。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、仮想ホスト Junction *support-vhost-https* から Junction サーバー *int4.ibm.com* を除去します。除去するサーバーの UUID を判別するには、**server task virtualhost show** コマンド (1 行で入力します) を実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下ようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
Junction hard limit: 0 - using global value
Junction soft limit: 0 - using global value
Active worker threads: 0
Basic authentication mode: filter
Forms based SSO: disabled
Authentication HTTP header: do not insert
Remote Address HTTP header: do not insert
Stateful junction: no
Boolean Rule Header: no
Delegation support: no
Mutually authenticated: no
Insert WebSphere LTPA cookies: no
Insert WebSEAL session cookies: no
Request Encoding: UTF-8, URI Encoded
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Total requests: 1
Server 2:
ID: xycecc77-19ve-81y5-4h0a-90267hj5nn57
Server State: running
Hostname: int4.ibm.com
Port: 444
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Total requests: 1
```

仮想ホスト Junction からサーバーを除去します (1 行で入力します)。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost remove -i xycecc77-19ve-81y5-4h0a-90267hj5nn57 support-vhost-https
```

関連項目

749 ページの『server task virtualhost add』
752 ページの『server task virtualhost create』
760 ページの『server task virtualhost delete』
761 ページの『server task virtualhost list』
770 ページの『server task virtualhost show』

server task virtualhost show

server task virtualhost show コマンドは、指定した仮想ホスト Junction に関する情報を表示します。仮想ホスト Junction は存在するものでなければなりません。そうでない場合はエラーが表示されます。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

server task *instance_name-webseald-host_name* **virtualhost show** *vhost_label*

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。**webseald** の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が **default** で、WebSEAL サーバー がインストールされているホストのマシン名が **abc.ibm.com** である場合、完全な WebSEAL サーバー名は **default-webseald-abc.ibm.com** になります。

追加の WebSEAL インスタンスが構成され、**web2** という名前が付けられると、完全な WebSEAL サーバー名は **web2-webseald-abc.ibm.com** になります。

vhost_label

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、**/WebSEAL/host_name-instance_name/@vhost_label** オブジェクトを統括する ACL 内で **1** (リスト) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。

注: コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および **16** 進形式 (例えば、**0x14c012f2**) のエラー状況コードを提

供します。「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー abc.ibm.com 上に構成され、バックエンド Junction サーバー int3.ibm.com にある仮想ホスト support.ibm.com をサポートする support-vhost-https というラベルの仮想ホスト Junction の情報を表示します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
Junction hard limit: 0 - using global value
Junction soft limit: 0 - using global value
Active worker threads: 0
Basic authentication mode: filter
Forms based SSO: disabled
Authentication HTTP header: do not insert
Remote Address HTTP header: do not insert
Stateful junction: no
Boolean Rule Header: no
Delegation support: no
Mutually authenticated: no
Insert WebSphere LTPA cookies: no
Insert WebSEAL session cookies: no
Request Encoding: UTF-8, URI Encoded
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Total requests: 1
```

関連項目

749 ページの『server task virtualhost add』
752 ページの『server task virtualhost create』
760 ページの『server task virtualhost delete』
761 ページの『server task virtualhost list』
767 ページの『server task virtualhost remove』

server task virtualhost throttle

server task virtualhost throttle コマンドは、この仮想ホスト Junction に配置されているサーバーをスロットル操作状態にします。

このコマンドの使用には、認証 (管理者 ID およびパスワード) が必要です。

構文

```
server task instance_name-webseald-host_name virtualhost throttle [-i server_uuid]  
vhost_label
```

説明

server task virtualhost throttle コマンドは、この仮想ホスト Junction に配置されているサーバーをスロットル操作状態にします。指定されたサーバーが継続して処理するのは、このコマンドを呼び出す前に WebSEAL とのセッションを作成したユーザーからの要求のみです。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがスロットル操作状態になります。

オプション

instance_name-webseald-host_name

インストール済み WebSEAL インスタンスの完全なサーバー名を指定します。**server list** コマンドの出力で表示されるのと厳密に同じ形式で完全なサーバー名を指定する必要があります。

instance_name には、WebSEAL インスタンスの、構成名を指定します。*webseald* の指定は、その WebSEAL サービスがコマンド・タスクを実行することを表します。*host_name* は、WebSEAL サーバーがインストール済みの物理的なマシン名です。

例えば、単一 WebSEAL インスタンスの構成名が `default` で、WebSEAL サーバー がインストールされているホストのマシン名が `abc.ibm.com` である場合、完全な WebSEAL サーバー名は `default-webseald-abc.ibm.com` になります。

追加の WebSEAL インスタンスが構成され、`web2` という名前が付けられると、完全な WebSEAL サーバー名は `web2-webseald-abc.ibm.com` になります。

-i *server_uuid*

スロットルするサーバーの UUID を指定します。サーバーが指定されない場合は、この仮想ホスト Junction のすべてのサーバーがスロットル操作状態になります。特定のバックエンド・サーバーの ID を判別するには、**server task virtualhost show** コマンドを使用します。

vhost_label

仮想ホスト Junction のラベル名を指定します。

許可

このコマンドにアクセスする必要があるユーザーおよびグループには、`/WebSEAL/host_name-instance_name/@vhost_label` オブジェクトを統括する ACL

内で **c** (制御) 許可が付与されていることが必要です。例えば、**sec_master** 管理ユーザーには、デフォルトでこの許可が付与されます。

戻りコード

0 コマンドが正常に完了しました。

注: WebSEAL の **server task** コマンドでは、コマンドが WebSEAL サーバーにエラーなしに送信された場合の戻りコードは **0** になります。コマンドが正常に送信された場合でも、WebSEAL サーバーが正常にコマンドを実行できない可能性があり、エラー・メッセージを返すことがあります。

1 コマンドが失敗しました。コマンドが失敗すると、**pdadmin** コマンドがエラーの説明および 16 進形式 (例えば、0x14c012f2) のエラー状況コードを提供します。

「*IBM Security Access Manager for Web: Error Message Reference*」を参照してください。このリファレンスでは、Security Access Manager エラー・メッセージを 10 進数および 16 進数のコードでリストします。

例

以下の例では、WebSEAL サーバー **abc.ibm.com** 上に構成されている **support-vhost-https** というラベルの仮想ホスト **Junction** により、バックエンド **Junction** サーバー **int3.ibm.com** にある仮想ホスト **support.ibm.com** がサポートされます。

int3.ibm.com サーバーをスロットル操作状態にする要求があります。この **Junction** サーバーの **UUID** を判別するには、**server task virtualhost show** コマンド (1 行で入力します) を実行します。

```
pdadmin> server task default-webseald-abc.ibm.com
virtualhost show support-vhost-https
```

出力は以下のようになります。

```
Virtual Host label: support-vhost-https
Type: SSL
...
Virtual hostname: support.ibm.com
Alias: ibm.com
Alias: support
Virtual Host junction protocol partner: support-vhost-http
Server 1:
ID: bacecc66-13ce-11d8-8f0a-09267ea5aa77
Server State: running
Operational State: Online
Hostname: int3.ibm.com
Port: 443
Server DN:
Query_contents URL: /cgi-bin/query_contents
Query-contents: unknown
Case insensitive URLs: no
Allow Windows-style URLs: yes
Current requests: 0
Total requests: 1
```

次のコマンドを使用して、このサーバーをスロットル操作状態にします (1 行で入力します)。

```
pdadmin> server task default-webseald-cruz virtualhost throttle  
-i bacecc66-13ce-11d8-8f0a-09267ea5aa77 support-vhost-https
```

関連項目

747 ページの『server task throttle』

733 ページの『server task offline』

735 ページの『server task online』

763 ページの『server task virtualhost offline』

765 ページの『server task virtualhost online』

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものであり、本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。

国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのもと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォーム

フォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。お客様は、IBM のアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。© Copyright IBM Corp. _年を入れる_. All rights reserved.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、<http://www.ibm.com/legal/copytrade.shtml> をご覧ください。

Adobe、PostScript は、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

IT Infrastructure Library は英国 Office of Government Commerce の一部である the Central Computer and Telecommunications Agency の登録商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

ITIL は英国 The Minister for the Cabinet Office の登録商標および共同体登録商標であって、米国特許商標庁にて登録されています。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Cell Broadband Engine は、Sony Computer Entertainment, Inc. の米国およびその他の国における商標であり、同社の許諾を受けて使用しています。



Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

【ア行】

アカウント管理ページ
構成 94
アカウント期限切れ 94
アカウント期限切れエラー・メッセージの構成 94
アクセシビリティ xxiii
アクセス制御 695
アクセス・コントロール・リスト (ACL) 10
圧縮
データ圧縮ポリシーの構成 65
ユーザー・エージェント・タイプに基づく 63
HTTP データ 62
MIME タイプに基づく 62
POP ポリシー 64
アプライアンス 5
アプリケーション 693
アプリケーション・サポート
要件 610
アプリケーション・サポート、バックエンド 671
暗号
リスト、サポートされる 396
イベント相関
バックエンド・サーバー用 682
イベント・プール
http 26
http.agent 26
http.clf 26
http.cof 26
http.ref 26
イベント・ログ
HTTP 26
logcfg 26
インスタンス、WebSEAL 663
構成ファイルへの 35
定義 663
名前 663
名前に使用できる文字 663
プライマリー・インターフェースの IP アドレス 664
保護オブジェクト・スペースでの 36

インスタンス、WebSEAL (続き)
ホスト名 663
listen ポート 664
pdadmin サーバー・リストでの 36
インターネット・プロトコル (IP)
バージョン 4 のサポート 54
バージョン 6 のサポート 54
インターフェース 693
仮想ホスト junction 558
プロパティ 558
エラー・ページ
スロットルおよびオフライン junction サーバー 462
エラー・メッセージ・ページ
構成 96
エラー・メッセージ・ログ 26
ルーティング・ファイル 26
Tivoli Common Directory 26
Tivoli メッセージ・フォーマット 26
UTF-8 形式のメッセージ 26
WebSEAL 26
エンコードされた URL 481
エンコード方式、要求内のデータの検証 76
応答
サーバー、HTML ページ 79
応答コード 27
応答ヘッダー
WebSEAL コンテンツ・キャッシュの影響 42
オブジェクト・スペースへのマッピング
動的 URL 694

【カ行】

外部許可サービス 139
外部許可サービス (EAS)
ランタイム・セキュリティ・サービス
外部許可サービス (EAS) 139, 143
リスクベース・アクセス 139
rtss-eas スタンザ 143
外部認証インターフェース 213, 263
外部認証インターフェース指定リダイレクト 277
概要 263
カスタム・アプリケーションの作成 273
構成 267
再認証 278
資格情報交換 271

外部認証インターフェース (続き)
資格情報の生成 271
使用可能化および使用不可化 267
セッション処理 277
デモンストレーション・プログラム 273
トリガー URL 268
認証強度レベル 277
認証後リダイレクト 277
認証処理の開始 268
フラグ・ヘッダー 274
プロセス・フロー 263
ユーザー名の検証 273
要求キャッシング 276
ログイン・ページおよびマクロのサポート 279
HTTP ヘッダーからの認証データの抽出 270
HTTP ヘッダーのリファレンス 275
HTTP ヘッダー名 269
WebSEAL 指定 (自動) リダイレクト 277
鍵 xxi
管理 403
鍵管理
クライアント・サイド証明書の管理 404
サーバー・サイド証明書の管理 404
CRL キャッシュの構成 410
CRL 検査の構成 409
LMI 404
WebSEAL 鍵データベース・スタンザ・エントリーの構成 405
WebSEAL スタンザ・エントリーの概要 403
鍵ファイル情報 178
拡張属性
資格サービス 249
資格情報 249, 677
トークンへの追加 655
CDSSO 627
cdsso トークンから 628
document-cache-control (POP 拡張属性) 45
ECSSO 655
ecssso トークンからの抽出 656
HTTP-Tag-Value (junction 属性) 252, 677
reauth (POP 拡張属性) 195
tagvalue_user_session_id (credential attribute) 677

拡張属性 (extended attribute)

junction 677

仮想ホスト

オブジェクト・スペース内での表示

551

仮想ホスト junction の概念 547

ドメイン Cookie に関する注意点 568

ドメイン Cookie の使用 566

e-community 654

e-community の使用 564

junction 構成 551

junction スロットル 568

junctions 547

SSL セッション ID 使用不可 571

WebSEAL で使用する 564

仮想ホスト junction

インターフェース 558

概念 547

仮想ホスト・ラベル 553

機能 548

基本概念 548

サーバーの追加 582

作成 359, 576

シナリオ 1: 基本的な仮想ホスト

junction 556

シナリオ 2: インターフェース構成

560

シナリオ 3: 仮想ホストの拡張構成

568

制限 570

動的 URL 566

標準 junction でのフィルター要件

547

無視されるスタンザ 550

リモート TCP および SSL タイプの

junction の構成 552

ローカル・タイプの junction の構成

554

URL フィルターの課題 548

/pkmslogout 454

仮想ホスト junction (-v) 470

仮想ホスト機能 (virtual hosting) 541

仮想ホスト・ラベル 553

監査

イベント・ロギング 26

WebSEAL サーバー・アクティビティ

ー 26

監査およびロギング 26

監査証跡ファイル 26

管理オブジェクト 9

管理ディレクトリ 85

期限切れパスワード操作 324

技術上の注意点 692

基本認証

構成 160

キャッシュ

サーバー・サイド 235

キャッシュ、HTML 文書のフラッシュ

714

キャッシング、コンテンツ 40

強制終了 (セッション) 377, 378

強制リダイレクト

ホーム・ページへの 229

ロード・バランシング環境内の 670

共通ログ・フォーマット (request.log) 27

許可

決定データ 399

構成 391

許可決定情報 397, 399

決定データ 399

許可サービス 12

許可データベース

概念の更新およびポーリング 393

許可データベースのポーリングの構成

394

更新通知の listen を構成する 394

更新とポーリングの構成 393

許可データベースのポーリング

概念 393

構成 394

区切り文字、ユーザー・マッピング・ルー

ル評価プログラム 218

クライアント

セキュア 694

クライアントとサーバー間のセッション・

アフィニティ

ロード・バランサー 288

クライアント・サイド証明書

概要 164

クラスター

クラスター環境におけるデプロイメン

トの考慮事項 288

クラスター環境におけるフェイルオー

バー処理のオプション 289

再始動 24

再始動プロセス 24

クラスターの再始動

サーバー・タスク・コマンド 24

クラスター・サポート

図 24

グローバル設定

指定 247

グローバル・サインオン (GSO) 600

クロスサイト・スクリプト

ぜい弱性の回避 120

illegal-url-substrings スタンザ 120

クロスサイト・リクエスト・フォージェリ

ー

対策 123

クロスサイト・リクエスト・フォージェリ

ー (CSRF) 121

クロスドメイン・サインオン

CDSSO 617

e-community 635

形式、ユーザー・マッピング・ルール

218

継承された ACL ポリシー 11

研修 xxiii

更新通知の listen

概念 393

構成 394

構成 512

ランタイム・セキュリティ・サービ

ス外部許可サービス (EAS) 140,

143

構成データ・ログ・ファイル 28

location 28

構成ファイル

一般ガイドライン 703

ストリング値 704

整数値 705

定義済みストリング 704

デフォルト値 703

ブール値 705

ファイル名 704

構成ファイル (WebSEAL) 37

各インスタンス用の 666

スタンザ 38

設定の変更 38

組織 38

名前とロケーション 38

構造

アカウント (account) 694

group (グループ) 694

構文の読み取り 708

後方互換性

HTML サーバー応答ページ 98

コマンド・リファレンス

pdadmin 707

コンテナ、UMI の 215

コンテナ名

UMI コンテナ用 215

コンテンツ・キャッシュ 40

概念 41

構成 41

構成の条件 42

条件 44

スタンザ 40, 41

フラッシュ、キャッシュの 45

document-cache-control (POP 拡張属性)

45

[サ行]

サーバー応答ページ

カスタム・ヘッダーの追加 101

サーバー応答ページ (HTML) 70

[サ行]

サーバー応答ページ	
カスタム・ヘッダーの追加	101
サーバー応答ページ (HTML)	79

サーバー応答ページ (HTML) (続き)
 アカウント管理ページのロケーション 85
 アカウント期限切れエラー・メッセージの構成 94
 イメージの追加 93
 エラー・ページ・ロケーション 85
 カスタマイズ・ガイドライン 87
 構成ファイル・エントリーおよび値 94
 時刻エラー・ページの使用可能化 97
 新規 HTML エラー・メッセージ・ページの作成 97
 パスワード・ポリシー・オプションの構成 95
 マクロ・データ・ストリング形式 89
 マクロ・リソース 87
 マルチロケール・サポート 98
 WebSEAL の前のバージョンとの互換性 98
サーバー管理 21
サーバー識別 (HTTP) の抑止 124, 125
サーバー証明書の DN 値
 取得 374
サーバー相対パス名、定義 476
サーバー名表示 407
サーバー・クラスター
 クラスター環境におけるデプロイメントの考慮事項 288
 クラスター環境におけるフェイルオーバー処理のオプション 289
サーバー・サイド要求キャッシング 233
サーバー・サイド・アプリケーション・サポート 671
サーバー・サイド・キャッシュ
 構成 235
 要求 233
 要求プロセス・フロー 234
サーバー・サイド・キャッシュ要求
 概念 233
サーバー・タスク・コマンド
 trace 401
サーバー・ヘッダー (HTTP) 124, 125
再処理要求 511
最大並行セッション・ポリシー 375
 ユーザー切り替えを使用 380
再認証
 概念 193
 再認証用のログイン・フォームのカスタマイズ 200
 サポート 192
 セキュリティ (POP) ポリシーに基づく 193
 セッション非アクティブに基づく 193
 セッション・キャッシュ・エントリーの除去の回避 198

再認証 (続き)
 セッション・キャッシュ・エントリーの存続時間値の延長 197
 セッション・キャッシュ・エントリーの存続時間値のリセット 196
 ログイン失敗ポリシー制限でのユーザー・セッションの削除 199
 reauth POP 拡張属性 195
 reauth-extend-lifetime スタンザ・エントリー 197
 reauth-for-inactive スタンザ・エントリー 196
 reauth-reset-lifetime スタンザ・エントリー 197
サンプル
 ランタイム・セキュリティ・サービス外部許可サービス (EAS)
 rtss-eas スタンザ 143
資格サービス
 資格情報 249
 amwcredpolsvc 365
 azn_ent_registry_svc 365
資格情報
 拡張属性 (タグ/値) 249, 677
 キャッシュのリフレッシュ 256
 資格サービス 249
 資格情報リフレッシュ 254
 定義 150, 287
 HTTP ヘッダーへの資格情報の挿入 252
 HTTP ヘッダーへのユーザー・セッション ID の挿入 677
 Kerberos 589
 tagvalue_user_session_id credential attribute 677
資格情報交換 271
資格情報属性付与サービス 249
資格情報の処理 249
資格情報リフレッシュ 260
 概要 254
 構成 258
 構成の構文 257
 制限 258
 特定ユーザーの 260
 保存およびリフレッシュ 257
 保存またはリフレッシュする属性の指定 258
 ルール 255
 header-data スタンザ・エントリー (header-names スタンザ) 259
識別、クライアント/ユーザー
 定義 150
システム環境変数 78
システム・リソース 8
自動リダイレクト
 概要 229

自動リダイレクト (続き)
 ホーム・ページへの 229
 マクロの URI エンコード 232
 マクロの内容の長さ 232
 マクロ・サポートの指定 231
 ロード・バランシング環境内の 670
証明書 405
 管理 404
 GSKit 405
 LMI 405
 prompt_as_needed 167
 WebSEAL テスト証明書 411
証明書 SSL ID キャッシュ 172
 disable 174
証明書属性
 有効な 222
証明書取り消しリスト 409
証明書取り消しリスト (CRL)
 CRL キャッシュの構成 410
 CRL 検査の構成 409
証明書認証 164
 オプション・モード 165
 技術上の注意点 174
 構成の要約 167
 遅延 172
 遅延モード 165
証明書認証モード
 要件 164
証明書の失効 408
証明書の分散ポイント 410
証明書の例、XML 216
証明書ユーザー・マッピング・ルールの言語
 言語 214
証明書ログイン・エラー・ページ 170
資料
 アクセス、オンライン xviii
 本製品用のリスト xviii
シングル・サインオフ
 応答 660
 概要 659
 構成 660
 複数の保護された Web リソースから 659
 requests 660
シングル・サインオン 587
 概念 591, 608
 グローバル・サインオン (GSO) 600
 クロスドメイン 617
 結論 695
 セッション・レلم内 381
 フォーム認証 607
 フォーム・ベースの例 615
 プロセスのフロー 608
 BA ヘッダーへのクライアント識別の提供 591

シングル・サインオン (続き)

- CDSSO 617
- e-community 635
- GSO キャッシュの構成 603
- Junction を介したソリューション 587
- LTPA 631, 632
- LTPA (WebSphere) 604
- Windows デスクトップ 174
 - b filter 594
 - b gso 595
 - b ignore 593
 - b supply 592
- シングル・サインオン・ソリューション
 - junctions 473
- 推奨構成例
 - 絶対 URL フィルター操作 672
 - Host ヘッダー情報の提供 (-v) 672
- スクリプトのフィルター操作 (絶対 URL) 487
- スケーラビリティ 18
 - 複製バックエンド・サーバー 20
 - 複製フロントエンド・サーバー 18
- スタンザ
 - session-http-headers 337
- ステートフル junction 449, 450
 - 構成 449
 - 例 452
- スティッキー・ロード・バランシング 321
- ステップアップ認証 (認証強度) 200
- ストリング ID 218
- すべてのユーザー・セッションの終了 681
- スリー・ストライク・ログイン・ポリシー 239
- スロットル
 - junction 455, 568
- 正規表現
 - 動的 URL 用 687
 - フォーム・シングル・サインオン用 613
 - リスト 613
- 制限 176
 - WebSEAL 231
- 制約、ユーザー・マッピング・ルール 218
- セキュリティ
 - 認証トークン保護 626
 - ポリシー 694
 - ポリシー再認証 195
- セキュリティ・ポリシー
 - 計画およびインプリメント 13
 - コンテンツ・タイプの識別 14
 - 保護レベル 14
- セキュリティ・モデルの概念 7

セッション

- キャッシュ・タイムアウトのサポート 192
 - データ・タイプ 182
- セッション Cookie 331
 - 概念 331
 - 条件 332
 - 除去されたセッションに対するカスタム・ログイン応答 334
 - 通常ログアウト時のブラウザーからの Cookie の除去 335
 - 古いセッション Cookie に対するカスタム・ログイン応答 333
 - Cookie 名のカスタマイズ 332
 - resend-webseal-Cookies 333
- セッション ID
 - サポートされるデータ・タイプ 285
 - ユーザー・セッション ID 676
 - WebSEAL セッション ID 149, 285, 286, 676
- セッション ID データ・タイプ 285
- セッション鍵
 - クライアント要求の調査 149, 286
 - 有効なセッション鍵データ・タイプ 329
 - WebSEAL セッション ID 149, 285, 286
- セッション強制終了 334, 377, 378
- セッション共用
 - 一時セッション・キャッシュ 340
 - 概要 339
 - 構成 342, 382
 - 一時キャッシュ応答ページ 341
 - 一時セッション Cookie 名 341
 - 一時セッションの存続時間 340
 - Microsoft SharePoint サーバー 342
- セッション・レルム内 381
- temp-cache-response スタンザ・エントリー 341
- temp-session-cookie-name スタンザ・エントリー 341
- temp-session-max-lifetime スタンザ・エントリー 340
- [acnt-mgt] スタンザ 341
- [session] スタンザ 340, 341
- セッション終了 334, 679
 - すべてのユーザー・セッションの終了 681
 - 単一のユーザー・セッションの終了 680
 - 古いユーザー・セッション ID フォーマット 679
- セッション状態
 - 概要 285
 - クライアントとバックエンド・サーバー間 675

セッション状態 (続き)

- クラスター化された環境 363
- クラスター環境におけるデプロイメントの考慮事項 288
- クラスター環境におけるフェイルオーバー処理のオプション 289
- 異なるトランスポートでの同一セッション鍵 328
- サポートされるセッション ID データ・タイプ 285
- すべてのユーザー・セッションの終了 681
- セッション Cookie 331
- セッション鍵データ・タイプのコントロール 327
- セッション強制終了 377, 378
- セッション・タイムアウトの決定 330
- 単一のユーザー・セッションの終了 680
- 非クラスター環境 327
- 非クラスター環境での維持 327
- フェイルオーバー Cookie 303
- ユーザー・セッション ID 管理 675
- ユーザー・セッション ID 管理の使用可能化 676
- 有効なセッション鍵データ・タイプ 329
- HTTP ヘッダー、使用 336
- Netscape 4.7x の制限 330
- SSL セッション ID 327
- SSL セッション ID キャッシュの構成 294
- WebSEAL セッション・キャッシュ 287
- WebSEAL セッション・キャッシュの構成 295
- セッション存続時間タイム・スタンプ (フェイルオーバー Cookie) 307
- セッションの共用
 - 複数の DNS ドメインでの 353
 - Microsoft Office アプリケーション 339
- セッションの有効期限切れ 279, 281, 296, 297, 299
- セッション非アクティブ再認証 195
- セッション・アクティビティ・タイム・スタンプ (フェイルオーバー Cookie) 307
- セッション・アフィニティ 321
- セッション・キャッシュ
 - 概要と構造 287
 - 構成、WebSEAL 293
 - 構成の要約 293
 - 最大エントリー数 295

セッション・キャッシュ (続き)
除去されたセッションに対するカスタム・ログイン応答 334
すべてのユーザー・セッションの終了 681
制限、WebSEAL 301
セッション強制終了 377, 378
存続時間タイムアウト (グローバル設定) 296
存続時間タイムアウト (ユーザーごとの設定) 279, 297
単一のユーザー・セッションの終了 680
非アクティブ・タイムアウト 299
非アクティブ・タイムアウト (ユーザーごとの設定) 281
古いセッション Cookie に対するカスタム・ログイン応答 333
並行セッションの制限 301
SSL キャッシュの構成 294
SSL セッション・キャッシュ 293
WebSEAL キャッシュの構成 295
WebSEAL セッション・キャッシュ 287, 293
セッション・タイムアウト 279, 281, 296, 297, 299, 330, 334
セッション・レلم 381
絶対 URL フィルター操作の推奨構成例 672
絶対パス名、定義 476
相互認証 junction 443
相対パス名、定義 476
属性
資格情報への追加 250
azn_cred_groups 220
azn_cred_principal_name 220
azn_cred_registry_id 220

[タ行]

タイムアウト
考慮事項 528
タイムアウト設定
cgi-timeout 51
client-connect-timeout 49
HTTP および HTTPS 49
https-timeout (junction) 51
http-timeout (junction) 51
intra-connection-timeout 49
persistent-con-timeout 50
ping-time (junction) 51
SSL セッション・キャッシュ 294
WebSEAL セッション・キャッシュ 295
多重方式プロキシ・エージェント (認証) 181

単一のユーザー・セッションの終了 680
チャンク転送コーディング
サポート 54
通信プロトコル構成 46
データ
イベント相関 683
構成ログ・ファイルの増加 28
ログ・ファイルの形式 29
データ圧縮
制限 64
データ圧縮ポリシーの構成 65
ユーザー・エージェント・タイプに基づく 63
gzip 62
HTTP 62
MIME タイプに基づく 62
POP ポリシー 64
データ・ログ・ファイル
構成メッセージ 29
ディレクトリーの索引付け
構成 39
ディレクトリー索引アイコン 40
ディレクトリー名、表記 85
ディレクトリー・エラー 85
テスト証明書 411
デフォルトのパスワード
ストレングス・ポリシーの値 246
テンプレート
マクロの組み込み 90
トークン
拡張属性の追加 655
抽出する拡張属性 656
トークン作成
vouch-for サーバーでの構成 645
トークン作成機能 621
トークン消費
受信側サーバーでの構成 646
統一された Web スペース
図 19
透過パス Junction
概念 428
構成 429
例 429
透過パス Junction (-x) 427
動的
URL 685
動的 URL
アクセス制御 685
アクセス・コントロールの提供 685
オブジェクト・スペースへのマッピング 694
解決 689
概要 685
更新、dynurl update 689
使用可能化 685
使用不能化 77

動的 URL (続き)
文字エンコードと照会ストリング検証 688
要約および技術上の注意点 691
例 692
ACL オブジェクトのマッピング 687
dynurl-allow-large-posts 690
dynurl-map 687
dynurl.conf 685
GET および POST 要求 686
POST 要求に対する制限の設定 690
request-body-max-read 690
URL 照会ストリングでの動的データの処理 686
URL 照会ストリング内の無効文字の処理 77
特殊な HTML 文字、定義 477
ドメイン Cookie 433, 566
SMS を使用 381
トラブルシューティング xxiii, 61
資格情報リフレッシュ 261
P3P 構成 671
トリガー URL 268
トレーラー (junction Cookie JavaScript の付加) 495

[ナ行]

入力フォーム
設計 190
認証 149
外部認証インターフェース 213, 263
概要 15, 149
基本認証 160
強制リダイレクト 229, 670
強度ポリシーの確立 203
クライアント識別 150
サーバー・サイド要求キャッシング 233
再認証 (セキュリティ・ポリシー) 193
再認証 (セッション非アクティブ) 193
サポートされるセッション ID タイプ 285
サポートされる方式 154
資格情報 150
自動リダイレクト 229, 670
証明書 164
ステップアップ 200
スリー・ストライク・ログイン・ポリシー 239
制限 527
操作 157
多重方式プロキシ・エージェント (MPA) 181

認証 (続き)

- トークン 626
- トークン保護 626
- 認証強度ポリシー (ステップアップ) 200
- パスワード・ストレングス・ポリシー 244
- フェイルオーバー 303
- フォーム 161
- フォーム・シングル・サインオン 607
- 複数レベル 204
- プロセス・フロー 151
- 方式 157
- 方法 157, 182
- メカニズム 157
- モジュール 157
- ユーザー切り替え 185
- リソースへの認証アクセス 152
- リソースへの非認証アクセス 152
- ログアウトおよびパスワード・ユーティリティー 158
- ログイン失敗ポリシー 239
- ログイン・データの直接送付 162
- CDSSO 617
- e-community 635
- handle-auth-failure-at-mas 654
- Kerberos 174
- LTPA 177
- SMS 372
- WebSEAL への情報のマッピング 602
- 認証 Cookie
 - UTF-8 エンコード、LTPA 75
- 認証強度ポリシー (ステップアップ) 200
 - 概念 201
 - 構成タスクの要約 202
- 認証後の処理 229
- 認証済みユーザー
 - 認証プロセス・フロー 152
- 認証データ
 - 各要求に含まれる 101, 289
- 認証方式
 - 拡張 181
- 認証方式の要約 154

[ハ行]

- パーソナライゼーション・サービス
 - 概要 673
 - 例 674
 - WebSEAL の構成 674
- パス名、表記 85
- パスワード
 - 処理 239
 - 無効な例 246
 - 有効な例 246
- パスワードの期限切れ 95, 324

パスワード変更

- パスワード・ストレングス・ポリシー 244
- pkmspasswd 159
- Windows 上での Active Directory の問題 159
- パスワード変更操作 324
- パスワード・ストレングス
 - 構文 244
 - ポリシーの概念 244
 - policy コマンド 244
- パスワード・ストレングス・ポリシー 244
- パスワード・ポリシーの構成 95
- パターン・マッチング (ワイルドカード文字) 77
- バックエンド・アプリケーション・サポート 671
- 非アクティブ・タイムアウト・スタンザ・エントリー
 - session スタンザ 281
- 引数スタンザ 614
- 非クラスター環境
 - セッション状態の維持 327
- 非スティッキー・フェイルオーバー・ソリューション 321
- 非スティッキー・ロード・バランシング 321
- 非認証ユーザー
 - 制御 152
 - 認証プロセス・フロー 152
 - 非認証 HTTPS の使用 154
 - ユーザーの強制ログイン 153
 - SSL を介したアクセス条件 153
- 評価
 - ACL および POP 689
- 評価プログラム、ユーザー・マッピング・ルール 217
- 表記
 - パス名 85
- フィルター操作時の Content-Length ヘッダーの変更 489
- フェイルオーバー
 - 環境 303
 - 新規マスター 288
 - Cookie 290
 - WebSEAL サーバーから別の WebSEAL サーバーへ 289
- フェイルオーバー Cookie
 - 拡張属性 318
 - 欠落したフェイルオーバー Cookie の再発行 315
 - セッション存続時間タイム・スタンプ 307, 316
 - セッション・アクティビティ・タイム・スタンプ 307

フェイルオーバー Cookie (続き)

- 抽出、データの 309
- データおよび属性の追加 306
- resend-webseal-Cookies 333
- フェイルオーバー環境 349
- フェイルオーバー認証
 - 概要 303
 - 拡張属性の追加 318
 - 欠落したフェイルオーバー Cookie の再発行 315
 - 構成 311
 - 構成ステップ 312
 - 構成の例 306
 - 使用可能化 313
 - セッション存続時間タイム・スタンプ 307, 316
 - セッション・アクティビティ・タイム・スタンプ 307, 316
 - 抽出のための属性の指定 318
 - 追加、認証レベルの 315
 - ドメイン全体 310
 - ドメイン全体の Cookie の使用可能化 319
- パスワード変更操作 324
- 非スティッキー・フェイルオーバーの概念 321
- 非スティッキー・フェイルオーバー・ソリューション 321
- 非スティッキー・フェイルオーバー・ソリューションの構成 323
- フェイルオーバー Cookie の説明 305
- フェイルオーバー Cookie のデータおよび属性 306
- フェイルオーバー Cookie 用の UTF-8 エンコード 314
- プロセス・フロー 305
- change-password-auth スタンザ・エントリー 324
- Cookie データの暗号化 313
- Cookie データの抽出 309
- Cookie の存続時間の構成 314
- LMI
 - SSO キー管理ページ 313
- SSO キー管理 313
- フェイルオーバー・イベント
 - WebSEAL が処理しない 289
- フォーム、ログイン
 - イメージの追加 93
 - カスタマイズ 86
- フォーム認証 161
 - シングル・サインオン・ソリューション 607
- フォーム・ベース
 - シングル・サインオンの例 615
- フォーム・ベースのログイン 615

複製
 junction データベース 422
複製、フロントエンド WebSEAL サーバーの 668
プライマリー・インターフェース
 IP アドレス 664
プラグイン
 OAuth EAS 42
フラッシュ、キャッシュの 45
古いセッション Cookie
 カスタマイズ応答の使用可能化 335
プロキシー・エージェント
 多重の概要 181
プログラム出力 435
プログラムの入力 435
フロントエンド WebSEAL サーバー
 複製 668
 login_success 応答のコントロール 669
文書キャッシュ参照、コンテンツのキャッシングを 40
文書のキャッシュ、HTML フラッシュ 714
文書モデル、UMI の 214
並行セッション・ポリシー
 最大 360
ヘッダー
 変更 516
変換エラー 524
保護オブジェクト 8
保護オブジェクト・スペース 8
 管理オブジェクト 9
 システム・リソース 8
 保護オブジェクト 8
 ユーザー定義オブジェクト 9
 Web オブジェクト 8
 WebSEAL server-name 36
保護オブジェクト・ポリシー 10
保護オブジェクト・ポリシー (POP)
 外部許可サービス (EAS)
 ランタイム・セキュリティ・サービス外部許可サービス (EAS) 140
 リスクベース・アクセス 140
 ランタイム・セキュリティ・サービス外部許可サービス (EAS) 140
保護品質
 デフォルトのレベル 394
 ネットワーク 396
 ホスト 396
 POP 393
保守容易性メッセージ 26
「保証」
 応答 644
 要求 643

ポリシー
 セキュリティ 694
 disable-time-interval 241
 max-concurrent-web-sessions 375
 max-login-failures 240
 max-password-repeated-chars 244
 min-password-alphas 244
 min-password-length 244
 min-password-non-alphas 244
 password-spaces 244
ポリシー・エンフォーサー 12

[マ行]

マクロ
 エンコード 90
 テンプレート 91
 テンプレートへの組み込み 90
 JavaScript 92
マクロ、HTML サーバー応答ページで使われる
 AUTHNLEVEL 87
 BACK_NAME 87
 BACK_URL 87
 BASICAUTHN 87
 CERTAUTHN 87
 EAIAUTHN 87
 ERROR 87
 ERROR_CODE 87
 ERROR_TEXT 87
 EXPIRE_SECS 87
 FAILREASON 87
 HOSTNAME 87
 HTTPS_BASE 87
 HTTP_BASE 87
 LOCATION 87
 METHOD 87
 OLDSESSION 87, 334, 335, 377, 378
 PROTOCOL 87
 REFERER 87
 REFERER_ENCODED 87
 STEPUP 87
 URL 87
 URL_ENCODED 87
 USERNAME 87, 200
マクロの URI エンコード 111, 232
マクロ・サポート
 HTML サーバー応答ページ 87
マシン名
 解決 622
マスター認証サーバー (MAS) 636
マルチロケール・サポート 65, 98
 条件 100
 プロセス・フロー 99
 Accept-Language HTTP ヘッダー 98
マルチロケール・サポート構成 70

明示的な ACL ポリシー 11
メッセージ 26
 メッセージ ID フォーマット 26
 ルーティング・ファイル 26
 Tivoli Common Directory 26
 Tivoli メッセージ・フォーマット 26
 UTF-8 形式のメッセージ 26
 WebSEAL ログ・ファイル 26
文字エンコード方式、要求内のデータの検証 76
モジュールのマッピング
 正常に行われたことの検証 227
モデル、文書 214
問題判別 xxiii, 28
 構成データ・ログ・ファイル 28
 statistics ユーティリティ 30
 trace ユーティリティ 30

[ヤ行]

ユーザー ID
 突き合わせ、ステップアップとの 211
ユーザー切り替え
 概要 185
 機能のサポート 192
 構成 188
最大並行セッション・ポリシー 380
使用法 191
ユーザーの除外 188
有効な認証方式 190
 securitygroup 188
 su-admin 拡張属性 193
 su-admins グループ 185, 188
 su-excluded グループ 188
ユーザー設定 247
 指定 247
ユーザー定義オブジェクト 9
ユーザー・エージェント
 認証ユーザー確認質問 155
ユーザー・セッション ID 管理 675
 概念 675
 使用可能 676
 すべてのユーザー・セッションの終了 681
 単一のユーザー・セッションの終了 680
 古いユーザー・セッション ID フォーマット 679
 ユーザー・セッション ID ストリングのフォーマット 679
 HTTP ヘッダーへのユーザー・セッション ID の挿入 677
 tagvalue_user_session_id 676
 user-session-ids 676
ユーザー・セッション管理
 サポート 193

ユーザー・マッピング・ルール
区切り文字 218
形式および制約 218
評価プログラム 217
例
資格付与データの UMI 220
要求、クライアントを調査
セッション鍵 286
認証データ 149
要求キャッシング 233
要求ヘッダー
WebSEAL コンテンツ・キャッシュへの影響 43
用語集 xviii

[ラ行]

ランタイム・セキュリティ・サービス 139
ランタイム・セキュリティ・サービス外部許可サービス (EAS)
構成 140
サンプル構成 143
定義 139
保護オブジェクト・ポリシー (POP) 140
リスクベース・アクセス 139
rtss-eas スタンザ 143
リスクベース・アクセス
ランタイム・セキュリティ・サービス外部許可サービス (EAS)
構成 140
サンプル構成 143
定義 139
リスト
Server 710
リソースへの認証アクセス 152
リソースへの非認証アクセス 152
リソース・マネージャー 12
リダイレクト、自動
ホーム・ページへの 229
ロード・バランシング環境内の 670
リダイレクト、ローカル応答 103
リダイレクト、HTML 115
リバース・プロキシ管理ページ 85
管理ルート 21
再始動
WebSEAL インスタンス 21
ログイン 21
Junction の管理 21
LMI
リバース・プロキシ管理ページ 21
WebSEAL インスタンス
管理 21
「リバース・プロキシ管理ルートの管理」ページ 85
ルーティング・ファイル 26
形式と操作 26
テンプレートの場所 26
メッセージ重大度レベル 26
メッセージ・ログ・ロールオーバー 26
ルート要求
プロセス・フロー 499
レジストリー属性資格サービス 249
構成 250
レベル (認証) 203
ローカル junction
disable 426
ローカル junction の無効化 426
ローカル応答リダイレクト 103
概要 104
技術上の注意点 113
構成の例 112
使用可能化および使用不可化 105
操作の指定 106
プロセス・フロー 104
マクロの URI エンコード 111
マクロの内容の長さ 111
マクロ・サポートの指定 108
マクロ・フィールド名のカスタマイズ 110
リダイレクトされた応答の内容 105
ロケーション URI の指定 105
ロード・バランシング環境 668
スティッキー 321
非スティッキー 321
フロントエンド WebSEAL サーバーの複製 668
login_success 応答のコントロール 669
ログ
HTTP 27
HTTP (イベント・ログ) 26
ログイン
スリー・ストライク・ログイン・ポリシー 239
パスワード・ストレングス・ポリシー 244
非認証ユーザーのステップアップ・ログインの制御 212
古いセッション Cookie に対するカスタム・ログイン応答 333
ユーザーの強制ログイン 153
ログイン失敗の通知 (SMS) 385
ログイン失敗ポリシー 239
ログイン・データの直接送付 162
ログイン・ヒストリー (SMS) 385
ログイン試行
回数を減少させる 243

ログイン失敗の通知 (SMS) 385
ログイン失敗ポリシー 239
ログイン・スタンザ・エントリー (ログイン・フォーム) 94
ログイン・ヒストリー (SMS) 385
ログイン・フォーム
イメージの追加 93
カスタマイズ 86
ログイン・フォームのカスタマイズ 86
イメージの追加 93
ログ・データ
utf8 エンコード 27
ロケーション・ヘッダー、ローカル応答リダイレクト 105

[ワ行]

ワーカー・スレッド 59
管理 58
グローバル割り振り 60
考慮事項 529
junction 単位の割り振り 61
Junction フェアネス 59
WebSEAL 58
ワイルドカード文字 77

A

Accept-Charset HTTP ヘッダー 100
accept-client-certs スタンザ・エントリー 167, 173, 558
Accept-Language HTTP ヘッダー 98, 100
account-expiry-notification スタンザ・エントリー 94
account-inactivated スタンザ・エントリー 95
account-locked スタンザ・エントリー 94
acct_locked.html 79
ACL
許可 392
ACL および POP
小文字オブジェクト名への付加 470
ACL ポリシー
概念 10
概要 9
継承 11
明示 11
ACL 名に使用できる文字 392
WebSEAL 固有の 391
acct-mgt スタンザ 205, 230
allowed-referers 122
enable-secret-token-validation 121
agents スタンザ・エントリー 27
agents-file スタンザ・エントリー 27
agent.log 27

allowed-referers
 acnt-mgt スタンザ 122
allow-backend-domain-cookies スタンザ・エントリー 433
allow-unauthenticated-logout
 [acnt-mgmt] スタンザ・オプション 454
allow-unsolicited-logins
 server スタンザ 123
amwcredpolsvc 資格サービス 365
argument-stanza スタンザ・エントリー 611
auditcfg スタンザ・エントリー 27
auditlog スタンザ・エントリー 27
authentication-levels スタンザ 203, 207
AUTHENTICATION_LEVEL スタンザ・エントリー 315
AUTHNLEVEL マクロ 279
authtoken-lifetime スタンザ・エントリー 624
aznapi-configuration スタンザ 26, 394
aznapi-external-authzn-services スタンザ
 ランタイム・セキュリティ・サービス
 ス外部許可サービス (EAS) 140
azn-decision-info スタンザ
 ランタイム・セキュリティ・サービス
 ス外部許可サービス (EAS) 140
azn_cred_groups 属性 220
azn_cred_principal_name 属性 220
azn_cred_registry_id 属性 220
azn_entitlement_get_entitlements() メソッド 218
azn_ent_registry_svc 資格サービス 365

B

b filter 596
b ignore 595
b supply 595
 オプション 593
 制限 593
backicon 40
basicauth-dummy-passwd スタンザ・エントリー 592
basic-auth-realm 161
ba-auth スタンザ・エントリー 160
bind-dn (LDAP 構成) 58
bind-pwd (LDAP 構成) 58

C

c junction
 使用の条件 598
 例 598
CA 証明書
 import 358
cache-refresh-interval スタンザ・エントリー 394
CDAS
 概要 213
 管理 221
 ルールの例 213
CDMF ライブラリー 618
 cdmf_get_usr_attributes 627
cdmf_get_usr_attributes 627
CDSSO 617
 構成 621
 トークン消費機能 621
 トークン作成機能 621
 トークン消費機能 621
CDSSO 認証 617
 概要 617
 拡張ユーザー属性 618
 仮想ホストを使用したクロスドメイン・シングル・サインオン 565, 627
 構成 621
 構成の条件 622
 構成の要約 621
 デフォルトおよびカスタム・トークン 618
 プロセス・フロー 618
 ID マッピング 618
 utf-8 629
cdsso-argument スタンザ・エントリー 625
cdsso-auth スタンザ・エントリー 623
cdsso-create スタンザ・エントリー 623
cdsso-incoming-attributes
 WebSEAL 構成ファイル 628
cdsso-incoming-attributes スタンザ 627
cdsso-peers スタンザ 623
cdsso-token-attributes スタンザ
 WebSEAL 構成ファイル 628
certfailure.html 79
certificate-login スタンザ・エントリー 94, 171
certlogin.html 79
certstepuphttp.html 79
cert-cache-max-entries 172
cert-cache-timeout スタンザ・エントリー 172
cert-failure スタンザ・エントリー 94
cert-ssl スタンザ・エントリー 168
cert-stepup-http スタンザ・エントリー 94, 173
cgi-timeout スタンザ・エントリー 51
change-password-auth スタンザ・エントリー 324
client-connect-timeout スタンザ・エントリー 49

client-notify-tod 97
compress-mime-types stanza 62
compress-user-agents スタンザ 63
concurrent-session-threads-hard-limit スタンザ・エントリー
 server スタンザ 301
concurrent-session-threads-soft-limit スタンザ・エントリー
 server スタンザ 301
config-data-log 28
content-index-icons スタンザ 40
Cookie
 固有の Cookie 名前属性の確認 (-I) 504
 指定した名前の保持 503
 すべての名前の保持 503
 セッション 331, 466
 ドメイン 433, 566
 ドメイン (SMS を使用) 381
 フェイルオーバー 290
 変更 519
 hostname-junction-cookie スタンザ・エントリー 448
 IV_JCT (junction Cookie ヘッダー) 493
 junction Cookie JavaScript のコントロール (-J) 495
 junction Cookie JavaScript の付加 (-J トレーラー) 495
 junction Cookie のリセット (-J onfocus) 496
 junction Cookie、回避、ネーミング競合 448
 LTPA 291
 WebSEAL での管理 464
 -j junction 全体の Cookie の保存 503
 -j およびマッピング・テーブル junction の Cookie パスの変更 501
 -j およびマッピング・テーブル junction の Cookie 名の変更 502
Cookie 名
 クライアント 179
 Junction 179
credential-refresh-attributes スタンザ 257, 324
cred-attribute-entitlement-services スタンザ・エントリー 365
CRL 409
CRL キャッシュ、構成 410
 gsk-crl-cache-entry-lifetime 411
 gsk-crl-cache-size 410, 411
CRL 検査、構成 409
crl-ldap-server スタンザ・エントリー
 junction スタンザ 410
 ssl スタンザ 409, 412

crl-ldap-server-port スタンザ・エントリー
 junction スタンザ 410
 ssl スタンザ 409, 412
crl-ldap-user スタンザ・エントリー
 junction スタンザ 410
 ssl スタンザ 409, 412
crl-ldap-user-password スタンザ・エントリー
 junction スタンザ 410
 ssl スタンザ 409, 412
CSRF アタック
 ぜい弱性の回避 121
CSRF 対策
 秘密トークンの検証 121
 リファラー検証 122
 allow-unsolicited-logins 123

D

DB2 xxi
decode-query スタンザ・エントリー 76
default-webseal ACL ポリシー 392
default.html 79
deletesuccess.html 79
directory-index 39
directory-index スタンザ・エントリー 39
diricon 40
disable-ec-cookie 653
disable-local-junctions 426
disable-ssl-v2 スタンザ・エントリー
 junction スタンザ 425
 ssl スタンザ 47
disable-ssl-v3 スタンザ・エントリー
 junction スタンザ 425
 ssl スタンザ 47
disable-time-interval ポリシー 239
disable-tls-v1 スタンザ・エントリー
 junction スタンザ 425
 ssl スタンザ 47
disable-tls-v11 スタンザ・エントリー
 ssl スタンザ 47
disable-tls-v12 スタンザ・エントリー
 ssl スタンザ 47
document-cache-control (POP 拡張属性)
 45
domain スタンザ・エントリー 384
dsess-cluster スタンザ 364
dsess-enabled スタンザ・エントリー 363
dsess-last-access-update-interval スタンザ・
 エントリー 370
dsess-sess-id-pool-size スタンザ・エントリー
 372
dynurl-allow-large-posts スタンザ・エント
 リー 690
dynurl-map スタンザ・エントリー 77,
 687

dynurl.conf 685

E

EAI
 外部認証インターフェース 263
eai スタンザ 273, 274
EAIAUTHN マクロ 279
eai-auth スタンザ・エントリー 267
eai-auth-error スタンザ・エントリー 94
eai-auth-level-header 270
eai-auth-level-header スタンザ・エントリー
 275, 278
eai-flags-header 270
eai-flags-header スタンザ・エントリー
 274
eai-pac-header 270
eai-pac-header スタンザ・エントリー 275
eai-pac-svc-header 270
eai-pac-svc-header スタンザ・エントリー
 275
eai-redirect-url-header 270
eai-redirect-url-header スタンザ・エントリー
 276, 277
eai-trigger-urls スタンザ 269
eai-user-id-header 270
eai-user-id-header スタンザ・エントリー
 275
eai-verify-user-identity スタンザ・エントリ
 ー 273
eai-xattrs-header 270
eai-xattrs-header スタンザ・エントリー
 275
EAS
 トレース情報 401
 OAuth 397
ECSSO 655
 拡張属性 655
ecssso-allow-unauth 653
ec-cookie-lifetime スタンザ・エントリー
 652
enabled (LDAP 構成) 58
enable-failover-cookie-for-domain スタン
 ザ・エントリー 319
enable-html-redirect スタンザ・エントリー
 115
enable-local-response-redirect スタンザ・エ
 ントリー 105
enable-passwd-warn スタンザ・エントリー
 95
enable-redirects スタンザ 230
enable-secret-token-validation
 acct-mgt スタンザ 121
enforce-max-sessions-policy スタンザ・エン
 トリー 379
ERROR エラー・メッセージ 26

error-dir スタンザ・エントリー 85
Extensible Markup Language 214
Extensible Style Language 214
e-community
 仮想ホスト 654
 シングル・サインオン 635
 シングル・サインオン構成 645
 マシン名の解決 647

e-community Cookie 643
 resend-webseal-Cookies 333

e-community 認証 635
 概要 635
 仮想ホストの使用 564
 機能 637
 構成の要約 645
 条件と要件 646
 ドメイン鍵 649
 非認証アクセス 653
 プロセス・フロー 637
 マスター認証サーバー 636
e-community Cookie 643
 utf-8 657
 vouch-for トークン 644
 vouch-for トークンの暗号化 648
 vouch-for 要求と応答 643

e-community-domains スタンザ 564

e-community-domain-keys スタンザ 648,
 649

e-community-domain-keys:domain スタンザ
 564

e-community-name スタンザ・エントリー
 648

e-community-sso-auth スタンザ・エントリ
 ー 647

F

failover-add-attributes スタンザ 323
failover-auth スタンザ・エントリー 313
failover-Cookies-keyfile 313
failover-cookie-lifetime スタンザ・エント
 リー 314
failover-include-session-id スタンザ・エン
 トリー 323
failover-require-activity-timestamp-validation
 スタンザ・エントリー 320
failover-require-lifetime-timestamp-validation
 スタンザ・エントリー 320
failover-restore-attributes スタンザ 318,
 323
failover-update-cookie スタンザ・エントリ
 ー 317
FATAL エラー・メッセージ 26
favicon.ico
 Mozilla Firefox における 404 問題
 100

Federated Identity Manager (TFIM) サポート

セッション・キャッシュ・エントリー
存続時間 (ユーザーごとの設定)

279, 297

セッション・キャッシュ・エントリー
非アクティブ (ユーザーごとの設定)

281

HTTP ヘッダーを使用したセッション
状態の維持 336

filter

応答の中の URL の変更 478

サーバー相対 URL のルール 479

新規コンテンツ (MIME) タイプの指定

482

スクリプトのフィルター操作 (絶対
URL) 487

スクリプト・フィルター操作による絶
対 URL のフィルター操作 487

絶対 URL に関する推奨構成例 672

絶対 URL のルール 480

相対 URL のフィルター操作のルール

479

タグおよび属性の指定 482

タグ・ベース URL フィルター操作

478

タグ・ベースの URL フィルター操作
のルール 479

タグ・ベースの静的 URL のデフォル
ト・フィルター操作 480

バックエンド・アプリケーションから
の URL の変更 475

フィルター掛けされていないサーバー
相対リンク 489

フィルター・スキーム 486

複数の -j junction にわたる

Cookie 501

要求の中の URL の変更 490

要求の中のサーバー相対 URL 処理の
コントロール 499

Content-Length ヘッダーの変更 489

filter-nonhtml-as-xhtml 482

HTML BASE HREF タグ 483

HTML META リフレッシュ・タグ

483

junction Cookie を使用したサーバー相
対 URL の変更 493

junction マッピングによるサーバー相
対 URL の変更 491

preserve-base-href スタンザ・エントリー

483

preserve-base-href2 スタンザ・エントリー

483

process-root-requests スタンザ・エン
トリー 499

filter (続き)

Referer ヘッダーを使用したサーバー相
対 URL の変更 498

rewrite-absolute-with-absolute 488

URL で使用されるパスのタイプ 476

URL の特殊な HTML 文字 477

URL の変更の概念 475

X-Old-Content-Length 489

filter-content-types スタンザ 480

text/html 480

text/vnd.wap.wml 480

filter-nonhtml-as-xhtml 482

filter-schemes スタンザ 486

filter-url スタンザ 480, 482

Firefox、Mozilla

favicon.ico 404 問題 100

flush-time スタンザ・エントリー 27

forms-auth スタンザ・エントリー 161

forms-sso-login-pages スタンザ 611

G

GET 要求 686

gmt-time スタンザ・エントリー 27

gskcapicmd xxi

gskikm.jar xxi

GSKit 405

構成 589

資料 xxi

CRL キャッシュ 410

SMS 接続用の構成 375

GSKit 属性

SSL 接続 589

gsk-attr-name スタンザ・エントリー

[ssl] スタンザ 119

gsk-crl-cache-entry-lifetime スタンザ・エン
トリー 411, 412

gsk-crl-cache-size スタンザ・エントリー

410, 411, 412

GSO 600

概要 601

メカニズムの図示 601

GSO キャッシュの構成 603

-b 596

GSO キャッシュ、構成 603

GSO 使用可能例

WebSEAL junction 603

gso-cache-enabled スタンザ・エントリー

603

gso-cache-entry-idle-timeout スタンザ・エ
ントリー 603

gso-cache-entry-lifetime スタンザ・エン
トリー 603

gso-cache-size スタンザ・エントリー 603

gso-resource スタンザ・エントリー 611

gzip 62

H

handle-auth-failure-at-mas 654

handle-pool-size スタンザ・エントリー

372

header-data スタンザ・エントリー

(header-names スタンザ) 259

help スタンザ・エントリー 94

help.html 79

host (LDAP 構成) 58

Host ヘッダー

仮想ホスト junction 548

junction に関する推奨構成例 672

hostname-junction-cookie スタンザ・エン
トリー 448

HTML

属性 92

タグ 92

HTML 4.01 準拠のための JavaScript プロ
ック (inhead)

挿入 495

HTML サーバー応答

ページのロケーション 84

HTML サーバー応答ページ 79

アカウント管理ページのロケーション

85

アカウント期限切れエラー・メッセー
ジの構成 94

イメージの追加 93

エラー・ページ・ロケーション 85

カスタマイズ・ガイドライン 87

構成ファイル・エントリーおよび値

94

時刻エラー・ページの使用可能化 97

新規 HTML エラー・メッセージ・ペ
ージの作成 97

パスワード・ポリシー・オプションの
構成 95

マクロ・データ・ストリング形式 89

マクロ・リソース 87

マルチロケール・サポート 98

WebSEAL の前のバージョンとの互換
性 98

HTML 文書のキャッシュのフラッシュ

714

HTML リダイレクト 115

使用可能化と使用不可化 115

HTML フラグメントの保持 115

HTTP

持続接続 48

データ圧縮 62

変換のシナリオ 513

HTTPOnly Cookie 48

HTTP BA ヘッダー

サインオン 590

http イベント・プール 26

HTTP 応答の置換 509
HTTP 共通ログ・フォーマット 27
HTTP サーバー識別
 抑止 124, 125
HTTP サポート
 RPC 525
 WebSEAL 525
HTTP 持続接続 48
http スタンザ・エントリー 46, 558
HTTP ヘッダー 599
 イベント相関の挿入 683
 サーバー 125
 サイズの制限 600
 セッション状態、維持 336
 提供される ID 情報 596
 ホスト 548
 ロケーション 102
 Accept-Charset 100
 Accept-Language 98, 100
 HTTP_IV_CREDS 596
 HTTP_IV_GROUPS 596
 HTTP_IV_REMOTE_ADDRESS 599
 HTTP_IV_USER 596
 HTTP_IV_USER_L 596
 Referer ヘッダー、サーバー相対 URL
 の変更 498
HTTP ヘッダー内の LDAP データ 249
HTTP ヘッダー・セッション
 基本概念 336
HTTP 変換 507, 509, 510, 511, 512
 Extensible Stylesheet Language
 Transformation 508
 HTTP 応答オブジェクト 509
 HTTP 要求オブジェクト 508
 XSLT 508
HTTP 変換ルール 507
HTTP メソッドの使用不可化 125
HTTP 要求
 応答の生成 522
 WebSEAL の構成 46
HTTP ログ 27
 イベント・ログを使用する 26
 使用可能化および使用不可化 27
 タイム・スタンプ 27
 バッファのフラッシュ頻度 27
 ログ・ファイル・ロールオーバーのし
 きい値 27
 HTTP 共通ログ・フォーマット 27
 utf8 エンコード 27
HTTP を介した Microsoft RPC
 サポート 54
HTTP を介した RPC 525
HTTPOnly Cookie 48
HTTPResponse 519
 応答行の変更 518
 ヘッダーの変更 516

HTTPResponse (続き)
 Cookie の変更 519
 URI の変更 513
HTTPResponseChange 522
https スタンザ・エントリー 47, 558
https-port スタンザ・エントリー 47, 558
https-timeout スタンザ・エントリー
 (junction) 51
http-method-disabled-local スタンザ・エン
 トリー 125
http-method-disabled-remote スタンザ・エン
 トリー 125
http-port スタンザ・エントリー 47, 558
HTTP-Tag-Value
 junction 属性の処理 678
 junction の拡張属性 677
 junction の属性の設定 678
HTTP-Tag-Value junction 属性 252, 677
HTTP-Tag-Value 属性 253
http-timeout スタンザ・エントリー
 (junction) 51
http-transformations スタンザ 512
http.agent イベント・プール 26
http.clf イベント・プール 26
http.cof イベント・プール (NCSA) 26
http.ref イベント・プール 26
HTTP/1.1 応答 433
HTTP_IV_CREDS ヘッダー 596
HTTP_IV_GROUPS ヘッダー 596
HTTP_IV_REMOTE_ADDRESS ヘッダー
 599
HTTP_IV_USER ヘッダー 596
HTTP_IV_USER_L ヘッダー 596
HTTP_PD_USER_SESSION_ID 253

I

IBM
 ソフトウェア・サポート xxiii
 Support Assistant xxiii
IBM Security Access Manager
 WebSEAL 3
IBM Security Web Gateway Appliance 5
ICAP 698
 構成 699
 サポート 697
ICAP サポート 697
ICAP と WebSEAL 698
icons スタンザ 40
iKeyman xxi
illegal-url-substrings スタンザ 120
inactive-timeout スタンザ・エントリー
 セッション スタンザ 330
 session スタンザ 299, 317
interfaces スタンザ 558

Internet Content Adaptation Protocol 697,
 698, 699
intra-connection-timeout スタンザ・エン
 トリー 49
IP Endpoint Authentication Method
 Policy 206
ipauth 207
IPv4 54
 アップグレード 56
 概要 54
 互換性のサポート 56
 ipv6-support スタンザ・エントリー
 55
 ip-support-level 57
IPv6 54
 アップグレード 56
 概要 54
 互換性のサポート 56
 ipv6-support スタンザ・エントリー
 55
 ip-support-level 57
ipv6-support
 互換性のサポート 56
 資格情報属性 57
 WebSEAL アップグレード 56
ipv6-support スタンザ・エントリー 55
ip-support-level スタンザ・エントリー 57
is-master-authn-server スタンザ・エン
 トリー 650
ivadmin_server_performtask() 関数 680
iv-creds ヘッダー 596, 671
iv-groups ヘッダー 596, 671
iv-remote-address ヘッダー 599, 671
iv-remote-address-ipv6 ヘッダー 671
iv-user ヘッダー 596, 671
iv-user-l ヘッダー 596, 671
IV_JCT (junction Cookie ヘッダー) 493
 junction Cookie JavaScript のコント
 ロール (-J) 495
 junction Cookie JavaScript の付加 (-J
 トレーラー) 495
 junction Cookie のリセット (-J
 onfocus) 496

J

JMT 730
jmt-map 491
jmt.conf 491
JSP
 ログイン・ヒストリーへのアクセス
 386
 ログイン・ヒストリーを表示するため
 のカスタマイズ 387
junction
 拡張構成 443

junction (続き)

- 拡張属性 (extended attribute) 677
- 構成 526
- 作成 533
- 初期サーバー用の作成 533
- 静的サーバー応答ページ 86
- タイプ 418

junction Cookie 493

- junction Cookie JavaScript のコントロール (-J) 495

- junction Cookie JavaScript の付加 (-J トレーラー) 495

- junction Cookie のリセット (-J onfocus) 496

junction Cookie、回避、ネーミング競合 448

Junction サーバー

- add 711

Junction されたバックエンド・サーバー図 19

junction スタンザ 60

junction スロットル 455, 568

- エラー・メッセージ・ページ 462
- オフライン操作状態 458
- オンライン操作状態 460
- 概念 455
- スロットル操作状態 456
- メッセージ 462
- WebSEAL 機能での使用 463

Junction の管理

- LMI 420

Junction フェアネス 59

junction マッピング・テーブル 491

junctions

- インポートおよびエクスポート 422
- 応答の中の URL の変更 478
- 大文字小文字を区別しない URL (-i) 467
- 概要 16, 417
- 仮想ホスト 547
- 仮想ホスト junction 547
- 仮想ホスト (-v) 470
- 仮想ホスト名 (-v) 672
- 技術上の注意点 430
- 基本 junction コマンド 422
- 基本 junction の構成 422
- 許可を実施する 432
- クライアント証明書 (WebSEAL) (-K) 445
- グローバル・サインオン (GSO) 600
- コマンド・オプションの参照 (junction) 531
- コマンド・オプション・リファレンス (仮想ホスト junction) 573
- 固有の Cookie 名前属性の確認 (-I) 504

junctions (続き)

- 作成ガイドライン 430
- 参照 (junction) 531
- 識別名 (DN) の突き合わせ (-D) 445
- 指定の Cookie 名の保存 503
- 証明書認証 432
- 新規 junction の強制適用 (-f) 453
- シングル・サインオン・ソリューション 473
- 推奨構成例 672
- スケラビリティ 18
- ステートフル 448
- ステートフル junction サポート (-s, -u) 449
- すべての Cookie 名の保存 (-n) 503
- 相互認証 (-D, -K, -B, -U, -W) 443
- 相互認証される junction に対する -b オプションの影響 595
- 透過パス Junction (-x) 427
- バックエンド UUID の指定 (-u) 450
- バックエンド・アプリケーションからの URL の変更 475
- フォーム・シングル・サインオン (-S) 615
- 複数の -j junction にわたる Cookie の処理 501
- プロキシ junction (-H, -P) 446
- ポータル・サーバーへのセッション Cookie の (-k) 466
- 要求の中の URL の変更 490
- 要求の中のサーバー相対 URL 処理のコントロール 499
- リファレンス (仮想ホスト junction) 573
- レプリカ・セットの割り当て (仮想ホスト junction) 366
- レプリカ・セットの割り当て (標準 junction) 366
- ローカル・タイプの junction の構成 426
- ワーカー・スレッド割り振り (-L) 61
- ワーカー・スレッド割り振り (-I) 61
- BA ヘッダーによる認証 (-B, -U, -W) 446
- Cookie を使用したサーバー相対 URL の変更 (-j) 493
- gso オプション (-b gso, -T) 602
- Host ヘッダーの推奨構成例 (-v) 672
- HTTP ヘッダーへの IP アドレスの挿入 (-r) 599
- HTTP ヘッダーへのクライアント識別の挿入 (-c) 596
- HTTP ヘッダー用のエンコード (-e) 471
- HTTP-Tag-Value junction 属性 252, 677

junctions (続き)

- HTTP/1.0 および 1.1 応答 433

- junction Cookie JavaScript のコントロール (-J) 495

- junction Cookie JavaScript の付加 (-J トレーラー) 495

- junction Cookie のリセット (-J onfocus) 496

- junction Cookie、回避、ネーミング競合 448

- Junction サーバーの追加 711

- junction スロットル 455, 568

- junction への複数のバックエンド・レプリカ・サーバーの追加 426, 431

- junction マッピングによるサーバー相対 URL の変更 491

- LTPA (-A, -F, -Z) 605

- mutual junction の構成 424

- pdadmin server task virtualhost 573

- pdadmin サーバー・タスク 421, 531
- process-root-requests スタンザ・エントリー 499

- query_contents 434

- query_contents スクリプト・ロケーション (-q) 438

- Referer ヘッダーを使用したサーバー相対 URL の変更 498

- SSL タイプ 425

- SSL タイプの junction の構成 423

- TCP タイプの junction の構成 423

- Web Portal Manager での作成 419

- Web Portal Manager でのリスト 420

- WebSEAL から WebSEAL (-C) 447

- WebSEAL クライアント証明書

- (-K) 445

- Windows ファイル・システム

- (-w) 468

- WPM の使用 419

- b filter 594

- b gso 595

- b ignore 593

- b supply 592

- j およびマッピング・テーブル

- junction の Cookie パスの変更 501

- j およびマッピング・テーブル

- junction の Cookie 名の変更 502

- junction-root ディレクトリー 85

K

Kerberos

- 資格情報 589

Kerberos 認証 174

- 構成 175

L

late-lockout-notification スタンザ・エントリー 242
lcp_bin 76
lcp_uri 76
LDAP サーバー
 z/OS 上 xxi
LDAP ディレクトリー・サーバー構成 57
 ポート 58
 bind-dn 58
 bind-pwd 58
 enabled 58
 host 58
 ldap.conf 57
 max-search-size 58
 replica 58
 ssl-port 58
ldap.conf (LDAP 構成) 57
libsslauthn (共用ライブラリー) 168
listen ポート 664
listen-flags スタンザ・エントリー 394
LMI
 拡張構成ファイル・エディター 38
 「クライアント証明書マッピング」管理ページ 221
 「照会サイトのコンテンツ」管理ページ 436
 フォーム・ベースのシングル・サインオン管理ページ 611
 リバース・プロキシ管理ページ 85
 「リバース・プロキシ管理ルートの管理」ページ 85
 リバース・プロキシの基本構成 38
 「リバース・プロキシ・トレースの管理」管理ページ 30
 「リバース・プロキシ・ログ・ファイルの管理」管理ページ 28
 「HTTP 変換ルール・ファイル」管理ページ 511
 junction 管理ページ 420
 LTPA 鍵管理ページ 404
 SSL 証明書管理ページ 404
 SSO キー管理ページ 404
 URL マッピング管理ページ 685
local-response-redirect-uri スタンザ・エントリー 105
Location HTTP ヘッダー
 リダイレクト・ロケーション・ヘッダーのフォーマット 102
logaudit スタンザ・エントリー 27
logcfg スタンザ・エントリー 26
logflush スタンザ・エントリー 27
logging スタンザ 27, 528

logging スタンザ (続き)
 server-log-cfg スタンザ・エントリー 528
login-form-action スタンザ・エントリー 611
login-page スタンザ・エントリー 611
login-page-stanza 611
login-redirect-page スタンザ・エントリー 230
login-success スタンザ・エントリー 94
login.html 79, 162
login_success.html 79
 ログイン・データの直接送付 162
 login_success 応答のコントロール 669
logout
 pkmslogout 158
 pkmslogout-nomas 654
 use-filename-for-pkmslogout スタンザ・エントリー 158
logout スタンザ・エントリー 94
logout-remove-cookie スタンザ・エントリー 335
logout.html 79
logsize スタンザ・エントリー 27
LTPA 631
 概要 604
 シングル・サインオン 632
 シングル・サインオンに関する技術上の注意点 632
 シングル・サインオンの概要 631
 トークンの存続時間の制御 180
 認証 75, 177
 認証 Cookie 用の UTF-8 エンコード 75
 認証の概要 177
 認証の使用可能化 178
 認証を使用不可にする 180
 Cookie 291
LTPA (WebSphere) 604
 junction の構成 605
 LTPA キャッシュの構成 606
LTPA 鍵
 LMI 管理ページ 404
LTPA キャッシュ、構成 606
LTPA シングル・サインオン
 技術上の注意点 607
ltpa-cache スタンザ 606
ltpa-cache-enabled スタンザ・エントリー 606
ltpa-cache-entry-idle-timeout スタンザ・エントリー 606
ltpa-cache-entry-lifetime スタンザ・エントリー 606
ltpa-cache-size スタンザ・エントリー 606

M

MAS (マスター認証サーバー) 636
master-authn-server スタンザ・エントリー 650
master-https-port スタンザ・エントリー 651
master-http-port スタンザ・エントリー 651
max-concurrent-web-sessions ポリシー 375
 ユーザー切り替えを使用 380
max-entries スタンザ・エントリー 295
max-login-failures ポリシー 239
max-password-repeated-chars ポリシー 244
max-search-size (LDAP 構成) 58
max-size スタンザ・エントリー 27
max-webseal-header-size スタンザ・エントリー 600
Microsoft Office 340, 341, 342
 セッション共用 339
Microsoft Office アプリケーション
 セッションの共用 339
MIME タイプ 27
min-password-alphas ポリシー 244
min-password-length ポリシー 244
min-password-non-alphas ポリシー 244
Mozilla Firefox
 favicon.ico 404 問題 100
MPA
 認証に関する制限 184
 ユーザー・アカウントを作成する 184
mpa スタンザ・エントリー 184
MPA 認証 181
msg_webseald-instance-name.log 26
mutual junction 424

N

NCSA 結合フォーマット 26
network-interface スタンザ・エントリー 558
nexttoken.html 79
NOTICE エラー・メッセージ 26
NOTICE_VERBOSE エラー・メッセージ 26

O

OAuth
 許可決定サポート 397
 directory 85
 EAS 397
OAuth EAS
 エラー応答 400
 概要 397
 プラグイン 382

oauth-eas スタンザ
 trace-component 401
obligations-levels-mapping スタンザ
 ランタイム・セキュリティ・サービ
 ス外部許可サービス (EAS) 140
OLDSESSION マクロ 334, 335
onfocus (junction Cookie のリセット) 496
online
 資料 xviii
 用語集 xviii
onLoad HTTP 属性
 ロード・バランシング環境内の 670

P

P3P
 構成のトラブルシューティング 137
p3p 126
 アクセス 131
 カスタム・コンパクト・ポリシー 137
 カテゴリ 131
 完全コンパクト・ポリシー 136
 構成 130
 デフォルト・ポリシー 129
 ヘッダーの保存 129
 ポリシー宣言 128
 ポリシーの概要 126
 目的 133
 disputes 132
 non-identifiable 133
 recipient 135
 remedies 133
 retention 136
passwd-change スタンザ・エントリー 94
passwd-change-failure スタンザ・エントリ
 ー 94
passwd-change-success スタンザ・エントリ
 ー 94
passwd-expired スタンザ・エントリー 94
passwd-warn スタンザ・エントリー 95
passwd-warn-failure スタンザ・エントリー
 95
passwd.html 79
passwd_exp.html 79
passwd_rep.html 79
passwd_warn.html 79
password-spaces ポリシー 244
pdadmin
 コマンド・リファレンス 707
 help 709
pdadmin サーバー・タスク
 オフライン 459
 作成 531, 532
 作成 (junction) 421
 表示 462, 532
 リスト 532
pdadmin サーバー・タスク (続き)
 add 532
 cache flush all 45
 delete 532
 dynurl update 689
 jdb インポート・ファイル 422
 jdb エクスポート・ファイル 422
 jmt clear 532
 jmt load 492
 offline 532
 online 460, 532
 refresh all_sessions 260
 remove 532
 terminate all_sessions 681
 terminate session 680
 throttle 457, 532
 virtualhost 552
 virtualhost add 574
 virtualhost create 573, 574
 virtualhost delete 574
 virtualhost list 574
 virtualhost offline 459, 574
 virtualhost online 461, 574
 virtualhost remove 574
 virtualhost show 462, 574
 virtualhost throttle 457, 574
pdadmin サーバー・リスト 36, 421, 531,
 574
pdadmin ポリシー
 グローバル設定 379
 コマンド 379
 ユーザーごとの設定 379
 disable-time-interval 241
 max-concurrent-web-sessions 375
 max-login-failures 240
 max-password-repeated-chars 244
 min-password-alphas 244
 min-password-length 244
 min-password-non-alphas 244
 password-spaces 244
pdaudit-filter スタンザ 26
pdaudit.conf 構成ファイル 26
pdweb.http.transformation 524
PD-H-SESSION-ID (デフォルトのセッション
 Cookie 名) 332
PD-S-SESSION-ID (デフォルトのセッション
 Cookie 名) 332
PD_PORTAL ヘッダー 674
persistent-con-timeout スタンザ・エントリ
 ー 50
ping-time (junction) 51
pkmscdsso 625
pkmsdisplace 377
pkmslogin.form 113, 114, 162
pkmslogout 158, 335, 377, 659
pkmslogout-nomas 654

pkmspasswd 159
pkmsvoucherfor 643, 651
Platform for Privacy Preferences 126
POP 513
 オブジェクトへの付加 210
 概念 10
 概要 9
 構成 527
 作成 206
 認証強度ポリシー (ステップアップ)
 200
 ネットワーク・ベースのアクセス制限
 207
 保護品質 393
 document-cache-control (拡張属性) 45
 IP Endpoint Authentication Method
 Policy 206
 reauth 拡張属性 195
port (LDAP 構成) 58
portal-map スタンザ 674
POST 方式
 制限の構成 690
 ログイン・データの直接送付 162
POST 要求 686
POST 要求のキャッシング 233
preserve-base-href スタンザ・エントリー
 483
preserve-base-href2 スタンザ・エントリー
 483
preserve-Cookie-names スタンザ 503
preserve-p3p-policy スタンザ・エントリー
 129
process-root-filter stanza 500
process-root-requests スタンザ・エントリ
 ー 499
prompt-for-displacement スタンザ・エント
 リー 377, 378
prompt_as_needed
 証明書認証 167
propagate-cdmf-errors スタンザ・エントリ
 ー 626, 652
putsuccess.html 79

Q

query_contents 434
 インストール 436
 概要 434
 カスタム・プログラム 435
 セキュア 440
 プロセス・フロー 440
query_contents スクリプト・ロケーシ
 ョン (-q) 438
UNIX でのインストールおよび構成
 437

query_contents (続き)

Windows でのインストールおよび構成
438

query_contents.c 436

query_contents.cfg 436

query_contents.exe 436

query_contents.html 79, 436

query_contents.sh 436

R

reauth POP 拡張属性 195

reauth-extend-lifetime スタンザ・エントリ
ー 197

reauth-for-inactive スタンザ・エントリ
ー 196

reauth-reset-lifetime スタンザ・エントリ
ー 197

redirect

ロケーション・ヘッダー URL フォー
マットの構成 102

redirect スタンザ・エントリ 94, 230,
277

redirect-using-relative スタンザ・エントリ
ー 102

Referer HTTP ヘッダー

サーバー相対 URL の変更 498

referers スタンザ・エントリ 27

referers-file スタンザ・エントリ 27

referer.log 27

register-authentication-failures スタンザ・エ
ントリ 385

reissue-missing-failover-cookie スタンザ・
エントリ 315

relocated.html 79

replica (LDAP 構成) 58

replica-sets スタンザ・エントリ 366

requests スタンザ・エントリ 27

requests-file スタンザ・エントリ 27

request-body-max-read スタンザ・エントリ
ー 236, 690

request-buffer-control

内部バッファのバイパス 472

request-max-cache スタンザ・エントリ
ー 236

request.log 27

require-mpa スタンザ・エントリ 338

resend-webseal-cookies スタンザ・エントリ
ー 333

ResponseLine/StatusCode のみ
(HTTPResponse)

変更 518

response-buffer-control

内部バッファのバイパス 472

response-by スタンザ・エントリ 371

rewrite-absolute-with-absolute スタンザ・エ
ントリ 488

rtss-eas スタンザ

サンプル構成 143

ランタイム・セキュリティ・サービ
ス外部許可サービス (EAS) 140

S

script-filter スタンザ・エントリ 487

script-filtering スタンザ 482, 487

Security Access Manager

概要 3

構成のガイドライン 703

トレース・メカニズム 401

マクロ 90

securitygroup 188

server list コマンド 710

server sync

command 22

server task コマンド

クラスターの再始動 716

add 711

cache flush all 714

server コマンド

server list 710

server task add 711

server task cache flush all 714

server task cluster restart 716

server スタンザ

allow-unsolicited-logins 123

server スタンザ・エントリ 364

server-log-cfg スタンザ・エントリ
(logging スタンザ) 528

server-name (server スタンザ) 35

server-name スタンザ・エントリ (server
スタンザ) 668

session-activity-timestamp スタンザ・エン
トリー 317

session-cookie-domains スタンザ 384, 566

session-http-headers スタンザ 329, 337

session-lifetime-timestamp スタンザ・エン
トリー 316

show-all-auth-prompts スタンザ・エントリ
ー 212

single-signoff-uri スタンザ・エントリ
ー 660

SMS 290, 350

概念 349

仮想ホストをレプリカ・セットに割り
当てる 366

構成 363

構成: SMS クラスターの指定 364

構成: SMS のロケーションの指定
364

SMS (続き)

構成: 使用可能化および使用不可化
363

構成: レプリカ・セット 365

サーバー・クラスター 351

最大並行セッション・ポリシー 375

事前割り振りセッション ID の最大数
372

セッション共用 381

セッション・レルム 351, 381

セッション・レルム内でのシングル・
サインオン 381

通信タイムアウトの構成 370

ドメイン Cookie 381

認証 372

[dsess-cluster] スタンザ 372

パフォーマンスの構成 372

ハンドル・プール・サイズ 372

標準 junction をレプリカ・セットに割
り当てる 366

フェイルオーバー環境 349

古いセッション Cookie に対するカス
タム・ログイン・メッセージ 333

ブロードキャスト・イベントの接続タ
イムアウト 371

プロセス・フロー 351

利点 350

レプリカ・セット 351

レプリカ・セット構成の例 367

ログイン失敗の通知 385

ログイン・ヒストリー 385

junction 359

SMS 応答タイムアウト 370

SSL 構成 373

SSL 証明書 DN 構成 374

ssl-session-cookie-name スタンザ・エン
トリー 383

WebSEAL junction の作成 386

WebSEAL 鍵データベース構成 373

WebSEAL の構成 363

WebSEAL の情報の収集 357

SMS セッション ID 270

SSL junction

相互認証 443

プロセスの要約 443

例 425

SSL 構成

WebSEAL から LDAP への 664

SSL 証明書

LMI 管理ページ 404

ssl スタンザ

suppress-client-ssl-errors スタンザ・エ
ントリー 528

SSL セッション ID 327

disable 171

SSL 接続 47

SSL タイプの junction 425
sslauthn (DLL) 168
ssl-id-sessions スタンザ・エントリー
171, 323, 327
ssl-keyfile スタンザ・エントリー
dsess-cluster スタンザ 373
ssl スタンザ 408
ssl-keyfile-label スタンザ・エントリー
dsess-cluster スタンザ 373
ssl スタンザ 408
ssl-keyfile-pwd スタンザ・エントリー
ssl スタンザ 408
ssl-keyfile-stash スタンザ・エントリー
dsess-cluster スタンザ 373
ssl スタンザ 408
ssl-listening-port スタンザ・エントリー
394
ssl-max-entries 172
ssl-max-entries スタンザ・エントリー
294
ssl-port (LDAP 構成) 58
ssl-qop-mgmt スタンザ・エントリー 394
ssl-qop-mgmt-default スタンザ 394
ssl-qop-mgmt-hosts スタンザ 396
ssl-qop-mgmt-networks スタンザ 396
ssl-session-cookie-name スタンザ・エント
リー 332, 383
ssl-v2-timeout スタンザ・エントリー
294, 330
ssl-v3-timeout スタンザ・エントリー
294, 330
ssl-valid-server-dn スタンザ・エントリー
374
SSO キー
LMI 管理ページ 404
standard-junction-replica-set スタンザ・エン
トリー 366
statistics ユーティリティ 30
stepuplogin.html 79, 205
stepup-login スタンザ・エントリー 94,
205
step-up-at-higher-level スタンザ・エントリ
ー 212
Suite B 暗号 119
suppress-backend-server-identity スタンザ・
エントリー 125
suppress-client-ssl-errors スタンザ・エン
トリー (ssl スタンザ) 528
suppress-server-identity スタンザ・エン
トリー 124
su-admin 拡張属性 193
su-admins グループ 185, 188
su-excluded グループ 188
switchuser.html 79
switch-user スタンザ・エントリー 94

system-environment-variables スタンザ
env-name スタンザ・エントリー 78

T

tagvalue_failover_amweb_session_id スタン
ザ・エントリー 323, 324
tagvalue_user_session_id 資格情報属性
677
tag-value
サポート 193
資格情報属性 250
HTTP-Tag-Value junction 属性 252,
677
tagvalue_user_session_id credential
attribute 677
TAM_OP、ローカル応答リダイレクト
106
tcp-session-cookie-name スタンザ・エン
トリー 332
template.html 79
TFIM (Federated Identity Manager) サポー
ト
セッション・キャッシュ・エントリー
存続時間 (ユーザーごとの設定)
279, 297
セッション・キャッシュ・エントリー
非アクティブ (ユーザーごとの設定)
281
HTTP ヘッダーを使用したセッション
状態の維持 336
timeout スタンザ・エントリー 370
session スタンザ 196, 279, 296, 297,
330
Tivoli Common Directory 26
Tivoli Directory Integrator xxi
Tivoli Directory Server xxi
Tivoli メッセージ・フォーマット 26
TLS 接続 47
tokenlogin.html 79
too-many-sessions スタンザ・エントリー
94
too_many_sessions.html 79
trace ユーティリティ 30
Transport Layer Security (TLS) 47
trigger スタンザ・エントリー 269

U

UMI
コンテナおよびコンテナ名 215
XML 文書モデル 214
UNIX
構成のテスト 438
unknownicon 40

uri シングル・サインオフ
WebSEAL の構成 660

URL

エンコードまたはエスケープされたも
のの変更 481
応答の中の URL の変更 478
サーバー相対パスについて 476
絶対パスについて 476
相対パスについて 476
単一エンコード 70
特殊な HTML 文字 477
パス・タイプ 476
バックエンド・リソースへの URL の
変更 475
複数の -j junction にわたる Cookie の
処理 501
要求の中の URL の変更 490
要求の中のサーバー相対 URL 処理の
コントロール 499
junction Cookie を使用したサーバー相
対 URL の変更 493
junction マッピングによるサーバー相
対 URL の変更 491
process-root-requests スタンザ・エン
トリー 499
Referer ヘッダーを使用したサーバー相
対 URL の変更 498
USERNAME マクロ
再認証ログイン・フォーム用の 200
user-session-ids スタンザ・エントリー
259, 676
user-session-ids-include-replica-set スタン
ザ・エントリー 679
use-filename-for-pkmslogout スタンザ・エン
トリー 158
use-new-stateful-on-error スタンザ・エン
トリー 453
use-same-session スタンザ・エントリー
328
Netscape 4.7x の制限 330
use-utf8
cdsso スタンザ 74
e-community-sso スタンザ 74
failover スタンザ 75
use-utf8 スタンザ・エントリー
cdsso スタンザ 629
e-community-sso スタンザ 657
utf8-form-support-enabled 72
utf8-form-support-enabled スタンザ・エン
トリー 77
utf8-qstring-support-enabled 73
utf8-qstring-support-enabled スタンザ・エン
トリー 77
utf8-template-macros-enabled スタンザ・エン
トリー 89
utf8-url-support-enabled 70

utf8_bin 76
utf8_uri 76
UTF-8
 認可に与える 影響 (動的 URL) 69
 ユーザー・レジストリー構成への依存
 関係 67
 WebSEAL データ処理 66
utf-8 65
 概要 65
 データ変換 67
 認証の影響 68
 ログ・データ 27
 cdsso 629
 e-community 認証 657
 HTML マクロ 89
 junction を介した HTTP ヘッダーのエ
 ンコード 471
 junctions 75
 UTF-8 形式のメッセージ 26
UTF-8 でのマルチロケール・サポート
65

V

verify-step-up-user 211
vf-argument スタンザ・エントリー 649
vf-token-lifetime スタンザ・エントリー
652
vf-url スタンザ・エントリー 651
vouch-for トークン
 disable-ec-cookie 653
vouch-for 要求と応答 643

W

WARNING エラー・メッセージ 26
Web Portal Manager 11
 Junction の管理 419
 junction の作成 419
 junction のリスト 420
Web オブジェクト 8
Web サーバー構成 35
Web サーバー・セキュリティ 119
Web スペースの構造 693
WebDAV 52
WebSEAL
 アプライアンスでの機能 6
 インスタンス 663
 インスタンスの構成タスク 666
 インスタンス名の指定 35
 応答ヘッダー 42
 外部認証インターフェースの使用 276
 概要 3, 4
 鍵データベース・ファイル 406
 サーバー名表示 407

WebSEAL (続き)
 鍵データベース・ファイル・パスワー
 ド 406
 仮想ホストの使用 564
 きめの細かいアクセス・コントロール
 418
 クライアントからの要求ヘッダー 43
 クラスター・サポート 25
 同期 23
 構成 366
 構成設定 358
 構成データ・ログ・ファイル 28
 構成のテスト 360
 構成の要約 691
 構成ファイル 37, 512
 コンテンツ・キャッシュ 43, 44
 サーバー応答 79
 サーバーおよびホスト名指定 35
 サーバー識別の抑止 124
 証明書マッピング・モジュールの使用
 223
 制限 231
 セッションの保管 290
 セッション・キャッシュの制限 301
 対応策
 絶対 URL リンクおよびサーバー相
 対 URL リンクの回避 490
 大まかなアクセス・コントロール 418
 データの同期化 22
 停止と始動 191
 デフォルトの構成ファイル 139, 143
 aznapi-external-authzn-services スタ
 ンザ 140
 azn-decision-info スタンザ 140
 obligations-levels-mapping スタンザ
 140
 rtss-eas スタンザ 140
 デフォルトの証明書鍵データベース
 pdsrv.kdb 406
 内部バッファのバイパス 472
 認証情報のマッピング 602
 パスワード処理 239
 標準 junction 427
 フィルター操作の概念 427
 フェイルオーバー Cookie の使用 324
 複数のレプリカ・セット 366
 複製されたサーバー 242
 ポリシー適用ポイント (PEP)
 ランタイム・セキュリティ・サー
 ビス外部許可サービス
 (EAS) 139
 問題
 未処理要求 489
 ランタイム・セキュリティ・サービ
 ス外部許可サービス (EAS) 139,
 140, 143

WebSEAL (続き)
 リスクベース・アクセス 139
 ログイン失敗ポリシー 242
 ACL 418
 ACL および POP の比較 470
 ACL 許可 392
 Cookie の管理 464
 host-instance_name 391
 HTTP 要求の構成 46
 HTTPS 要求の構成 47
 HTTP/1.1 応答 433
 ICAP 構成 699
 ICAP の統合 697, 698
 IPv6 および IPv4 のサポート 54
 junction Cookie サポート構成 494
 OAuth 397
 OAuth 決定 399
 oauth-eas スタンザ 400
 oauth_eas.conf.template 400
 query_contents 418
 RPC over HTTP のサポート 525
 server restart 359
 server sync コマンド 22
 server-name スタンザ・エントリー 35
 SMS 357
 SMS を使用する
 構成 357
 SSL 接続用の GSKit 属性 589
 statistics ユーティリティ 30
 trace ユーティリティ 30
 UTF-8 を使用した データ処理 66
 webseald-default.conf 37
 web-host-name スタンザ・エントリー
 37
 -c junction 598
WebSEAL junction
 他の参照情報 419
 バックエンド・サーバー 19
 GSO 使用可能例 603
 SMS 359
 SMS での作成 386
WebSEAL junction、junction を参照 417
WebSEAL インスタンス
 デプロイメント 663
WebSEAL サーバー・ログのエラー 528
WebSEAL セッション ID 149, 286
WebSEAL セッション・キャッシュ
 概要と構造 287
 構成 295
 最大エントリー数 295
 存続時間タイムアウト (グローバル設
 定) 296
 存続時間タイムアウト (ユーザーごと
 の設定) 279, 297
 非アクティブ・タイムアウト 299

WebSEAL セッション・キャッシュ (続き)
非アクティブ・タイムアウト (ユーザーごとの設定) 281
並行セッションの制限 301
WebSEAL との ICAP の統合 697, 698
WebSEAL 内の ICAP 構成 699
WebSEAL の Session Management Server (SMS) 構成 363
WebSEAL の構成
概要 663
各インスタンス用の 666
計画および設定 663
構成値の例 665
WebSEAL インスタンスの除去 667
WebSEAL インスタンスを追加する 666
webseald-default.conf 37
WebSEAL 要求ログ
イベント相関の挿入 683
WebSEAL レプリカ・サーバー
整合性のとれた構成 288
webseald-default.conf 37
websealerror.html 79
WebSEAL、インスタンス
新規追加 666
remove 667
webseal-cert-keyfile スタンザ 406
webseal-cert-keyfile スタンザ・エンタリー 405
webseal-cert-keyfile-label 412
webseal-cert-keyfile-label スタンザ・エンタリー 405, 432, 558
webseal-cert-keyfile-pwd スタンザ
コメント化
pdsrv.kdb 406
webseal-cert-keyfile-pwd スタンザ・エンタリー 405
webseal-cert-keyfile-sni スタンザ・エンタリー 405, 407
webseal-cert-keyfile-stash スタンザ・エンタリー 405
webseal-mpa-servers グループ 183, 184
WebSEAL/host-instance_name 391
WebSphere Application Server Network Deployment xxi
WebSphere eXtreme Scale xxi
WebSphere LTPA 604
Web-based Distributed Authoring and Versioning 52
web-host-name スタンザ・エンタリー 37, 563, 567
Windows
構成のテスト 439
例 469
後書き拡張子ドットを含める 469

Windows (続き)
例 (続き)
大文字小文字を区別しないファイル名を許可する 469
別名の作成 469
Windows デスクトップ・シングル・サインオン 174
構成ステップ 175
worker-threads スタンザ・エンタリー 59, 558
worker-thread-hard-limit スタンザ・エンタリー 60
worker-thread-soft-limit スタンザ・エンタリー 60
WPM
junction アプリケーション 434
junction の管理 419
junction の削除 420

X

XHTML 1.0 準拠 JavaScript ブロック (xhtml10)
挿入 497
XML
証明書の例 216
ユーザー・マッピング・ルール 214
UMI コンテナおよびコンテナ名 215
UMI 文書モデル 214
XSL
ユーザー・マッピング・ルール 214
xsl:template ステートメント 218
XSL Transformation ルール 510
XSLT テンプレート 511
XSLT ルール・ファイル
構成 224
xsl:template ステートメント 218
xsl:when ステートメント 220



Printed in Japan

SA88-5104-01



日本アイ・ビー・エム株式会社

〒103-8510 東京都中央区日本橋箱崎町19-21