

IBM Common Data Provider for z Systems
Version 1 Release 1

User Guide



Figures

1. Flow of operational data among IBM Common Data Provider for z Systems components to multiple analytics platforms..... 2

Tables

1. Required authorizations and associated information for each component.....	6
2. Working directories for IBM Common Data Provider for z Systems components.....	7
3. Data gatherer configuration of Data Streamer port number.....	8
4. Target libraries for IBM Common Data Provider for z Systems components.....	9
5. Protocol change rules.....	30
6. SMF record type 127 subtype 1000 layout.....	48
7. Common target destinations with the required streaming protocols and associated information.....	59
8. Mapping of the prefix that is used in a Logstash configuration file name to the content of the file.....	64
9. User exits for collecting z/OS SYSLOG data, with associated MVS installation exits and usage notes....	75
10. Example System Data Engine interval values that are a factor of the total time in one day.....	81
11. Configuration reference information for managing policies.....	92
12. Subgroup and data streams of Starter Sets.....	97
13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data.....	98
14. Fields in the SMF_110_1_KPI data stream.....	112
15. Icons on each data stream node in a policy.....	115
16. Icons on each transform node in a policy.....	115
17. Icons on each subscriber node in a policy.....	116
18. Correlation between the sources from which the Log Forwarder gathers data and the data streams that can be defined for those sources.....	117
19. Logical operation NOT.....	155
20. Logical operations AND and OR.....	155
21. Field formats for the FIELDS clause of the DEFINE RECORD statement.....	165
22. z/OS console commands for starting, stopping, or viewing status or configuration information for individual Log Forwarder data streams.....	180

23. Headers for data that is sent by using the Data Transfer Protocol.....	185
24. Unsplit payload format.....	186
25. Split payload format.....	187
26. Metadata keywords and values.....	188
27. IBM Tivoli Decision Support for z/OS lookup table members to customize.....	197
28. Sample jobs for adding tables to IBM Db2 Analytics Accelerator for z/OS.....	197
29. Sample jobs for moving lookup table contents to IBM Db2 Analytics Accelerator for z/OS.....	197
30. IBM Common Data Provider for z Systems lookup table members.....	198
31. Sample jobs for generating Db2 UNLOAD format.....	198
32. Sample jobs for loading data into IBM Db2 Analytics Accelerator.....	200
33. Sample jobs for enabling tables for acceleration in IBM Db2 Analytics Accelerator.....	200
34. Sample jobs that are provided by IBM Tivoli Decision Support for z/OS for removing tables from IBM Db2 Analytics Accelerator for z/OS.....	200
35. IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System Data Engine.....	201
36. Tables for Analytics - z/OS Performance component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables.....	202
37. Tables for Analytics - Db2 component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables.....	204
38. Tables for Analytics - KPM CICS component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables.....	205
39. Tables for Analytics - KPM Db2 component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables.....	205
40. Tables for Analytics - KPM z/OS component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables.....	206
41. IBM Tivoli Decision Support for z/OS analytics component views that are based on multiple tables.....	207

Contents

Figures.....	iii
Tables.....	v
Chapter 1. Overview.....	1
Operational data.....	2
Analytics platforms.....	2
Components of Common Data Provider for z Systems.....	3
Chapter 2. Planning.....	5
z/OS system requirements.....	5
Data Receiver system requirements.....	6
Required authorizations for Common Data Provider components.....	6
Working directory definitions.....	7
Data Streamer port definition.....	7
Chapter 3. Installing.....	9
Chapter 4. Configuring.....	11
Getting started with the Configuration Tool.....	11
Getting started with the Configuration Tool on Liberty.....	11
Getting started with the Configuration Tool on z/OSMF.....	18
Output from the Configuration Tool.....	20
Managing policies.....	21
Purpose of transforming data in a policy.....	21
Subscribers to a data stream or transform.....	22
Creating a policy.....	24
Updating a policy.....	30
Migrating a policy.....	31
Adding a subscriber for a data stream or transform.....	31
Updating subscriptions of a subscriber.....	32
Exporting and importing subscribers.....	32
Managing custom data streams.....	33
Creating a System Data Engine data stream definition.....	33
Creating an application data stream definition.....	54
Deleting a custom data stream definition.....	55
Securing communications between the Data Streamer and its subscribers.....	56
Preparing the Common Data Provider and the target destinations to stream and receive data.....	59
Preparing to send data to Splunk.....	61
Preparing to send data to Elasticsearch.....	64
Configuring the Data Receiver.....	65
Configuring a Logstash receiver.....	67
Configuring the Data Streamer.....	68
File buffer function in the Data Streamer.....	70
Specifying the timestamp format information for streaming custom data types to Splunk via HEC.....	70
Configuring the data gatherer components.....	71
Configuring the Log Forwarder.....	72
Configuring the System Data Engine.....	78

Verifying the search order for the TCP/IP resolver configuration file.....	91
Configuration reference for managing policies.....	91
Global properties that you can define for all data streams in a policy.....	92
Groups of data streams in the Configuration Tool.....	96
SMF data stream reference.....	97
SMF_110_1_KPI data stream content.....	111
Icons on each node in a policy.....	115
Data stream configuration for data gathered by Log Forwarder.....	116
Data stream configuration for data gathered by System Data Engine.....	143
Transform configuration.....	143
Subscriber configuration.....	147
Language reference for System Data Engine.....	150
Language overview.....	150
DEFINE RECORD statement.....	161
DEFINE UPDATE statement	169
DEFINE TEMPLATE statement.....	174
Chapter 5. Operating.....	177
Running the Data Receiver.....	177
Running the Data Streamer.....	178
Running the Log Forwarder.....	179
Running the System Data Engine.....	183
Chapter 6. Sending user application data to the Data Streamer.....	185
Data Transfer Protocol.....	185
Sending data by using the Java API.....	188
Sending data by using the REXX API.....	190
Chapter 7. Loading data to IBM Db2 Analytics Accelerator.....	193
Configuring IBM Tivoli Decision Support for z/OS for loading the data.....	193
Running the System Data Engine to write data in Db2 UNLOAD format.....	198
Loading data to IBM Db2 Analytics Accelerator.....	199
Removing tables from IBM Db2 Analytics Accelerator.....	200
IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System	
Data Engine.....	201
Analytics component tables.....	202
Analytics component views that are based on multiple tables.....	206
Chapter 8. Troubleshooting.....	209
Log Forwarder log files.....	209
Log Forwarder: enabling tracing.....	209
Enabling static tracing for the Log Forwarder.....	209
Enabling dynamic tracing for the Log Forwarder.....	210
System Data Engine log files.....	211
System Data Engine: enabling tracing.....	211
Enabling tracing for the System Data Engine at startup.....	211
Enabling tracing for the System Data Engine after startup.....	212
Log Forwarder user ID has insufficient authority.....	213
BPX messages precede GLA messages in the z/OS SYSLOG.....	213
Log Forwarder message states that PPI issued return code 24.....	214
NetView message provider GLANETV issues message GLAL004E with return code 15.....	214
NetView message provider GLANETV issues message GLAL006E.....	214
Data Streamer does not start.....	215
System Data Engine does not start.....	215
Data Streamer is not receiving data from System Data Engine.....	216
Subscriber is not receiving data.....	216
syslogd message problems: inconsistencies in time stamps, or missing or misplaced messages.....	217

User ID of parameter AUTHORIZED_USER is not found.....	217
Notices.....	219
Trademarks.....	220
Terms and conditions for product documentation.....	220

Chapter 1. Common Data Provider for z Systems overview

IBM® Common Data Provider for z Systems® provides the infrastructure for accessing IT operational data from z/OS® systems and streaming it to the analytics platform in a consumable format. It is a single data provider for sources of both structured and unstructured data, and it can provide a near real-time data feed of z/OS log data and System Management Facilities (SMF) data to your analytics platform.

IBM Common Data Provider for z Systems automatically monitors z/OS log data and SMF data and forwards it to the configured destination.

In each logical partition (LPAR) from which you want to analyze z/OS log data or SMF data, a unique instance of IBM Common Data Provider for z Systems must be installed and configured to specify the type of data to be gathered and the destination for that data, which is called a *subscriber*.

IBM Common Data Provider for z Systems includes a web-based configuration tool that is provided as an application for IBM WebSphere® Application Server for z/OS Liberty, or as a plug-in for IBM z/OS Management Facility (z/OSMF).

Flow of operational data to your analytics platform

As illustrated in Figure 1 on page 2, operational data (such as SMF record type 30 or z/OS SYSLOG data) is gathered by data gatherers, such as the System Data Engine or the Log Forwarder, and can be streamed to multiple subscribers.

The data gatherers send the data to the Data Streamer, which transforms the data before it sends the data to the subscribers.

The flow of data is controlled by a policy that you define in the IBM Common Data Provider for z Systems Configuration Tool.

policy

In IBM Common Data Provider for z Systems, a set of rules that define what operational data to collect and where to send that data.

subscriber

In the IBM Common Data Provider for z Systems configuration, the software that you define to receive operational data. You can have both on-platform and off-platform subscribers.

on-platform subscriber

A subscriber that is on the same z/OS system, or in the same logical partition (LPAR), as the source from which the operational data originates.

off-platform subscriber

A subscriber that is **not** on the same z/OS system, or in the same logical partition (LPAR), as the source from which the operational data originates.

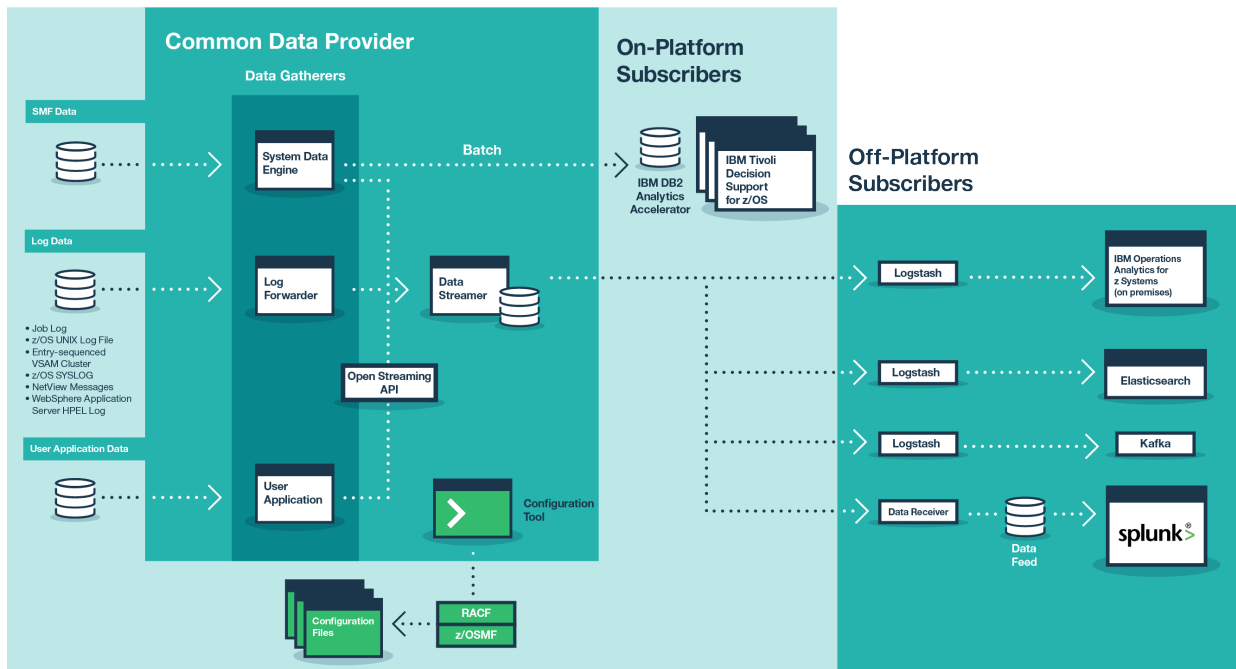


Figure 1. Flow of operational data among IBM Common Data Provider for z Systems components to multiple analytics platforms

Operational data

Operational data is data that is generated by the z/OS system as it runs. This data describes the health of the system and the actions that are taking place on the system. The analysis of operational data by analytics platforms and cognitive agents can produce insights and recommended actions for making the system work more efficiently and for resolving, or preventing, problems.

IBM Common Data Provider for z Systems can collect the following types of operational data:

- System Management Facilities (SMF) data
- Log data from the following sources:
 - Job log, which is output that is written to a data definition (DD) by a running job
 - z/OS UNIX log file, including the UNIX System Services system log (syslogd)
 - Entry-sequenced Virtual Storage Access Method (VSAM) cluster
 - z/OS system log (SYSLOG)
 - IBM Tivoli® NetView® for z/OS messages
 - IBM WebSphere Application Server for z/OS High Performance Extensible Logging (HPEL) log
 - IBM Information Management System (IMS) log
- User application data, which is operational data from your own applications

Analytics platforms

An analytics platform is a software program, or group of dedicated systems and software, that is configured to receive, store, and analyze large volumes of operational data.

The following analytics platforms are examples:

- IBM Db2® Analytics Accelerator for z/OS, a database application that provides query-based reporting

- IBM Z Operations Analytics, an on-premises product that can receive large volumes of operational data for analysis and can provide insights and recommended actions to the system owners, which are based on expert knowledge about z Systems and applications
- Platforms such as Elasticsearch, Apache Hadoop, and Splunk that can receive and store operational data for analysis. These platforms do not include expert knowledge about z Systems and applications, but users can create or import their own analytics to run against the data.

Tip: Apache Kafka is an open-source, high performance, distributed streaming platform that can be used to publish and subscribe to streams of records. It can be used as a path to data consumers such as Apache Hadoop, Apache Spark Streaming, and others.

Components of Common Data Provider for z Systems

IBM Common Data Provider for z Systems includes the following basic components: 1) a Configuration Tool for defining the sources from which you want to collect operational data, 2) the data gatherer components (System Data Engine and Log Forwarder) for gathering different types of operational data, and 3) a Data Streamer for streaming all data to its destination.

Other components include the Open Streaming API for gathering operational data from your own applications, and a Data Receiver that acts as a target subscriber for operational data if the intended subscriber cannot directly ingest the data feed.

The components are illustrated in [Figure 1 on page 2](#).

Basic components

Configuration Tool

The IBM Common Data Provider for z Systems Configuration Tool is a web-based user interface that is provided as an application for IBM WebSphere Application Server for z/OS Liberty, or as a plug-in for IBM z/OS Management Facility (z/OSMF). In the tool, you specify the configuration information as part of creating a *policy* for streaming operational data to its destination.

In the policy definition, you must define a *data stream* for each source from which you want to collect operational data. A stream of data is a set of data that is sent from a common source in a standard format, is routed to, and transformed by, the Data Streamer in a predictable way, and is delivered to one or more subscribers.

You must specify the following information for each data stream in the policy:

- The source (such as SMF record type 30 or z/OS SYSLOG)
- The format to which to transform the operational data so that it is consumable by the analytics platform.

Note: The Splitter transforms are available only for Generic z/OS Job Output data stream, Generic ZFS File data stream, Generic VSAM Cluster data stream, and zSecure Access Monitor.

- The subscriber or subscribers for the operational data that is output by IBM Common Data Provider for z Systems.

For example, subscribers include Logstash, the Data Receiver, the HTTP Event Collector (HEC) of Splunk, and a generic HTTP receiver.

Data gatherer components

Each of the following components gathers a different type of data:

System Data Engine

The System Data Engine gathers System Management Facilities (SMF) data and IBM Information Management System (IMS) log data in near real time. It can also gather SMF data and IMS data in batch.

The System Data Engine can process all commonly used SMF record types from the following sources:

- SMF archive (which is processed only in batch)
- SMF in-memory resource (by using the SMF real-time interface)
- SMF user exit HBOSMFEX
- SMF log stream

It can also convert SMF records into a consumable format, such as a comma-separated values (CSV) file, or into Db2 UNLOAD format for loading in batch.

The System Data Engine can also be installed as a stand-alone utility to feed SMF data into IBM Db2 Analytics Accelerator for z/OS (IDAA) for use by IBM Tivoli Decision Support for z/OS.

Log Forwarder

The Log Forwarder gathers z/OS log data from the following sources:

- Job log, which is output that is written to a data definition (DD) by a running job
- z/OS UNIX log file, including the UNIX System Services system log (syslogd)
- Entry-sequenced Virtual Storage Access Method (VSAM) cluster
- z/OS system log (SYSLOG)
- IBM Tivoli NetView for z/OS messages
- IBM WebSphere Application Server for z/OS High Performance Extensible Logging (HPEL) log
- IBM Information Management System (IMS) log

To reduce general CPU usage and costs, you can run the Log Forwarder on z Systems Integrated Information Processors (zIIPs).

Data Streamer

The Data Streamer streams operational data to configured subscribers in the appropriate format. It receives the data from the data gatherers, alters the data to make it consumable for the subscriber, and sends the data to the subscriber. In altering the data to make it consumable, the Data Streamer can, for example, split the data into individual messages, or translate the data into a different encoding (such as from EBCDIC encoding to UTF-8 encoding).

The Data Streamer can stream data to both on-platform and off-platform subscribers. To reduce general CPU usage and costs, you can run the Data Streamer on z Systems Integrated Information Processors (zIIPs).

Other components

Depending on your environment, you might want also want to use one, or both, of the following components:

Open Streaming API

The Open Streaming API provides an efficient way to gather operational data from your own applications by enabling your applications to be data gatherers. You can use the API to send your application data to the Data Streamer and stream it to analytics platforms.

Data Receiver

The Data Receiver is required only if the intended subscriber of a data stream cannot directly ingest the data feed from IBM Common Data Provider for z Systems. The Data Receiver writes any data that it receives to disk files, which can then be ingested into an analytics platform such as Splunk.

The Data Receiver typically runs on the same system as the analytics platform that processes the disk files. This system can be a distributed platform, or a z/OS system. For ingesting data to Splunk, install the Data Receiver on each Splunk forwarder that forwards data from IBM Common Data Provider for z Systems.

Chapter 2. Planning to use Common Data Provider for z Systems

Review the system and security requirements for using IBM Common Data Provider for z Systems to provide z/OS operational data. Also, review the information about the Data Streamer port definition and about the working directories for IBM Common Data Provider for z Systems components.

z/OS system requirements

Verify that your z/OS system meets the requirements for running IBM Common Data Provider for z Systems. You must run the IBM Common Data Provider for z Systems in each z/OS logical partition (LPAR) from which you want to gather z/OS operational data.

These requirements apply to the z/OS system where you are running the IBM Common Data Provider for z Systems Data Streamer, Log Forwarder, and System Data Engine.

Basic requirements

IBM Common Data Provider for z Systems must be run with the following software:

- IBM z/OS V2.1, V2.2, or V2.3 (product number 5655-ZOS)

For z/OS V2.1 or V2.2 only: If you use IBM z/OS V2.1 or V2.2, the following software is also required on the system where configuration is done:

For z/OS V2.1

IBM z/OS Management Facility V2.1 (product number 5610-A01), with APAR/PTF PI52426/UI36314

For z/OS V2.2

IBM z/OS Management Facility V2.2 (product number 5650-ZOS), with APAR/PTF PI52426/UI36315

- One of the following Java™ libraries:
 - IBM 31-bit SDK for z/OS Java Technology Edition V7 (product number 5655-W43)
 - IBM 64-bit SDK for z/OS Java Technology Edition V7 (product number 5655-W44)
 - IBM 31-bit SDK for z/OS Java Technology Edition V8 (product number 5655-DGG)
 - IBM 64-bit SDK for z/OS Java Technology Edition V8 (product number 5655-DGH)

Important considerations:

- Use the latest available service release of the version of IBM SDK for z/OS, Java Technology Edition, that you choose, and apply fix packs as soon as possible after they are released. To find the latest service release or fix pack, see [IBM Java Standard Edition Products on z/OS](#).
- Although the IBM Common Data Provider for z Systems runs on IBM SDK for z/OS, Java Technology Edition, Versions, 7 and 8, some tests indicate that CPU usage might be higher when you run the IBM Common Data Provider for z Systems on Version 7. If CPU usage of the IBM Common Data Provider for z Systems is a concern, consider running the IBM Common Data Provider for z Systems on IBM SDK for z/OS, Java Technology Edition Version 8.

Optional requirements

Depending on your environment, you might also want to run the following software with IBM Common Data Provider for z Systems:

- On an IBM z/OS V2.1 or V2.2 system, to collect System Management Facilities (SMF) data from SMF in-memory resources, you must apply APAR OA49263.

- To load data to IBM Db2 Analytics Accelerator for z/OS, you must run IBM Db2 Analytics Accelerator Loader for z/OS V2.1 (product number 5639-OLE).

Data Receiver system requirements

If you plan to use the IBM Common Data Provider for z Systems Data Receiver, verify that the system on which you plan to install the Data Receiver meets the requirements for running the Data Receiver.

The Data Receiver can be run on a Linux, Windows, or z/OS system. It requires Java Runtime Environment (JRE) 8.

If your target destination is Splunk, also see the [Splunk system requirements](#).

CPU usage, disk usage, and memory usage vary depending on the amount of data that is processed.

Required authorizations for Common Data Provider components

Various authorizations are required for installing and configuring IBM Common Data Provider for z Systems components and for accessing component-related libraries and configuration files during run time.

Table 1 on page 6 references the information about the required authorizations for each IBM Common Data Provider for z Systems component. The authorization requirements for installation of the components are described in the Program Directories.

Table 1. Required authorizations and associated information for each component	
Component	Information about required authorizations
Configuration Tool	<p>If your Configuration Tool is on Liberty:</p> <ul style="list-style-type: none"> • User IDs and group IDs for running the Configuration Tool: “Configuring user IDs, group IDs, and security product” on page 11 • User ID for running the setup script: “Setting up a Liberty server directory and a working directory for the Configuration Tool” on page 15 <p>If your Configuration Tool is on z/OSMF:</p> <ul style="list-style-type: none"> • User ID for running the setup script: “Setting up a working directory for the Configuration Tool” on page 18 • User ID for installing the tool: “Installing the Configuration Tool” on page 18 • User ID for uninstalling the tool: “Uninstalling the Configuration Tool” on page 20 • User ID for running the tool: “Running the Configuration Tool” on page 19
Data Streamer	<ul style="list-style-type: none"> • User ID that is associated with the Data Streamer started task: “Configuring the Data Streamer” on page 68

Table 1. Required authorizations and associated information for each component (continued)	
Component	Information about required authorizations
Log Forwarder	<ul style="list-style-type: none"> • User ID that is associated with the Log Forwarder started task: “Requirements for the Log Forwarder user ID” on page 73 • Security software updates to permit the Log Forwarder started task to run in your environment: “Creating the Log Forwarder started task” on page 72, step “4” on page 73
System Data Engine	<ul style="list-style-type: none"> • APF authorization: “Authorizing the System Data Engine with APF” on page 80 • User ID that is associated with the System Data Engine started task: “Requirements for the System Data Engine user ID” on page 82 • User ID for collecting IMS records: “Requirements for the System Data Engine user ID for collecting IMS records” on page 87

Working directory definitions

When you configure some IBM Common Data Provider for z Systems components, you must define a working directory for the component. To avoid possible conflicts, do not define the same directory as the working directory for multiple components.

Table 2 on page 7 indicates where you can find information about the working directories that must be defined.

Table 2. Working directories for IBM Common Data Provider for z Systems components	
Component	Information about working directory
Configuration Tool	“Setting up a working directory for the Configuration Tool” on page 18
Data Receiver	“Setting up a working directory and an output directory for the Data Receiver” on page 65
Data Streamer	“Configuring the Data Streamer” on page 68
Log Forwarder	“Log Forwarder properties configuration” on page 93

Data Streamer port definition

When you configure the IBM Common Data Provider for z Systems Data Streamer, you define the port number on which the Data Streamer listens for data from the data gatherers. All data gatherers must send data to the Data Streamer through this port. If you update this port in the Data Streamer configuration, you must also update it in the configuration for all data gatherers.

For information about how to update this port for the Data Streamer, see [“Configuring the Data Streamer” on page 68](#).

For each data gatherer, Table 3 on page 8 indicates where you can find information about the Data Streamer port number configuration.

<i>Table 3. Data gatherer configuration of Data Streamer port number</i>	
Data gatherer	Information about Data Streamer port number configuration
Log Forwarder	“Log Forwarder properties configuration” on page 93
System Data Engine	“Creating the System Data Engine started task for streaming SMF data” on page 80

Chapter 3. Installing Common Data Provider for z Systems

Install IBM Common Data Provider for z Systems by using SMP/E for z/OS (SMP/E). For installation instructions, see the two IBM Common Data Provider for z Systems Program Directories.

About this task

If you want to stream z/OS operational data to off-platform subscribers, such as those that are described and illustrated in Chapter 1, “Common Data Provider for z Systems overview,” on page 1, you must install all IBM Common Data Provider for z Systems components, according to the instructions in the following Program Directories:

- [Common Data Handler Program Directory](#)
- [System Data Engine Program Directory](#)

Tip: The term *Common Data Handler* is a generic concept that covers the following IBM Common Data Provider for z Systems components:

- Configuration Tool
- Data Streamer
- Log Forwarder

If you plan only to load data in batch mode to, for example, an on-platform subscriber such as IBM Db2 Analytics Accelerator for z/OS, for use by IBM Tivoli Decision Support for z/OS, you need to install only the System Data Engine, according to the instructions in the [System Data Engine Program Directory](#).

Table 4 on page 9 lists the target libraries.

Table 4. Target libraries for IBM Common Data Provider for z Systems components	
Component	Target library
Configuration Tool	• /usr/lpp/IBM/cdpz/v1r1m0/UI/LIB
Data Streamer	• /usr/lpp/IBM/cdpz/v1r1m0/DS/LIB
Log Forwarder	• /usr/lpp/IBM/zscala/V3R1
System Data Engine	For the following libraries, customize the high-level qualifier (.hlq) according to site requirements. <ul style="list-style-type: none">• hlq.SHBOCNTL• hlq.SHBOLOAD• hlq.SHBODEFS• hlq.SHBOSAMP

Chapter 4. Configuring Common Data Provider for z Systems

To configure IBM Common Data Provider for z Systems, you must set up the Configuration Tool, use the Configuration Tool to create your policies for streaming data, prepare the target destinations to receive data from the Data Streamer, configure the Data Streamer, and configure the primary data gatherer components, which are the Log Forwarder and the System Data Engine.

Getting started with the Configuration Tool

The IBM Common Data Provider for z Systems Configuration Tool is a web-based user interface that is provided as an application for IBM WebSphere Application Server for z/OS Liberty, or as a plug-in for IBM z/OS Management Facility (z/OSMF). You use the tool to specify what data you want to collect from your z/OS system and where you want that data to be sent. This configuration information is contained in a policy.

About this task

The Configuration Tool helps you create and manage policies for streaming operational data to its destination. The Data Streamer needs this policy information to know what to do with the data that it receives from the data gatherers (such as the System Data Engine and the Log Forwarder).

Each policy definition is stored on the host and secured by the System Authorization Facility (SAF) product that is protecting the system.

Determine on which platform you want to run the Configuration Tool and follow corresponding instructions to set up the Configuration Tool.

Getting started with the Configuration Tool on Liberty

The IBM Common Data Provider for z Systems Configuration Tool can be deployed as an application to IBM WebSphere Application Server for z/OS Liberty.

Before you begin

You must first obtain the **LC27-9535-00 IBM WebSphere Application Server for z/OS DVD** if you plan to use the Liberty based Configuration Tool. If you ordered IBM Common Data Provider for z Systems after 4 March 2019, this DVD is included in your order. However, if you ordered the product on or before 4 March 2019, you must reorder it to obtain this DVD. This DVD includes a Liberty archive, installation JCL, and a README file that provides instructions for setting up Liberty for the application.

The following web browsers are supported:

- Apple Safari
- Google Chrome
- Microsoft Internet Explorer 10 or later
- Mozilla Firefox

Configuring user IDs, group IDs, and security product

You must create user IDs and group IDs with necessary permissions to run the IBM Common Data Provider for z Systems Configuration Tool.

About this task

A default properties file `/usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/cdpui.properties` is provided with default user IDs and group IDs to run the Configuration Tool. You can run the `defracf.cmd` script to

change the default values. The new values are saved in `/var/cdp-uiconfig/cdpui.properties` for the `savingpolicy.sh` script to use in the next task. If you are using RACF as your SAF product, you can allow the script to run necessary RACF commands to create the IDs and permissions. If you do not use RACF, you can exit the script after verifying or changing the values and continue with the configuration.

To run the `defracf.cmd` script, you must be logged in to the z/OS system with a user ID that has the RACF SPECIAL authority.

Important: If this is not the first time you run the script and you are trying to change the user ID and group ID for the started task, before you run the script, you must delete the certificate authority, the certificate, and the keyring that were created last time.

Procedure

1. Run the following script under UNIX System Services to start verifying the default values or changing the values. Only the default z/OS shell is supported.

```
/usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/defracf.cmd
```

2. If necessary, change the user IDs and group IDs to meet your requirements.

STC_USRID

The user ID that is assigned to the Configuration Tool server started task procedure. The default value is HBOSTCID.

Tip: If you want to change this user ID after it is created by running the `defracf.cmd` script, you must first delete the profiles `HBOCFGA.*` and `HBOCFGT.*`, then you can rerun the `defracf.cmd` script to change values. Otherwise, you will see the following messages when you rerun the `defracf.cmd` script, and you will not be able to start the Liberty server:

```
ICH10102I HBOCFGA.* ALREADY DEFINED TO CLASS STARTED.  
ICH10102I HBOCFGT.* ALREADY DEFINED TO CLASS STARTED.
```

STC_GROUP

The group that contains **STC_USRID**. The default value is HBOSTCGP.

AUTHORIZED_GROUP

The group that is granted the permission of logging in and using the Configuration Tool. The default value is HB0USRGP.

GUEST_USER

The user ID that is used by Liberty for accessing the Configuration Tool login page. The default value is HBOGUEST.

GUEST_GROUP

The group that contains **GUEST_USER**. The default value is HBOUNGRP.

AUTHORIZED_USER

The user ID that is granted the permission of logging in and using the Configuration Tool. The default value is HB0USER. You must specify an existing user for this parameter. If you don't specify any value for this parameter, no user is able to access the Configuration Tool. To allow a user to use the Configuration Tool, you must connect the user to the **AUTHORIZED_GROUP** as instructed in [“Allowing users to use the Configuration Tool” on page 17](#).

AUTOID

Determines whether the UID and GID are automatically assigned. The default value is OFF, and you must set values for the following parameters. Make sure that the UIDs and GIDs that you specify meet the requirements of your environment. If the UIDs and GIDs are not accepted by your security product, the Configuration Tool cannot be installed successfully.

STC_USRID_UID

The UID for **STC_USRID**.

STC_GROUP_GID

The GID for **STC_GROUP**.

AUTHORIZED_GROUP_GID

The GID for **AUTHORIZED_GROUP**.

GUEST_USER_UID

The UID for **GUEST_USER**.

GUEST_GROUP_GID

The GID for **GUEST_GROUP**.

If automatic assignment of UID and GID is enabled on your environment, you can change the value of this parameter to ON to have required UIDs and GIDs automatically assigned by the system. In this case, skip the UID and GID parameters that are listed previously.

Remember: All user IDs and group IDs that are specified in the `cdpui.properties` file must be unique. If **AUTOID** is set to OFF, the UIDs and GIDs that are specified must be unique.

3. When you are prompted to choose exit or go, if you are using RACF as your SAF product and you want the script to run RACF commands to create the IDs and permissions, enter G0. Otherwise, enter EXIT to end the script.
 - If you enter G0, check the output from the RACF commands in the `/var/cdp-uiconfig/defracf.log` file and verify that all commands are successfully issued by the script.
 - There should be no RACF error messages from the UNIX System Services issued to the terminal after the script finishes running.
 - If you see the messages ICH10006I, ICH06011I, and IRRD175I indicating that RACLISTED PROFILES must be refreshed before they are effective, and a message "All related RACLIST CLASS are refreshed successfully" after the script finishes running, it means that the RACLISTED PROFILES are refreshed by the script and are effective.
 - Message ICH10102I that says BBG.AUTHMOD.BBGZSAFM, and BBG.AUTHMOD.BBGZSAFM.SAFCRED are already defined, can be safely ignored. These profiles are shared with other Liberty Angel Servers, and they might be defined by a Liberty Angel Server that was created before.
 - If you enter EXIT, you must configure your security product by using the information that is saved in `/var/cdp-uiconfig/cdpui.properties`. If you are using RACF as your SAF product, you can use the commands in [“Configuring the security product by running commands” on page 13](#). If you are not using RACF, you can use these commands to compose equivalent commands for your SAF product.

Tip: If the user ID of **AUTHORIZED_USER** is not found after you run the script, see the troubleshooting topic [“User ID of parameter AUTHORIZED_USER is not found” on page 217](#) for solution.

Important: If you run the script again to change the user ID and group ID for the started task, you must first delete the certificate authority, the certificate, and the keyring that are created this time.

Configuring the security product by running commands

If you are using RACF as your SAF product and you do not want to run the `defracf.cmd` script, you can run RACF commands to create the user IDs and group IDs, and grant them necessary permissions.

About this task

In this task, the following default values of the parameters in the `cdpui.properties` file are used in the code samples.

```
STC_USRID = HBOSTCID
STC_GROUP = HBOSTCGP
AUTHORIZED_GROUP = HBOUSRGP
GUEST_USER = HBOGUEST
GUEST_GROUP = HBOUNGRP
AUTHORIZED_USER = HBOUSER
```

If you are not using default values, make sure to change the values in the samples to the values that you use.

Procedure

1. If any one of classes STARTED, APPL, FACILITY, SERVER, EJBROLE, DIGTCERT, and DIGTRING are not active, run one or more of the following RACF commands to activate them.

```
SETROPTS RACLIST(STARTED) CLASSACT(STARTED)
SETROPTS CLASSACT(APPL)
SETROPTS CLASSACT(FACILITY)
SETROPTS CLASSACT(SERVER)
SETROPTS CLASSACT(EJBROLE)
SETROPTS CLASSACT(DIGTCERT)
SETROPTS CLASSACT(DIGTRING)
```

2. Run the following commands to define the groups and users that are specified in the `cdpui.properties` file.

```
ADDGROUP HBOSTCGP OMVS(GID(3701))
ADDGROUP HBOUSRGP OMVS(GID(3702))
ADDGROUP HBOUNGRP OMVS(GID(3703))
ADDUSER HBOSTCID DFLTGRP(HBOSTCGP) OMVS(UID(2701) HOME(/u/hbostcid)
PROGRAM(/bin/sh)) NAME('CDP UI Server Started Task USERID')
NOPASSWORD NOIDCARD
ADDUSER HBOGUEST RESTRICTED DFLTGRP(HBOUNGRP) OMVS(UID(2702))
NAME('CDPz Unauthenticated USERID') NOPASSWORD NOIDCARD
```

3. Run the following command to allow a user to use the Configuration Tool.

```
CONNECT HBOUSER GROUP(HBOUSRGP)
```

4. Run the following commands to define resource profiles and grant permission to these resource profiles to the user and group for the Configuration Tool server started task procedure.

```
RDEF STARTED HBOCFGA.* UACC(NONE) STDATA(USER(HBOSTCID)
GROUP(HBOSTCGP) PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
RDEF STARTED HBOCFGT.* UACC(NONE) STDATA(USER(HBOSTCID)
GROUP(HBOSTCGP) PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
RDEFINE SERVER BBG.ANGEL.HBOCFG A UACC(NONE)
RDEFINE SERVER BBG.AUTHMOD.BBGZSAFM UACC(NONE)
RDEFINE SERVER BBG.AUTHMOD.BBGZSAFM.SAFCRED UACC(NONE)
PERMIT BBG.ANGEL.HBOCFG A CLASS(SERVER) ACCESS(READ) ID(HBOSTCID)
PERMIT BBG.AUTHMOD.BBGZSAFM CLASS(SERVER) ACCESS(READ) ID(HBOSTCID)
PERMIT BBG.AUTHMOD.BBGZSAFM.SAFCRED CLASS(SERVER) ACCESS(READ)
ID(HBOSTCID)
RDEFINE APPL HBOCFGT UACC(NONE)
RDEFINE SERVER BBG.SECPFH.HBOCFGT UACC(NONE)
PERMIT BBG.SECPFH.HBOCFGT CLASS(SERVER) ACCESS(READ) ID(HBOSTCID)
RDEFINE FACILITY BBG.SYNC.HBOCFGT UACC(NONE)
PERMIT BBG.SYNC.HBOCFGT CLASS(FACILITY) ID(HBOSTCID)
ACCESS(CONTROL)
RDEFINE EJBROLE HBOCFGT.CDPUIserver.cdpUser UACC(NONE)
PERMIT HBOCFGT CLASS(APPL) ID(HBOSTCID) ACCESS(READ)
PERMIT HBOCFGT CLASS(APPL) ID(HBOGUEST) ACCESS(READ)
PERMIT HBOCFGT CLASS(APPL) ID(HBOUSRGP) ACCESS(READ)
PERMIT HBOCFGT.CDPUIserver.cdpUser CLASS(EJBROLE) ID(HBOUSRGP)
ACCESS(READ)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(HBOSTCID)
ACCESS(READ)
RACDCERT CERTAUTH GENCERT SUBJECTSDN(CN('CDPz CA Certification'))
WITHLABEL('HBOCA') TRUST NOTAFTER(DATE(2023/12/31))
RACDCERT ID(HBOSTCID) GENCERT SUBJECTSDN(CN('CDPz DEFAULT CERT'))
WITHLABEL('HBODefaultCert') SIGNWITH(CERTAUTH LABEL('HBOCA'))
NOTAFTER(DATE(2023/12/31))
RACDCERT ADDRING(HBO.Keyring.DFLT) ID(HBOSTCID)
RACDCERT ID(HBOSTCID) CONNECT (LABEL('HBODefaultCert')
RING(HBO.Keyring.DFLT) DEFAULT)
RACDCERT ID(HBOSTCID) CONNECT (LABEL('HBOCA')
RING(HBO.Keyring.DFLT) CERTAUTH)
```

5. If the sharing of in-storage profile is active for any one of classes STARTED, APPL, FACILITY, SERVER, EJBROLE, DIGTCERT, and DIGTRING, run one or more of the following RACF commands to refresh them so that the changes to these classes take effect.

```
SETROPTS RACLIST(STARTED) REFRESH
SETROPTS RACLIST(SERVER) REFRESH
SETROPTS RACLIST(FACILITY) REFRESH
SETROPTS RACLIST(EJBROLE) REFRESH
```



```
SETROPTS RACLIST(APPL) REFRESH
SETROPTS RACLIST(DIGTCERT, DIGTRING) REFRESH
```

Setting up a Liberty server directory and a working directory for the Configuration Tool

You must set up a Liberty server directory to contain the configuration of the Configuration Tool server, and a working directory to store the policy definition files. A setup script (`savingpolicy.sh`) is provided to automate this process.

About this task

User ID criteria for running the setup script

To run the setup script, you must be logged in to the z/OS system with a user ID that has the UID 0 attribute.

Procedure

1. To set up the directories, run the following command with a user ID that meets the criteria that is specified in [About this task](#). Only the default z/OS shell is supported.

```
sh /usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/savingpolicy.sh
```

2. Follow the prompts of the script to provide necessary values.

To accept the default value that is shown in the parentheses, enter a blank value.

- a) When you are prompted to choose if you are deploying the Configuration Tool on z/OSMF or Liberty, enter 2 to select Liberty.
- b) When you are prompted to specify the full path of directory where the Configuration Tool server is installed, accept the default value, or specify the directory that you want to use.
 - The directory must be readable by the **STC_USRID** and **STC_GROUP** that are specified in the `/var/cdp-uiconfig/cdpui.properties` file that is created in [“Configuring user IDs, group IDs, and security product”](#) on page 11.
 - To avoid possible conflicts, do not use a directory that is defined as the Data Streamer working directory (`CDP_HOME`) or the Log Forwarder working directory (`ZLF_WORK`).
- c) If you specify an existing directory, you are prompted for confirmation because the files in the existing directory might be replaced. Enter y to continue with the directory, or enter n to specify another directory.
- d) When you are prompted to specify the Java home directory, accept the default value, or specify the Java home directory for your system.
- e) When you are prompted to specify the Configuration Tool Source Script Directory, accept the default value, or specify a directory that is allowed on your system.

Results

The directory `config_tool_server_install_dir/servers/cdp_ui_server` is created as the Liberty server directory, and the directory `config_tool_server_install_dir/cdpConfig` is created as the working directory, where `config_tool_server_install_dir` is the full path of directory where the Configuration Tool server is installed that you specify in [“2.b”](#) on page 15.

What to do next

The Configuration Tool server uses 17977 as the default port number. If you want to change the port number, complete the following steps:

1. Open the file `server.xml` under the Liberty server directory of the Configuration Tool. By default, you can find this file under `/var/local/CDPServer/servers/cdp_ui_server/`.
2. Change the value of the variable `httpsPort` to the port number you want to use.

Creating the started tasks of the Configuration Tool server and its angel server

Before you can start the IBM Common Data Provider for z Systems Configuration Tool server, you must create the started tasks for the Configuration Tool server and its angel server by copying the sample procedures into a user procedure library, and updating the copies.

Procedure

To create the started tasks, complete the following steps:

1. Copy the procedure HBOCFGT from *hlq*.SHBOSAMP to a user procedure library.
2. Update the procedure HBOCFGT.
 - Change the value of the variable INSTDIR to the path where the WebSphere Application Server for z/OS Liberty is installed.
 - Change the value of the variable USERDIR to the path where the Configuration Tool server is installed. This value is specified in [Step 2.b in Setting up a Liberty server directory and a working directory for the Configuration Tool](#)
3. Copy the procedure HBOCFGGA from *hlq*.SHBOSAMP to a user procedure library.
4. Change the value of the variable WLPDIR in HBOCFGGA to the path where the WebSphere Application Server for z/OS Liberty is installed.

Starting the Configuration Tool server

Start the angel server and the Configuration Tool server before you can access the Configuration Tool in a web browser.

Procedure

1. Start the angel server for the Configuration Tool server by running the following z/OS system console command:

```
START HBOCFGGA
```

2. Verify that the angel server starts successfully.
You see the following message in the job log if the server starts successfully:

```
CWWKB0069I INITIALIZATION IS COMPLETE FOR THE HBOCFGGA ANGEL PROCESS
```

3. Start the Configuration Tool server by running the following console command:

```
START HBOCFGT
```

4. Verify that the Configuration Tool server starts successfully.
You see the following messages in the job log if the server starts successfully:

```
CWWKF0011I: The server cdp_ui_server is ready to run a smarter planet.  
CWWKT0016I: Web application available (default_host): https://hostname:port/cdp/
```

5. Verify that the Configuration Tool server and the angel server are connected successfully.

In the log file `/var/local/CDPServer/servers/cdp_ui_server/logs/messages.log`, you can see the following message:

```
CWWKB0103I: Authorized service group KERNEL is available.
```

If the SAF authorized user registry services and SAF authorization services (SAFCRED) are enabled, the following message is in the log file:

```
CWWKB0103I: Authorized service group SAFCRED is available
```

Tip: The user ID that can read the file `messages.log` must belong to the **STC_GROUP** user group that is specified in the `/var/cdp-uiconfig/cdpui.properties` file that is created in [“Configuring user IDs, group IDs, and security product”](#) on page 11.

Accessing the Configuration Tool

After the Configuration Tool server and the angel server are started, you can access the Configuration Tool in a web browser.

Procedure

1. Access the following URL in a web browser:

```
https://HostName:port/cdp
```

HostName is the host name of your server that runs the Configuration Tool server. *port* is the port number that is used by the Configuration Tool server. The default port number is 17977.

2. Log in as the user ID which is connected to the user group that is defined by the **AUTHORIZED_GROUP** parameter in the `/var/cdp-uiconfig/cdpui.properties` file that is created in [“Configuring user IDs, group IDs, and security product” on page 11](#).

Results

The **"Common Data Provider"** tab opens. Predefined policies are listed.

Allowing users to use the Configuration Tool

To allow a user to use the Configuration Tool, you must connect the user to the user group that is specified in the `cdpui.properties` file. By default, the group ID is `HBOUSGRP`.

About this task

RACF is used as an example in the following instructions. For other SAF products, run corresponding commands to connect the user to the user group.

Procedure

1. Run the following RACF command on the TSO command line as a user ID with RACF `SPECIAL` authority.

```
CONNECT HBOUSER GROUP(HBOUSGRP)
```

Change `HBOUSER` to the user ID that you want to allow to access the Configuration Tool. If you don't use the default `HBOUSGRP` group ID, change it to the value of the **AUTHORIZED_GROUP** parameter that is specified in the `/var/cdp-uiconfig/cdpui.properties` file.

2. Verify that the user ID is connected to the group by running the following RACF command on the TSO command line:

```
LISTUSER HBOUSER
```

Stopping the Configuration Tool server

Run z/OS system console commands to stop the Configuration Tool server and the angel server when necessary.

Procedure

1. Run the following console command to stop the Configuration Tool server:

```
STOP HBOCFG
```

2. Run the following console command to stop the angel server for the Configuration Tool server:

```
STOP HBOCFG
```

Getting started with the Configuration Tool on z/OSMF

The IBM Common Data Provider for z Systems Configuration Tool can be deployed as a plug-in for IBM z/OS Management Facility (z/OSMF).

Before you begin

For information about how to set up z/OSMF so that you can use the IBM Common Data Provider for z Systems Configuration Tool, see the [z/OSMF setup documentation for IBM Common Data Provider for z Systems](#).

Setting up a working directory for the Configuration Tool

Before you install the IBM Common Data Provider for z Systems Configuration Tool, you must set up a working directory where the tool can store policy definition files. A setup script (`savingpolicy.sh`) is provided to automate this process.

About this task

Guidelines for the working directory

Use the following guidelines to help you decide which directory to use as the working directory:

- The directory must be readable and writable by the user ID that runs the Configuration Tool.
- To avoid possible conflicts, do not use a directory that is defined as the Data Streamer working directory (`CDP_HOME`) or the Log Forwarder working directory (`ZLF_WORK`).
- The setup script prompts you for input. To accept the default directory that is shown in the prompt, enter a blank value. If the name for the z/OSMF administrator group of your site is not `IZUADMIN`, ensure that you specify the correct name.

User ID criteria for running the setup script

To run the setup script, you must be logged in to the z/OS system with a user ID that meets the following criteria:

- Because IBM z/OS Management Facility (z/OSMF) administrators need to write updates to the policy definition files, the user ID must be in z/OSMF administrator group 1, which is the UNIX group to which z/OSMF administrators are added. By default, z/OSMF administrator group 1 is `IZUADMIN`.
- The user ID must be a TSO ID that has the UID 0 attribute.

Procedure

To set up the working directory, run the following command with a user ID that meets the criteria that is specified in [About this task](#):

```
sh /usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/savingpolicy.sh
```

The script creates the following path and file:

/u/*userid*/cdpConfig/HBOCDEUI/v1r1m0/LIB path

This path is a symbolic link to target libraries. The `cdpConfig.properties` file is in this path, and it must be imported into z/OSMF when you install the Configuration Tool. For more information about the installation steps, see [“Installing the Configuration Tool” on page 18](#).

/u/*userid*/cdpConfig/HBOCDEUI/v1r1m0/LIB/cdpConfig.json file

This file includes the variable `configPath` that defines the working directory for the Configuration Tool.

Installing the Configuration Tool

To install the IBM Common Data Provider for z Systems Configuration Tool, you must log in to the IBM z/OS Management Facility (z/OSMF), and import the `cdpConfig.properties` file.

Before you begin

Before you install the Configuration Tool, the following tasks must be complete:

1. IBM Common Data Provider for z Systems is installed, and all SMP/E tasks are complete.
2. z/OSMF is installed, configured, and running.

Tip: z/OSMF is running if the started tasks CFZCIM, IZUANG1, and IZUSVR1 are active.

3. The working directory for the Configuration Tool must be set up. For more information, see [“Setting up a working directory for the Configuration Tool”](#) on page 18.

About this task

To install the Configuration Tool, you must be logged in to z/OSMF with a TSO user ID that is in z/OSMF administrator group 1, which is the UNIX group to which z/OSMF administrators are added. By default, z/OSMF administrator group 1 is IZUADMIN.

Procedure

To install the Configuration Tool, complete the following steps:

1. Open z/OSMF in a web browser, and log in with your TSO ID.

Tips:

- The web address, such as `https://host/zosmf/`, is dependent on the configuration of your z/OSMF installation.
- If you cannot log in to z/OSMF, verify that the z/OSMF WebSphere Liberty Profile server is started. The default procedure is IZUSVR1.

2. In the left navigation pane, expand **z/OSMF Administration**, and click **Import Manager**.
3. Type the path and name of the `cdpConfig.properties` file, which is in the path `/u/userid/cdpConfig/HBOCDEUI/v1r1m0/LIB`, and click **Import**.

The import can take several seconds. When it is complete, the following message is shown:

```
Plug-in "Common Data Provider" with tasks
"Common Data Provider" was added to z/OSMF.
To control access, define SAF resource profiles in the
"ZMFAPLA" class for the following SAF resources:
"IZUDFLT.ZOSMF.IBM_CDP.CONFIG.CDPConfiguration".
```

Tip: If you click this resulting message, you are directed to documentation that your systems programmer might need to grant permissions for accessing the plug-in, as indicated in the next step.

4. Have your systems programmer run a SAF command to grant permissions for the z/OSMF administrator group 1 (default is IZUADMIN group) to access the plug-in. For example, if you are using RACF, run the following command:

```
RDEFINE ZMFAPLA +
(IZUDFLT.ZOSMF.IBM_CDP.CONFIG.CDPConfiguration) UACC(NONE) PERMIT +
IZUDFLT.ZOSMF.IBM_CDP.CONFIG.CDPConfiguration +
CLASS(ZMFAPLA) ID(IZUADMIN) ACCESS(CONTROL) PERMIT +
IZUDFLT.ZOSMF.IBM_CDP.CONFIG.CDPConfiguration +
CLASS(ZMFAPLA) ID(IZUUSER) ACCESS(READ)
```

Running the Configuration Tool

To run the IBM Common Data Provider for z Systems Configuration Tool, you must log in to the IBM z/OS Management Facility (z/OSMF).

Before you begin

Install the Configuration Tool, as described in [“Installing the Configuration Tool”](#) on page 18.

About this task

To run the Configuration Tool, you must be logged in to z/OSMF with a TSO user ID that is in z/OSMF administrator group 1, which is the UNIX group to which z/OSMF administrators are added. By default, z/OSMF administrator group 1 is IZUADMIN.

Procedure

1. Open z/OSMF in a web browser, and log in with your TSO ID.
2. In the left navigation pane, expand **Configuration**, and click **Common Data Provider**.

The Common Data Provider tab opens, and you can manage your policies for streaming z/OS operational data to subscribers.

Uninstalling the Configuration Tool

To uninstall the IBM Common Data Provider for z Systems Configuration Tool, you must log in to the IBM z/OS Management Facility (z/OSMF), and import the `cdpConfig.remove.properties` file.

About this task

To uninstall the Configuration Tool, you must be logged in to z/OSMF with a TSO user ID that is in z/OSMF administrator group 1, which is the UNIX group to which z/OSMF administrators are added. By default, z/OSMF administrator group 1 is IZUADMIN.

Procedure

To uninstall the plug-in, complete the following steps:

1. Open z/OSMF in a web browser, and log in with your TSO ID.
2. In the left navigation pane, expand **z/OSMF Administration**, and click **Import Manager**.
3. Type the path and name of the `cdpConfig.remove.properties` file, which is in the path `/u/userid/cdpConfig/HBOCDEUI/v1r1m0/LIB`, and click **Import**.

Results

The Configuration Tool is removed after you import the `cdpConfig.remove.properties` file. There is no confirmation message for the removal.

Output from the Configuration Tool

For each policy that you save, the IBM Common Data Provider for z Systems Configuration Tool creates several files in its working directory.

For example, if you create and save a policy that is named `Sample1`, the following files are created in the Configuration Tool working directory:

- `Sample1.policy`
- `Sample1.layout`
- `Sample1.sde`
- `Sample1.zlf.conf`
- `Sample1.config.properties`
- `cdpkey`

Important: Do not edit the files that the Configuration Tool creates.

The following descriptions explain the purpose of each output file, based on the file name extension:

.policy file

Contains configuration information for the Data Streamer.

.layout file

Contains information about how a policy definition is visually presented in the Configuration Tool.

.sde file

Contains configuration information for the System Data Engine.

.zlf.conf file

Contains environment variables for the Log Forwarder.

.config.properties file

Contains configuration information for the Log Forwarder.

cdpkey file

Contains the key for encrypting the HEC token if used in a policy. This file must be in the same directory as the policy files, which means if you copy the policy files to other directories, you must copy the cdpkey file together.

Managing policies

From the IBM Common Data Provider for z Systems Configuration Tool, you can manage (for example, create, update, and search for) your policies for streaming z/OS operational data to subscribers.

Before you begin

For information about how to run the Configuration Tool, see [“Running the Configuration Tool” on page 19](#).

About this task

A policy includes the following components, which are shown in the Configuration Tool as interconnected nodes in a graph so that you can more easily see how data flows through your system:

Data streams

You must define a *data stream* for each source (such as SMF record type 30 or z/OS SYSLOG) from which you want to collect operational data.

Transforms

In a transform, you specify how to alter the operational data in the stream so that the data is consumable at its target destination. For example, you can use the FixedLength Splitter transform to split data records that have a fixed record length into multiple messages, based on configuration values that you provide.

Subscribers

You must define the subscriber or subscribers for each data stream. A subscriber can be subscribed to multiple data streams.

Tip: Only one policy can be active in each logical partition (LPAR).

Purpose of transforming data in a policy

In a transform, you specify how to alter the operational data in the stream so that the data is consumable at its target destination. The two categories of transform are splitter transforms and filter transforms.

Split and unsplit data

The Data Streamer accepts data in two formats, split and unsplit.

Split data

Split data is divided into records and is sent as an ordered list of individual text strings. Each individual text string represents an individual record from the data source.

Unsplit data

Unsplit data is sent as a single text string and is not divided into records.

Overview of transform categories

The following information provides a brief overview of each transform category and some tips about when to use a transform from that category. Each category includes one or more transforms, which are listed and described in [“Transform configuration” on page 143](#).

Splitter transforms

Based on specified criteria, a splitter transform splits data that is received as one message into multiple messages. Splitter transforms are available only for Generic z/OS Job Output data stream, Generic ZFS File data stream, Generic VSAM Cluster data stream, and zSecure Access Monitor,

Usage tips

- For splitting a single message into multiple messages, based on occurrences of carriage return (CR) or line feed (LF) characters, use the [“CRLF Splitter transform” on page 143](#).
- For splitting data records that have a fixed record length into multiple messages, based on configuration values that you provide, use the [“FixedLength Splitter transform” on page 144](#).

Filter transforms

Based on specified criteria, a filter transform discards messages from the data stream.

Usage tips

- For filtering messages in the data stream according to a regular expression (regex) pattern, which you can define, use the [“Regex Filter transform” on page 145](#).
- For filtering messages in the data stream according to a specified schedule, which you can define, use the [“Time Filter transform” on page 147](#).

For filtering SMF or IMS data: For information about how to filter System Management Facilities (SMF) or IBM Information Management System (IMS) data, see [“Filtering a System Data Engine data stream by using a data stream definition” on page 41](#).

Subscribers to a data stream or transform

A subscriber defines a destination that IBM Common Data Provider for z Systems sends operational data to. A subscriber can be an analytics platform like Splunk, but typically is intermediary software (such as Logstash and the Data Receiver) that is configured to send data to its target destination, which can include analytics platforms such as IBM Z Operations Analytics, Splunk, Elasticsearch, and others.

Streaming protocols for sending data from the Data Streamer to its subscribers

If the subscriber is Logstash or the Data Receiver, the Data Streamer uses a persistent TCP socket connection to send data to the subscriber.

If the subscriber is the Splunk HTTP Event Collector or a generic HTTP or HTTPS subscriber, the Data Streamer uses HTTP or HTTPS to send data to the subscriber.

When you configure a subscriber in a policy, the streaming protocol that you select in the configuration (in the **Protocol** field) defines the following characteristics for sending the data:

- The destination software (such as Logstash)
- The communications protocol, such as a persistent TCP socket connection, HTTP, or HTTPS
- Whether communications are secured by using Transport Layer Security (TLS)

For more information about the streaming protocols, see [“Subscriber configuration” on page 147](#).

Logstash

Logstash is an open source data collection engine that in near real-time, can dynamically unify data from disparate sources and normalize the data into the destinations of your choice for analysis and visualization.

Because IBM Common Data Provider for z Systems does not support forwarding data directly to IBM Z Operations Analytics, for example, the z/OS log data and SMF data that is gathered by IBM Common Data Provider for z Systems must be forwarded to a Logstash instance. The Logstash instance then forwards the data to IBM Z Operations Analytics. Through Logstash, you can also forward data to other destinations, such as Elasticsearch, or route data through a message broker, such as Apache Kafka.

For more information about Logstash, see the [Logstash documentation](#).

Data Receiver

The IBM Common Data Provider for z Systems Data Receiver is software that runs on any platform supporting Java and acts as a target subscriber when the intended subscriber of a data stream cannot directly ingest the data feed from IBM Common Data Provider for z Systems. The Data Receiver writes any data that it receives to disk into files that are sorted by the data source type. These files can then be ingested by an analytics platform.

Depending on the target destinations for the operational data in your environment, you might want to use the IBM Common Data Provider for z Systems Data Receiver. For example, you can use the Data Receiver as the subscriber if the target destination for your operational data is Splunk.

Splunk HTTP Event Collector (HEC)

HEC is an HTTP API endpoint that allows you to send data directly to Splunk over HTTP or HTTPS. For more information about HEC, see [Splunk documentation](#).

If this feature is enabled on Splunk, IBM Common Data Provider for z Systems is able to send data directly to Splunk via HEC instead of sending data through the Data Receiver.

Enabling HEC on Splunk generates a token value for communicating with Splunk. You must provide this value in the subscriber settings when you create a policy.

Generic HTTP subscriber

A generic HTTP subscriber is an application that can receive data that is sent by using an HTTP POST request. An example of a generic HTTP subscriber is a Logstash event processing pipeline that is configured with an HTTP input that uses the JavaScript Object Notation (JSON) codec.

The body of the HTTP POST request is a JSON object with the following members:

host

The TCP/IP host name of the system from which the data originates.

path

The virtual path of the data, which can be a real file path, such as `/home/etc/cdpConfig.log` or a virtual file path, such as `SMF/SMF_030`.

sourceType

The type of the data.

sourceName

A name that is associated with the data stream.

timezone

The base time zone offset for the time stamps in the data.

systemName

The name of the system from which the data originates, as it is defined to the z/OS system.

sysplexName

The name of the sysplex from which the data originates, as it is defined to the z/OS system.

message

The data itself. Depending on whether the data was split, this value might be one string or an array of strings.

Creating a policy

From the Common Data Provider tab in the IBM Common Data Provider for z Systems Configuration Tool, you can create a policy.

About this task


In a policy, you must define at least one data stream with at least one subscriber. For each data stream, you can also define one or more transforms and multiple subscribers. A subscriber can also be subscribed to multiple data streams and transforms.

“Icons on each node in a policy” on page 115 describes the icons that are shown on each data stream, transform, and subscriber node that you define in a policy.

Procedure

Watch this interactive video [How to create a policy in the Configuration Tool](#) to understand the general procedures of creating a policy.

You can also see the following instruction:

1. Click the **Create a new policy** box.
2. In the resulting **Policy Profile Edit** window, type or update the required policy name and, optionally, a policy description.
3. Define any global properties, which are properties that apply to all data streams in the policy.
For information about what you can define, see [“Global properties that you can define for all data streams in a policy”](#) on page 92.
4. Click the **Add Data Stream** icon  **DATA STREAM**.

The “**Select data stream**” window is shown with a list of categorized data streams. You can expand the categories to view the possible data streams that you can define for this policy.





The data stream group **Starter Sets** includes commonly used data streams. These data streams are categorized into various data stream units based on some z/OS components and subsystems instead of based on data types. For more information about **Starter Sets** and other data stream groups, see [“Groups of data streams in the Configuration Tool”](#) on page 96.

Tip: If you create any custom data stream definitions, the associated data streams are listed under the groups that you specified in the stream definitions. For more information about these definitions, see the following information:

- [“Creating a System Data Engine data stream definition”](#) on page 33
- [“Creating an application data stream definition”](#) on page 54

5. Select one or more data streams and click **Select**.

After you click **Select**, each data stream that you choose is shown as a node in the graph. Each node includes the icons that are described in [“Icons on each node in a policy”](#) on page 115.

6. Depending on what you want to define in the policy, use the **Configure** , **Transform** , and **Subscribe**  icons on each node to complete the policy definition. If you want to add more data streams to the policy, use the **Add Data Stream** icon  **DATA STREAM**.

Tips:

- [“Icons on each node in a policy”](#) on page 115 indicates where you can find more information about configuring data streams, transforms, and subscribers, including information about the configuration values.
- To remove a data stream, transform, or subscriber node from a policy definition, click the **Remove** icon (X mark) on the node. When you remove a data stream node or a transform node, any connected transform nodes are also removed.

- If you select **Generic z/OS Job Output**, **Generic ZFS File**, **Generic VSAM Cluster**, or **zSecure**

Access Monitor data streams, you must click the **Transform**  icon on a data stream node, and configure **CRLF Splitter** transform or **FixedLength Splitter** transform. For more information about transforms, see [Transform configuration](#).

7. To save the policy, click **Save**.

The box for the new policy is then shown on the page.

Creating a policy to stream SMF data





From IBM Common Data Provider for z Systems Configuration Tool, you can create a policy to stream SMF data to various subscribers.

About this task

In the policy, select one or more SMF data streams, and add at least one subscriber.

For detailed steps about creating a policy to stream SMF data, refer to the following procedure.

Procedure

1. From the Configuration Tool, click the **Create a new policy** box.
2. In the resulting **Policy Profile Edit** window, type the required policy name and, optionally, a policy description.
3. Click the **Add Data Stream** icon  **DATA STREAM**.
The "**Select data stream**" window is shown with a list of categorized data streams. You can expand the categories to view the possible data streams that you can define for this policy.
4. Select one or more SMF data streams and click **Select**. Each data stream that you choose is shown as a node in the graph.
5. If you want to alter the operational data in the stream, click the **Transform**  icon on a data stream node.
For more information about transforms, see [Transform configuration](#).
6. Click the **Subscribe**  icon on a data stream node, the **Policy Profile Edit** window opens where you can select a previously defined subscriber, or define a new subscriber by completing the following steps.
 - a) In the "**Subscribe to a data stream**" or "**Subscribe to a transform**" window, click the **Add Subscriber** icon .
 - b) In the resulting "**Add subscriber**" window, update the associated configuration values, and click **OK** to save the subscriber.

You can update the following values in the "**Add subscriber**" window:

Name

The name of the subscriber.

Description

An optional description for the subscriber.

Protocol

The streaming protocol that the Data Streamer uses to send data to the subscriber. For example, you can choose **CDP Elasticsearch via Logstash** if you want to use Elastic Stack as a subscriber, or you can choose **CDP Splunk via Data Receiver** if you want to use Splunk as a subscriber. Make sure you choose the protocol that meets your requirements. For more information about protocols, see ["Subscriber configuration" on page 147](#).

Host

The host name or IP address of the subscriber.

Port

The port on which the subscriber listens for data from the Data Streamer.

Auto-Qualify

A specification of whether to prepend system names and sysplex names to data source names in the data streams that are sent to the subscriber. The data source name is the value of the **dataSourceName** field in the data stream configuration.

If you use the same policy file for multiple systems within one sysplex, the data source names must be unique across all systems in that sysplex. If you use the same policy file for multiple sysplexes, the data source names must be unique across all systems in all sysplexes. You can use this field to fully qualify these data source names.

You can choose any of the following values. The default value is None.

None

Indicates that the data source name from the **dataSourceName** field in the data stream configuration is used.

System

Specifies that the system name and the data source name are used in the following format:

```
systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple systems within one sysplex, you might want to use the System value.

Sysplex

Specifies that the sysplex name, system name, and data source name are used in the following format:

```
sysplexName-systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs. *sysplexName* represents the name of the sysplex in which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple sysplexes, you might want to use the Sysplex value.

For more information about the **dataSourceName** field in the data stream configuration, see the following topics:

- [“Data stream configuration for data gathered by Log Forwarder” on page 116](#)
- [“Data stream configuration for data gathered by System Data Engine” on page 143](#)

7. In the "**Subscribe to a data stream**" or "**Subscribe to a transform**" window, select one or more subscribers, and click **Update Subscriptions**.

The subscribers that you choose are then shown on the graph.

8. Click the **SYSTEM DATA ENGINE** button to set the configuration values for your System Data Engine environment.
 - a) In the Global Properties section of the **Policy Profile Edit** window, click **SYSTEM DATA ENGINE**.
 - b) In the "**Configure System Data Engine properties**" window, update the following configuration values for your environment, and click **OK**.

CDP Concatenation

This value must be the name of the SHB0DEFS data set that is installed with IBM Common Data Provider for z Systems in your environment. This data set is also referenced in the `concat.json` file, which is in the working directory for the IBM Common Data Provider for z Systems Configuration Tool.

9. To save the policy, click **Save**.





Creating a policy to stream log data

From IBM Common Data Provider for z Systems Configuration Tool, you can create a policy to stream log data to various subscribers.

About this task

In the policy, select one or more log data, and add at least one subscriber. For example, you can complete the following steps to create a policy to stream SYSLOG or job log data.

Procedure

1. From the Configuration Tool, click the **Create a new policy** box.
2. In the resulting **Policy Profile Edit** window, type the required policy name and, optionally, a policy description.
3. Click the **Add Data Stream** icon  **DATA STREAM**.
The "**Select data stream**" window is shown with a list of categorized data streams. You can expand the categories to view the possible data streams that you can define for this policy.
4. Select one or more SYSLOG or job log data streams and click **Select**. Each data stream that you choose is shown as a node in the graph.
5. If you want to alter the operational data in the stream, click the **Transform**  icon on a data stream node.
For more information about transforms, refer to [Transform configuration](#).
6. Click the **Subscribe**  icon on a data stream node, the **Policy Profile Edit** window opens where you can select a previously defined subscriber, or define a new subscriber to include in the selection list.
To define a new subscriber for a data stream or transform node, complete the following steps:
 - a) In the "**Subscribe to a data stream**" or "**Subscribe to a transform**" window, click the **Add Subscriber** icon  **ADD SUBSCRIBER**.
 - b) In the resulting "**Add subscriber**" window, update the associated configuration values, and click **OK** to save the subscriber.

You can update the following values in the "**Add subscriber**" window:

Name

The name of the subscriber.

Description

An optional description for the subscriber.

Protocol

The streaming protocol that the Data Streamer uses to send data to the subscriber. For example, you can choose **CDP Elasticsearch via Logstash** if you want to use Elastic Stack as a subscriber, or you can choose **CDP Splunk via Data Receiver** if you want to use Splunk as a subscriber. Make sure you choose the protocol that meets your requirements. For more information about protocols, see "[Subscriber configuration](#)" on page 147.

Host

The host name or IP address of the subscriber.

Port

The port on which the subscriber listens for data from the Data Streamer.

Auto-Qualify

A specification of whether to prepend system names and sysplex names to data source names in the data streams that are sent to the subscriber. The data source name is the value of the **dataSourceName** field in the data stream configuration.

If you use the same policy file for multiple systems within one sysplex, the data source names must be unique across all systems in that sysplex. If you use the same policy file for multiple sysplexes, the data source names must be unique across all systems in all sysplexes. You can use this field to fully qualify these data source names.

You can choose any of the following values. The default value is None.

None

Indicates that the data source name from the **dataSourceName** field in the data stream configuration is used.

System

Specifies that the system name and the data source name are used in the following format:

```
systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple systems within one sysplex, you might want to use the System value.

Sysplex

Specifies that the sysplex name, system name, and data source name are used in the following format:

```
sysplexName-systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs. *sysplexName* represents the name of the sysplex in which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple sysplexes, you might want to use the Sysplex value.

For more information about the **dataSourceName** field in the data stream configuration, see the following topics:

- [“Data stream configuration for data gathered by Log Forwarder” on page 116](#)
- [“Data stream configuration for data gathered by System Data Engine” on page 143](#)

7. In the **"Subscribe to a data stream"** or **"Subscribe to a transform"** window, select one or more subscribers, and click **Update Subscriptions**.

The subscribers that you choose are then shown on the graph.

8. Click the **LOG FORWARDER** button to set the configuration values for your Log Forwarder environment.

a) In the Global Properties section of the **Policy Profile Edit** window, click **LOG FORWARDER**.

b) In the **"Configure Log Forwarder properties"** window, update the configuration values for your environment, and click **OK**.

Port

The port on which the Data Streamer listens for data from the Log Forwarder.

Tip: For more information about the Data Streamer port, see [“Configuring the Data Streamer” on page 68](#).

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams.

The value must be an integer in the range 1 - 5. The default value is 1.

Pattern Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for new data sources that match wildcard specifications. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams.

The value must be an integer in the range 0 - 60. The default value is 1.

JRELIB

The fully qualified path to a set of native libraries that are required by the Java Runtime Environment (31-bit).

JRELIB64

The fully qualified path to a set of native libraries that are required by the Java Runtime Environment (64-bit).

REGJAR

The fully qualified path to the `ifaedjreg.jar` file, which provides access to z/OS product registration services.

RESOLVER_CONFIG

The TCP/IP resolver configuration file that the Log Forwarder must use.

The Log Forwarder is a z/OS UNIX System Services program. It uses TCP/IP functions that require access to the TCP/IP resolver configuration file.

For more information, see [“Verifying the search order for the TCP/IP resolver configuration file”](#) on page 91.

TZ

The time zone offset for the Log Forwarder and all data streams from the Log Forwarder.

ZLF_JAVA_HOME

The Java installation directory.

ZLF_HOME

The Log Forwarder installation directory.

ZLF_WORK

The Log Forwarder working directory, which contains files that are created and used during the operation of the Log Forwarder. For example, it includes files that contain information about the state of the Log Forwarder and its progress in collecting data.

Guidelines for the working directory

Use the following guidelines to help you decide which directory to use as the working directory:

- The working directory must be in a different physical location from the working directory for any other Log Forwarder instance.
- The directory must be readable and writable by the user ID that runs the Log Forwarder.
- To avoid possible conflicts, do not use a directory that is defined as the Configuration Tool working directory.

Important: Do not update, delete, or move the files in the Log Forwarder working directory.

ZLF_LOG

The directory for the `logging.properties` file.

ZLF_WAS_PLUGINS_ROOT

IBM WebSphere Application Server installation root directory for Web Server Plug-ins. This directory contains the `com.ibm.hpel.logging.jar` file that is used to retrieve log data from High Performance Extensible Logging (HPEL).

ZLF_GATHERER

The directory for use by data gatherers from a third party organization.

Transport Affinity (environment variable `_BPXK_SETIBMOPT_TRANSPORT`)

The TCP/IP stack to which the Log Forwarder must have affinity. If no value is specified, the Log Forwarder has affinity to the default TCP/IP stack.

9. To save the policy, click **Save**.

Updating a policy

From the Common Data Provider tab in the IBM Common Data Provider for z Systems Configuration Tool, you can update a policy, which can include editing, renaming, duplicating, or deleting the policy.

Procedure

Complete one or more of the following steps, depending on what you want to do:

- To edit a policy, click the box that has the policy name, make changes in the resulting **Policy Profile Edit** window, and click **Save** to save your changes.

Starting from 2Q 2019 PTF, when you edit policies that were created before you apply 2Q 2019 PTF, consider the following changes:

- Only the following transforms are available.
 - CRLF Splitter transform
 - FixedLength Splitter transform
 - Regex Filter transform
 - Time Filter transform




The transcribe transform nodes and their related links will be removed from the UI. The subscribers and other transform nodes that used to link to the transcribe transform will be re-connected to the data stream node directly. Other existing transforms are cleared after you save the changes to the policy.

- The protocols in the subscribers are automatically changed according to the following rules:

Table 5. Protocol change rules	
Old protocol	New protocol
CDP Logstash with the Send as value <code>unsplit</code>	IZOA on IOA-LA via Logstash
CDP Logstash with the Send as value <code>split</code>	IZOA on Elasticsearch via Logstash
CDP Data Receiver	IZOA on Splunk via Data Receiver

Tip: If you create any custom data stream definitions, the associated data streams are listed under the groups that you specified in the stream definitions. For more information about these definitions, see the following information:

- [“Creating a System Data Engine data stream definition” on page 33](#)
- [“Creating an application data stream definition” on page 54](#)

- To rename a policy, click the **Rename** icon  on the box for the policy.
- To duplicate a policy, click the **Duplicate** icon  on the box for the policy.
- To delete a policy, click the **Delete** icon  on the box for the policy.

Tip: “Output from the Configuration Tool” on page 20 describes the files that the Configuration Tool creates for each policy that you save. For recovery purposes, when you delete a policy, the file extension `.hidden` is appended as a suffix to the standard file extension of all the associated policy files. To recover a deleted policy, rename each of the associated policy files to remove `.hidden` from the file extension.

Migrating a policy

If maintenance or a new release of IBM Common Data Provider for z Systems requires additional or different information in the policy files, you must migrate the policies by using the IBM Common Data Provider for z Systems Configuration Tool before you restart IBM Common Data Provider for z Systems. Each policy should only be migrated when the maintenance or new release is applied to the LPAR (or LPARs).

About this task

After you install the PTF and start the Configuration Tool, the **MIGRATE ALL POLICIES TO THE LATEST**

FORMAT button appears, and a migrate button  and a warning icon  appears on each box of the policy that needs migration.

Procedure


Depending on how you want to migrate the policies, complete one of the following steps:

- To migrate all policies, click the **MIGRATE ALL POLICIES TO THE LATEST FORMAT** button.
- To migrate a specific policy, click the migrate button  in the lower-left of each policy box.
- After you apply the 2Q 2019 PTF, to migrate a policy created before you apply the 2Q 2019 PTF, you must open and save the policy again in the Configuration Tool.

Results

If a policy is migrated, the warning icon  and the migrate button  disappear from its policy box. If all policies are migrated, the **MIGRATE ALL POLICIES TO THE LATEST FORMAT** button disappears.

Adding a subscriber for a data stream or transform


When you click the **Subscribe** icon  on a data stream or transform node, a window opens where you can select a previously defined subscriber, or define a new subscriber to include in the selection list. This procedure focuses on how to define a new subscriber.

Before you begin

For information about the types of subscribers that you can choose, see [“Subscribers to a data stream or transform”](#) on page 22.


Procedure

To define a new subscriber for a data stream or transform node, complete the following steps:

1. In the "**Subscribe to a data stream**" or "**Subscribe to a transform**" window, click the **Add Subscriber** icon  **ADD SUBSCRIBER**.
2. In the resulting "**Add subscriber**" window, update the associated configuration values, and click **OK** to save the subscriber.
For more information about the configuration values, see [“Subscriber configuration”](#) on page 147.
3. In the "**Subscribe to a data stream**" or "**Subscribe to a transform**" window, select one or more subscribers, and click **Update Subscriptions**.

The subscribers that you chose are then shown on the graph.

Updating subscriptions of a subscriber


When you click the **Subscribe** icon  on a subscriber node, a window opens where you can update the subscriptions.

Before you begin

For information about how to define a subscriber, see [“Adding a subscriber for a data stream or transform” on page 31](#).

Procedure

To update subscriptions of a subscriber, complete the following steps:

1. In the **"Policy Profile Edit"** window, click the **Subscribe** icon  on a subscriber node.
2. In the resulting **"Select streams for subscription"** window, select or deselect data streams from the list.
The data streams that are invalid for the subscriber can not be selected.
3. Click **UPDATE SUBSCRIPTIONS** for the changes to take effect.


Results

The connection between data streams and the subscriber is shown in the **"Policy Profile Edit"** window.

Exporting and importing subscribers


You can export a subscriber with all transforms and data streams to which it is subscribed. The resulting subscriber file can then be imported into another policy.

About this task

Each subscriber node includes the **Export** icon  that you can use to export the subscriber and its associated data stream and transform nodes.

Procedure

To export a subscriber and import it into another policy, complete the following steps:

1. On the subscriber node, click the **Export** icon .

In the resulting **Export** window, the following check boxes are shown:

Omit layout

By default, an exported subscriber file includes information about how a policy definition is visually presented in the Configuration Tool. This layout information is used to reproduce the positioning of the subscriber node and its associated data stream and transform nodes.

If you do not want to save the layout information, select this check box to omit it. A new layout is then generated when the subscriber is imported.

Export as template

By default, an exported subscriber file includes configuration information that was previously provided (for example, ports, IP addresses, and user names).

If you do not want to save this configuration information, select this check box to export the subscriber file as a template in which all configurable information is reset to the default values.

2. After you optionally select one or both check boxes, click **Export** to download the policy information, which is in a file with the extension `.subscriber`.
The exported policy information in the `.subscriber` file can then be imported into another policy.
3. To import the preconfigured subscriber into a policy, complete either of the following actions:

- Drag and drop the `.subscriber` file onto the policy graph.
- Click the **Import** button at the top of the **Policy Profile Edit** window to browse for the `.subscriber` file to import.

The existing and imported graphs are then merged.

Managing custom data streams


From the IBM Common Data Provider for z Systems Configuration Tool, you can manage custom data stream definitions. For example, you might want to create custom SMF data stream definitions, or application data stream definitions for your own applications, so that you can add these data streams to your policy.

Before you begin

For information about how to run the Configuration Tool, see [“Running the Configuration Tool” on page 19](#).

About this task

You must create a stream definition for each custom data type (analogous to types such as SMF record type 30 or z/OS SYSLOG) from which you want to collect operational data.

These stream definitions are used to populate the Configuration Tool with your data streams, which you can then use in your policy. For example, in the **Policy Profile Edit** window of the Configuration Tool, when you click the **Add Data Stream** icon  **DATA STREAM** to add a data stream to your policy, the **"Select data stream"** window is shown with a list of categorized data streams. You can expand the categories to view the possible data streams that you can define for the policy. After you create your custom data stream definitions, your custom data streams are included in this categorized list.

You can create the following two categories of custom data stream definition:

System Data Engine data stream definition

Specifies a data stream for a type of data that is gathered by the System Data Engine.

Application data stream definition

Specifies a data stream for a type of user application data.

After you create an application data stream definition, you use the Open Streaming API to send your application data to the Data Streamer and stream it to analytics platforms. For more information, see [Chapter 6, “Sending user application data to the Data Streamer,” on page 185](#).

Creating a System Data Engine data stream definition

From the Common Data Provider tab in the IBM Common Data Provider for z Systems Configuration Tool, you can create a System Data Engine data stream definition to use in a policy.

Procedure

1. Before you create a System Data Engine data stream definition, complete the following steps:
 - a) On the z/OS system where the Configuration Tool runs, create a partitioned data set (PDS) that can be used as the concatenation library for storing the data set members that contain the record, update, and, optionally, template definitions for this data stream. Only one concatenation library is required for each Configuration Tool. If you already create a concatenation library, skip this step.

The data set must be defined with the attributes RECFM=VB and LRECL=255, which are the same as those for the SMP/E target data set `HLQ.SHBODEFS`. The following example shows the PDS file `USERID.LOCAL.DEFS`, where `USERID` represents your user ID:

```
Data Set Name . . . . : USERID.LOCAL.DEFS
```

General Data	Current Allocation
Management class . . . : **None**	Allocated cylinders : 10
Storage class . . . : CLASS2	Allocated extents . . : 1
Volume serial . . . : T10062	Maximum dir. blocks : 20
Device type . . . : 3390	
Data class . . . : **None**	Current Utilization
Organization . . . : PO	Used cylinders . . : 1
Record format . . . : VB	Used extents . . . : 1
Record length . . . : 255	Used dir. blocks . . : 1
Block size . . . : 27998	Number of members . : 0
1st extent cylinders: 10	
Secondary cylinders : 5	

Tip: For each new concatenation library (USER concatenation), you must later set it in the System Data Engine properties, as described in [What to do next](#).

- b) Use the System Data Engine language to create the custom record, update, and optionally template definitions, as described in [“Language reference for System Data Engine”](#) on page 150.
2. In the Configuration Tool, click the **MANAGE CUSTOM DATA STREAM DEFINITIONS** button.
3. In the resulting **Manage Custom Data Stream Definitions** window, click the **Create System Data Engine data stream definition** box.
4. In the resulting **Define System Data Engine Data Stream** window, provide values for the following fields:

Name

Specifies the name of the data stream. The name must be the same as the name of the update definition. For example, if the update is defined by the statement `DEFINE UPDATE SMF_CUST_030`, the data stream name must be `SMF_CUST_030`.

This name is converted to uppercase characters when it is saved.

The **Name** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

Group

In the Configuration Tool, the data stream is included in the list of categorized data streams under the main category **Customer Data Streams**. For the hierarchy under **Customer Data Streams**, you must specify the group and subgroup under which you want to include the data stream. The value of **Group** specifies the group. The following example illustrates the hierarchy, and indicates that MYGROUP is specified as the **Group** value:

- **Customer Data Streams**
 - **MYGROUP**

The **Group** value is case-insensitive. For example, if you specify MyGroup as the value, and a group that is named MYGROUP exists under the main category **Customer Data Streams**, the Configuration Tool includes the data stream under MYGROUP, and does not create the category MyGroup.

The **Group** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

Subgroup

In the Configuration Tool, the data stream is included in the list of categorized data streams under the main category **Customer Data Streams**. For the hierarchy under **Customer Data Streams**, you must specify the group and subgroup under which you want to include the data stream. The value of **Subgroup** specifies the subgroup. The following example illustrates the hierarchy, and indicates that MYSUBGROUP is specified as the **Subgroup** value:

- **Customer Data Streams**
 - **MYGROUP**
 - **MYSUBGROUP**

The **Subgroup** value is case-insensitive. For example, if you specify MySubGroup as the value, and a subgroup that is named MYSUBGROUP exists under the group, the Configuration Tool includes the data stream under MYSUBGROUP, and does not create the category MySubGroup.

The **Subgroup** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

SHBODEFS data set members

Specifies the names of the data set members that contain the record, update, and, optionally, template definitions for this data stream.

The value of **SHBODEFS data set members** is case-insensitive and is converted to uppercase characters when it is saved.

You must list one member per line, and the definitions must be listed in the following order:

- a. Record definition
- b. Update definition
- c. Template definition

Therefore, if the definitions are in separate data set members, the members must be listed in the following order:

- a. The member that contains the record definition
- b. The member that contains the update definition
- c. The member that contains the template definition

For example, assume that you want to specify the following data set members:

- HBORSZ30 for record definition
- HBOUSZ30 for update definition
- HBOTSZ30 for template definition

In this case, you must list the members as shown in the following example:

```
HBORSZ30
HBOUSZ30
HBOTSZ30
```

Important:

- When a *custom* data stream is added to a policy, the Configuration Tool searches the concatenation libraries in the following order to find the specified data set members:
 - a. USER concatenation library
 - b. CDP concatenation library

The concatenation libraries are defined in the System Data Engine properties, as described in [“SYSTEM DATA ENGINE properties: Defining your System Data Engine environment” on page 95](#).

5. Click **OK.**

The data stream is created and available to be included in policies.

What to do next

Add the custom data stream to your policy. For information about creating or updating a policy, see the following information:

- [“Creating a policy” on page 24](#)
- [“Updating a policy” on page 30](#)

Remember: Before you save a policy that includes a custom System Data Engine data stream, you must set the USER concatenation library in the System Data Engine properties, as described in [“SYSTEM DATA ENGINE properties: Defining your System Data Engine environment”](#) on page 95.

Streaming IMS user log records to your analytics platform by using a data stream definition

When the IMS Log Write (LOGWRT) user exit is enabled, IMS log records, including IMS user log records, are written to System Management Facilities (SMF) for processing by the System Data Engine. However, to have the System Data Engine process the IMS user log records, you must also use System Data Engine language statements to define custom record and update definitions, and, optionally, to define template definitions.

About this task

After you create custom definitions for System Data Engine data streams, create one or more data streams for the IMS user log records and then update your analytics platform so that it can process the data streams. After that, create or update the policy in the Configuration Tool to include the data streams.

Procedure

1. If you do not already have one, create a partitioned data set (PDS) that is used as the user concatenation library for the custom record and update definitions.

For more information about how to create the data set, see step 1a in [“Creating a System Data Engine data stream definition”](#) on page 33.

2. Copy the sample record definition IMS_USR_F801 from the member HB0URIMS of the SMP/E target data set *hlq.SHBODEFS* to a new member of the user concatenation library and edit the definition based on your requirements.

The following example shows how to define a custom record definition for IMS user log record type x 'F801' based on the sample record definition.

```

/*****
/*
/* IMS Ilog Record Type x'F801'
/*
/*
/*****
DEFINE RECORD IMS_USR_F801
  VERSION 'CDP.110'
  IN LOG SMF
  BUILT BY HBOSDIMS
  IDENTIFIED BY USRTYPE = 248
                AND USRSUBT = 1
  FIELDS
  (

-----
--- 1. IMS Record Prefix
-----
      USRLL    LENGTH 2 BINARY,      -- Length of log record
      USRZZ    LENGTH 2 HEX,         -- QSAM reserved bits
      USRTYPE   LENGTH 1 BINARY,      -- Record type
      USRTYPEH  OFFSET 4 LENGTH 1 HEX, -- Record type in HEX
      USRSUBT   LENGTH 1 BINARY,      -- Record subtype
      USRSUBTH  OFFSET 5 LENGTH 1 HEX, -- Record subtype in HEX

-----
--- 2. IMS Record Data
-----
      fld1 ,
      fld2 ,
      .....
      fldn
  )

-----
--- 3. IMS Record Suffix
-----
  SECTION SUFFIX
    OFFSET USRLL - 16
    LENGTH 16
    NUMBER 1
    FIELDS (

```

```

USRSTCK    LENGTH 8 TIMESTAMP(TOD), -- timestamp
USRLSN     LENGTH 8 HEX              -- log sequence number
);

```

DEFINE RECORD

The following naming convention is used for the record name:

- `IMS_USR_xx` is used if the IMS user log record has no subtype. `xx` is the HEX value of the IMS user log record type.
- `IMS_USR_xxyy` is used if the IMS user log record has subtypes. `xx` is the HEX value of the IMS user log record type, and `yy` is the subtype.

```

DEFINE RECORD IMS_USR_F801

```

USRTYPE

USRTYPE is the binary value of the IMS user log record type. In the example, 248 is the equivalent binary value for HEX `x'F8'`.

```

IDENTIFIED BY USRTYPE = 248

```

USRSUBT

USRSUBT is the binary value of the IMS user log record subtype. In the example, 1 is the equivalent binary value for HEX `x'01'`. If the IMS user log record does not have any subtype, remove this line.

```

AND USRSUBT = 1

```

USRSUBT LENGTH and USRSUBTH OFFSET

If the IMS user log record does not have any subtype, remove these lines.

```

USRSUBT    LENGTH 1 BINARY,          -- Record subtype
USRSUBTH    OFFSET 5 LENGTH 1 HEX,    -- Record subtype in HEX

```

fld1, fld2, fldn

This section defines the fields in the IMS user log record. Fields are separated by commas.

```

fld1 ,
fld2 ,
.....
fldn

```

You can define multiple record definitions in the same member if you have multiple IMS user log record types. For the language reference of the `DEFINE RECORD` statement, see [“DEFINE RECORD statement”](#) on page 161.

3. Copy the sample update definition `IMS_USR_F801` from the member `HB0UUIMS` of the SMP/E target data set `hlq.SHB0DEFS` to a new member of the user concatenation library and edit the definition based on your requirements.

The following example shows how to define an update definition for IMS user log record type `x'F801'` based on the sample update definition.

```

SET IBM_FILE = 'IMSF801';

DEFINE UPDATE IMS_USR_F801
  VERSION 'CDP.110'
  FROM IMS_USR_F801
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);

```

SET

The `SET` statement is needed only when the target of the update definition is a file, which means the variable `IBM_UPDATE_TARTGET` is set to `FILE &IBM_FILE`.

The following naming convention for IMS log record definitions is used in this example:

- `IMSxx` is used if the IMS user log record has no subtype. `xx` is the HEX value of the IMS user log record type.
- `IMSxxyy` is used if the IMS user log record has subtypes. `xx` is the HEX value of the IMS user log record type, and `yy` is the subtype.

```
SET IBM_FILE = 'IMSF801';
```

DEFINE UPDATE

The custom update definition name must be unique among update definitions. In this example, the custom update definition name is the same as the name of the custom record definition that the update definition is associated with.

```
DEFINE UPDATE IMS_USR_F801
```

FROM

The FROM clause identifies the source of the update definition, which is the name of the custom record definition.

```
FROM IMS_USR_F801
```

You can define multiple custom update definitions in the same member if you have multiple IMS user log record types. Also you can use the WHERE clause to select certain records for collection. For the language reference of the DEFINE UPDATE statement, see [“DEFINE UPDATE statement” on page 169](#).

4. Optional: If you want to filter the fields to be collected from the user log records, add a DEFINE TEMPLATE statement for the update definition in the same data set member of that update definition.

In the template definition, you must include the field `USRSTCK`, which is required for timestamp resolution when you ingest data to your analytics platform. For the language reference of the DEFINE TEMPLATE statement, see [“DEFINE TEMPLATE statement” on page 174](#).

5. Validate the syntax of the custom record, update, and, optionally, template definitions.

Use the following example job to verify the members for the custom record and update definitions.

```
//HBOJBCOL JOB ( ), 'DUMMY',MSGCLASS=X,MSGLEVEL=(,0),
//          CLASS=A,NOTIFY=&SYSUID
//*
//HBOJBCOL EXEC PGM=HBOJBCOL,REGION=0M,PARM='SHOWINPUT=YES'
//STEPLIB DD DISP=SHR,DSN=hlq.SHBOLoad
//HBOOUT DD SYSOUT=*
//HBOUMP DD SYSOUT=*
//HBOIN DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCSV)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCORY)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOLLSMF)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBORSIMS)
//        DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRRSIMS)
//        DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUSIMS)
//        DD *
COLLECT SMF
WITH STATISTICS
BUFFER SIZE 1 M;
//*
//HBOLOG DD
DUMMY
```

hlq

Change `hlq` to the high-level qualifier for the IBM Common Data Provider for z Systems SMP/E target data set.

```
//STEPLIB DD DISP=SHR,DSN=hlq.SHBOLoad
//HBOOUT DD SYSOUT=*
//HBOUMP DD SYSOUT=*
//HBOIN DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCSV)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCORY)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOLLSMF)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBORSIMS)
```


HBORSIMS

HBORSIMS contains the record definition of the original SMF record and the record procedure HBOSDIMS that is required for your IMS user log record definition. This member must be included before the member that contains your custom record definition.

```
//          DD    DISP=SHR,DSN=h1q.SHBODEFS(HBORSIMS)
```

DSN=USERID.LOCAL.DEFS(USRRSIMS) and DSN=USERID.LOCAL.DEFS(USRUSIMS)

These two statements specify the members for the custom record and update definitions. *USERID.LOCAL.DEFS* is the USER concatenation library. *USRRSIMS* is the member that contains the custom record definitions, and *USRUSIMS* is the member that contains the custom update definitions. Replace these values based on your configuration. Verify that the custom record definition member is included before the custom update definition member.

```
//          DD    DISP=SHR,DSN=USERID.LOCAL.DEFS(USRRSIMS)
//          DD    DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUSIMS)
```

Important: Verify that the definitions are error-free by running the validation job before you create the custom data stream.

If there is no syntax error, you see the following messages.

```
HB00125I IMS_USR_F801 was successfully defined.
HB00201I Update IMS_USR_F801 was successfully defined.
```

If there are syntax errors, correct the errors according to the messages in the output file that is defined by HB00OUT.

6. Create a custom System Data Engine data stream for each of your custom update definitions.

For more information, see [“Creating a System Data Engine data stream definition” on page 33](#). Verify that you specify the member HBORSIMS before the custom record, update, and, optionally, template definitions in the **SHBODEFS data set members** field.

7. Update your analytics platform so that it can process the new data streams.

- If you are ingesting data to the Elastic Stack, for each data stream, create a field name annotation configuration file, and a timestamp resolution configuration file in the Logstash configuration directory.

Field name annotation configuration file

The file is named *H_data_stream_name.lsh*, for example, *H_IMS_USR_F801.lsh*. Here is an example of the file:

```
# CDPz ELK Ingestion
#
# Field Annotation for stream zOS-IMS_USR_F801
#
filter {
  if [sourceType] == "zOS-IMS_USR_F801" {
    csv{ columns => [ "Correlator", "USRLI", "USRZZ",
"USRTYPE", "USRTYPEH", "USRSUBT", "USRSUBTH", "fld1", "fld2",
"fldn", "USRSTCK", "USRLSN" ]
      separator => "," }
  }
}
```

sourceType

The value of *sourceType* must match the data source type of the data stream. The naming convention is *zOS-data_stream_name*.

```
if [sourceType] == "zOS-IMS_USR_F801"
```

csv{ columns =>

If you have a custom template definition, change the column list to match the fields and order in the template definition. If the first column is `Correlator`, leave it as the first column in the list.

USRSUBT and USRSUBTH

If the IMS user log record does not have any subtype, remove `USRSUBT` and `USRSUBTH`.

fld1, fld2, and fldn

Replace `fld1`, `fld2`, and `fldn` with field names in your custom record definition.

Timestamp resolution configuration file

The file is named `N_data_stream_name.lsh`, for example, `N_IMS_USR_F801.lsh`. Here is an example of the file:

```
# CDPz ELK Ingestion
#
# Timestamp Extraction for stream zOS-IMS_USR_F801
#

filter {
  if [sourceType] == "zOS-IMS_USR_F801" {
    mutate{ add_field => {
      "[@metadata][timestamp]" => "%{USRSTCK}"
    }}

    date{ match => [
      "[@metadata][timestamp]", "yyyy-MM-dd HH:mm:ss.SSSSSS"
    ]}
  }
}
```

sourceType

The value of `sourceType` must match the data source type of the data stream. The naming convention is `zOS-data_stream_name`.

```
if [sourceType] == "zOS-IMS_USR_F801"
```

Restart Logstash after you create the files for all data streams. Refer to Logstash documentation for more information about the configuration files.

- If you are ingesting data to Splunk, define the layout of the data streams to the Splunk server by creating the `props.conf` file in the `Splunk_Home/etc/apps/ibm_cdpz_buffer/local` directory with the following content. If the `props.conf` already exists, append the following lines to that file.

```
#
# IMS_USR_F801
#

[zOS-IMS_USR_F801]
TIMESTAMP_FIELDS = USRSTCK, timezone
TIME_FORMAT= %F %H:%M:%S.%6Q %z
FIELD_NAMES = "sysplex","system","hostname","","","sourcename","timezone",
"Correlator", "USRLL", "USRZZ", "USRTYPE", "USRTYPEH", "USRSUBT", "USRSUBTH",
"fld1", "fld2", "fldn", "USRSTCK", "USRLSN"
INDEXED_EXTRACTIONS = csv
KV_MODE = none
NO_BINARY_CHECK = true
SHOULD_LINEMERGE = false
category = Structured
disabled = false
pulldown_type = true
```

[zOS-IMS_USR_F801]

You must specify the data source name of the data stream. The naming convention is `zOS-data_stream_name`.

FIELD_NAMES

If you have a custom template definition, change the column list to match the fields and order in the template definition. If the column `Correlator` exists, do not remove it.

USRSUBT and USRSUBTH

If the IMS user log record does not have any subtype, remove USRSUBT and USRSUBTH.

f1d1, f1d2, and f1dn


Replace f1d1, f1d2, and f1dn with field names in your record definition.

Tip: The example includes only one data stream. Duplicate the sample code for each new data stream.

In the Splunk user interface, you must also configure the file to data source type mapping for the new data stream. The file that the Data Receiver saves is named `zOS-data_stream_name-*.cdp`. For example, the data stream `IMS_USR_F801` has the file named `CDP-zOS-IMS_USR_F801-*.cdp`.

Restart the Splunk server after you make the changes.

Refer to Splunk documentation for more information.

8. Create or update the policy to add the new System Data Engine data streams so that the IMS user log records can be processed and streamed by the IBM Common Data Provider for z Systems.
 - a) In the Configuration Tool primary window, select the policy that you want to update .
 - b) Click the **Add Data Stream** icon  **DATA STREAM** in the **Policy Profile Edit** window.
 - c) Find and select the new data stream from the list in the **select data stream** window. Select more data streams if you have multiple IMS user log record types.
 - d) Assign a subscriber for each new data stream.
 - e) In the **Policy Profile Edit** window, click **SYSTEM DATA ENGINE** to ensure that values are provided for **USER Concatenation** and **CDP Concatenation** fields, and click **OK**. Following previous examples, `USERID . LOCAL . DEFS` should be specified for the **USER Concatenation** field. Fill in this field with the name of your user concatenation library.
 - f) Click **Save** to save the policy.

Important: Each time that the associated record definition or update definition is changed, you must edit and save the policy in the Configuration Tool so that the changes are reflected in the policy.

For more information on how to update a policy, see [“Updating a policy” on page 30](#).

9. Restart the Data Streamer and the System Data Engine.

Results

The IMS user log records are streamed to your analytics platform.

Filtering a System Data Engine data stream by using a data stream definition

For a System Data Engine data stream, you can use the WHERE clause in the custom update definition to filter the records to be processed, and use the custom template definition that is associated with the update definition to filter the fields to be streamed by IBM Common Data Provider for z Systems.

About this task

After you create custom definitions for System Data Engine data streams, create a custom System Data Engine data stream and then update your analytics platform so that it can process the new data stream. After that, create or update the policy in the Configuration Tool to include the new data stream. If you are adding filters to an existing data stream, you can create the custom definitions and data streams, and update your analytics platform based on the settings for that data stream.

Procedure

1. If you do not already have one, create a partitioned data set (PDS) that is used as the user concatenation library for the custom definitions.

For more information about how to create the data set, see step [1a](#) in [“Creating a System Data Engine data stream definition” on page 33](#).

2. Create a custom update definition in a new or an existing data set member of the user concatenation library. If you want to filter the records to be processed, add a `WHERE` clause to the definition. The name of the member for the custom update definition cannot be the same as any existing member in the `SHBODEFS` data set.
 - You can create a new custom update definition by using the `DEFINE UPDATE` statement. For the language reference of the `DEFINE UPDATE` statement, see [“DEFINE UPDATE statement”](#) on page 169.
 - To filter an existing data stream, copy the update definition that is used by the data stream to a new or an existing data set member of the user concatenation library and update the definition based on your requirements.
 - a. Locate the update definition for the existing data stream by using one of the following methods. The data set members for update definitions are named `HBOUxxxx`.
 - Because the IBM Common Data Provider for z Systems names the data stream with the name of the associated update definition, in your z/OS environment, use ISPF option 3.14 or `SRCHFOR ISPF` command to search the data stream name.
 - Review the data stream definition in the `sde.streams.json` file or the `ims.streams.json` file in the Configuration Tool directory `/usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/`. Check the `hboin` parameter that specifies all data set members for required definitions. Usually the last one is for the update definition.
 - b. Copy the update definition to a new or an existing data set member of the user concatenation library.

The following code sample shows the update definition `SMF_101_1_PACKAGE` in the member `HBOUS101` of the data set `SHBODEFS`.

```
SET IBM_FILE = 'SMF1011K';

DEFINE UPDATE SMF_101_1_PACKAGE
  VERSION 'CDP.110'
  FROM SMF_101_1 SECTION PACKAGE
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);
```

Copy the update definition `SMF_101_1_PACKAGE` to the data set member `USRUS101` in the user concatenation library `USERID.LOCAL.DEFS` with the following changes.

SET

The `SET` statement is needed only when the target of the update definition is a file, which means the variable `IBM_UPDATE_TARGET` is set to `FILE &IBM_FILE`. You can change it to `USR1011K`.

```
SET IBM_FILE = 'USR1011K';
```

DEFINE UPDATE

The data streams must have unique names, so you must rename the update definition to avoid conflict with the existing data stream. You can change it to `USR_101_1_PACKAGE`.

The updated `USERID.LOCAL.DEFS(USRUS101)` member has the following content:

```
SET IBM_FILE = 'USR1011K';

DEFINE UPDATE USR_101_1_PACKAGE
  VERSION 'CDP.110'
  FROM SMF_101_1 SECTION PACKAGE
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);
```

- c. If you want to filter the records to be processed, add a `WHERE` clause to the custom update definition. For example, if you want to collect only Db2 package accounting records whose

transaction name starts with MG, or the authorization ID is U@MUPJ2, add the following WHERE clause:

```
WHERE (SUBSTR(QWHCEUTX,1,2) = 'MG')
OR (QWHCAID = 'U@MUPJ2')
```

The updated *USERID.LOCAL.DEFS*(USRUS101) member has the following content:

```
SET IBM_FILE = 'USR1101K';

DEFINE UPDATE USR_101_1_PACKAGE
  VERSION 'CDP.110'
  FROM SMF_101_1 SECTION PACKAGE
  WHERE (SUBSTR(QWHCEUTX,1,2) = 'MG')
    OR (QWHCAID = 'U@MUPJ2 ')
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);
```

For more information about the WHERE clause, see [“WHERE” on page 171](#).

3. If you want to filter the fields to be streamed, add a **DEFINE TEMPLATE** statement for the update definition in the same data set member of that update definition.

Verify that the template definition is placed after the update definition. The following example shows a template definition in the member *USERID.LOCAL.DEFS*(USRUS101) for the update definition *USR_101_1_PACKAGE* to stream only a few fields in the *PACKAGE* section of record *SMF_101_1* record.

```
SET IBM_FILE = 'USR1101K';

DEFINE UPDATE USR_101_1_PACKAGE
  VERSION 'CDP.110'
  FROM SMF_101_1 SECTION PACKAGE
  WHERE (SUBSTR(QWHCEUTX,1,2) = 'MG')
    OR (QWHCAID = 'U@MUPJ2 ')
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);

DEFINE TEMPLATE USR_101_1_PACKAGE FOR USR_101_1_PACKAGE
  ORDER
  (SM101TME,
  SM101DTE,

  QPACLOCN,
  QPACCOLN,
  QPACPKID,
  QPACSQLC,
  QPACSCB,
  QPACSCE,
  QPACBJST,
  QPACEJST)
  AS &IBM_FILE_FORMAT;
```

DEFINE TEMPLATE

The template definition name must be the same as the update definition name to replace the default template definition that streams all fields for the update definition. In the template definition, you must include the date and time fields from the SMF record header for an SMF record, or the timestamp field in the record suffix for an IMS log record. These fields are required for timestamp resolution when you ingest data to your analytics platform. In this example, the fields are SM101DTE and SM101TME.

For the language reference of the **DEFINE TEMPLATE** statement, see [“DEFINE TEMPLATE statement” on page 174](#).

4. Validate the syntax of the custom update and template definitions.

Use the following example job to verify the members for the custom update and template definitions.

```
//HBOBCOL JOB (), 'DUMMY',MSGCLASS=X,MSGLEVEL=(,0),
//          CLASS=A,NOTIFY=&SYSUID
//*
```

```
//HBOSMFCB EXEC PGM=HBOPDE,REGION=0M,PARM='SHOWINPUT=YES'
//STEPLIB DD DISP=SHR,DSN=hlg.SHBOLoad
//HBOOUT DD SYSOUT=*
//HBODUMP DD SYSOUT=*
//HBOIN DD DISP=SHR,DSN=hlg.SHBODEFS(HBOCCSV)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBOCCORY)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBOLLSMF)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBORS101)
// DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUS101)
// DD *
COLLECT SMF
WITH STATISTICS
BUFFER SIZE 1 M;
//*
//HBOLOG DD DUMMY
```

hlg

Change *hlg* to the high-level qualifier for the IBM Common Data Provider for z Systems SMP/E target data set.

```
//STEPLIB DD DISP=SHR,DSN=hlg.SHBOLoad
//HBOOUT DD SYSOUT=*
//HBODUMP DD SYSOUT=*
//HBOIN DD DISP=SHR,DSN=hlg.SHBODEFS(HBOCCSV)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBOCCORY)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBOLLSMF)
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBORS101)
```

HBORS101

HBORS101 contains the record definition SMF_101_1. This member must be included before the member that contains your custom update and template definitions.

```
// DD DISP=SHR,DSN=hlg.SHBODEFS(HBORS101)
```

// DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUS101)

Specifies the data set member for the custom update and template definition.

Important: Verify that the definitions are error-free by running the validation job before you create the custom data stream.

If there is no syntax error, you see the following messages.

```
HB00201I Update USR_101_1_PACKAGE was successfully defined.
HB00500I Template USR_101_1_PACKAGE was successfully defined.
```

If there are syntax errors, correct the errors according to the messages in the output file that is defined by HBOOUT.

5. Create a custom System Data Engine data stream in the Configuration Tool based on the update definition and template definition that are created in previous steps.

For more information, see [“Creating a System Data Engine data stream definition” on page 33](#). Verify that the data stream name, the custom update definition name, and the custom template definition name are the same, and that you specify the member for the record definition before the member for the custom update and template definitions in the **SHBODEFS data set members** field.

6. Update your analytics platform so that it can process the new data stream.

- If you are ingesting data to the Elastic Stack, for each data stream, create a field name annotation configuration file, and a timestamp resolution configuration file in the Logstash configuration directory.

If your new data stream is created based on an existing one, you can create the two files by copying and editing the files for the old data stream. In previous examples, the new data stream USR_101_1_PACKAGE is created based on the existing data stream SMF_101_1_PACKAGE, and the two configuration files are H_SMF_101_1_PACKAGE.lsh and N_SMF_101_1_PACKAGE.lsh in the Logstash configuration directory. Copy these two files and change the file names to

H_USR_101_1_PACKAGE.lsh and N_USR_101_1_PACKAGE.lsh, then edit the files according to the following instructions.

Field name annotation configuration file

The file is named *H_data_stream_name.lsh*, for example, *H_USR_101_1_PACKAGE.lsh*. See the following example of the file:

```
# CDPz ELK Ingestion
#
# Field Annotation for stream zOS-USR_101_1_PACKAGE
#

filter {
  if [sourceType] == "zOS-USR_101_1_PACKAGE" {
    csv{ columns => [ "Correlator", "SM101TME",
"SM101DTE", "QPACLOCN", "QPACCOLN", "QPACPKID",
"QPACSQLC", "QPACSCB", "QPACSCE", "QPACBJST",
"QPACEJST"]
      separator => "," }
  }
}
```

sourceType

The value of *sourceType* must match the data source type of the data stream. The naming convention is *zOS-data_stream_name*.

```
if [sourceType] == "zOS-USR_101_1_PACKAGE"
```

csv{ columns => []

If you have a custom template definition, change the column list to match the fields and order in the template definition.

Timestamp resolution configuration file

The file is named *N_data_stream_name.lsh*, for example, *N_USR_101_1_PACKAGE.lsh*. See the following example of the file:

```
# CDPz ELK Ingestion
#
# Timestamp Extraction for stream zOS-USR_101_1_PACKAGE
#

filter {
  if [sourceType] == "zOS-USR_101_1_PACKAGE" {
    mutate{ add_field => {
      "[@metadata][timestamp]" => "%{SM101DTE} %{SM101TME}"
    }}

    date{ match => [
      "[@metadata][timestamp]", "yyyy-MM-dd HH:mm:ss:SS"
    ]}
  }
}
```

sourceType

The value of *sourceType* must match the data source type of the data stream. The naming convention is *zOS-data_stream_name*.

```
if [sourceType] == "zOS-USR_101_1_PACKAGE"
```

add_field =>

For an SMF record, you must specify the date and time fields in the SMF record header. In this example, the fields are SM101DTE and SM101TME.

```
"[@metadata][timestamp]" => "%{SM101DTE} %{SM101TME}"
```

For an IMS log record, you must specify the timestamp field in the record suffix. For example, the timestamp field in the IMS_07 record suffix is DLRSTCK.

```
"[@metadata][timestamp]" => "%{DLRSTCK}"
```

match =>

For an SMF record, use the following time format.

```
"[@metadata][timestamp]", "yyyy-MM-dd HH:mm:ss:SS"
```

For an IMS log record, use the following time format.

```
"[@metadata][timestamp]", "yyyy-MM-dd HH:mm:ss.SSSSSS"
```

Restart Logstash after you create the files for the new data stream. Refer to Logstash documentation for more information about the configuration files.

- If you are ingesting data to Splunk, define the layout of the data stream to the Splunk server by creating the `props.conf` file in the `Splunk_Home/etc/apps/ibm_cdpz_buffer/local` directory on the Splunk server.

If your new data stream is created based on an existing one, you can create the file by copying and editing the content for the old data stream. Based on previous examples, open the `props.conf` file in the `Splunk_Home/etc/apps/ibm_cdpz_buffer/default` directory and copy the section for `SMF_101_1_PACKAGE`. Paste the content to the `props.conf` file in `Splunk_Home/etc/apps/ibm_cdpz_buffer/local` and edit it according to the following example. If the `props.conf` file exists, append the content to the file.

```
#
# USR_101_1_PACKAGE (zOS-USR_101_1_PACKAGE)
#

[zOS-USR_101_1_PACKAGE]
TIMESTAMP_FIELDS = SM101DTE, SM101TME, timezone
TIME_FORMAT = %F %H:%M:%S:%2Q %z
FIELD_NAMES = "sysplex","system","hostname","","","sourcename",
"timezone", "Correlator", "SM101TME", "SM101DTE", "QPACLOCN",
"QPACCOLN", "QPACPKID", "QPACSQLC", "QPACSCB", "QPACSCE",
"QPACBJST", "QPACEJST"
INDEXED_EXTRACTIONS = csv
KV_MODE = none
NO_BINARY_CHECK = true
SHOULD_LINEMERGE = false
category = Structured
disabled = false
pulldown_type = true
TRUNCATE = 20000
```

[zOS-USR_101_1_PACKAGE]

You must specify the data source name of the data stream. The naming convention is `zOS-data_stream_name`.

TIMESTAMP_FIELDS

For an SMF record, you must specify the date and time fields in the SMF record header. In this example, the fields are SM101DTE and SM101TME.

```
TIMESTAMP_FIELDS = SM101DTE, SM101TME, timezone
```

For an IMS log record, you must specify the timestamp field in the record suffix. For example, the timestamp field in the IMS_07 record suffix is DLRSTCK.

```
TIMESTAMP_FIELDS = DLRSTCK, timezone
```

TIME_FORMAT

For an SMF record, use the following time format.

```
TIME_FORMAT = %F %H:%M:%S:%2Q %z
```


For an IMS log record, use the following time format.

```
TIME_FORMAT = %F %H:%M:%S.%6Q %z
```

FIELD_NAMES


If you have a custom template definition, change the column list to match the fields and order in the template definition. If the column Correlator exists, do not remove it.

In the Splunk user interface, you must also configure the file to data source type mapping for the new data stream. The file that the Data Receiver saves is named `zOS-data_stream_name-*.cdp`. For example, the data stream `USR_101_1_PACKAGE` has the file that is named `CDP-zOS-USR_101_1_PACKAGE-*.cdp`.

Restart the Splunk server after you make the changes.

Refer to Splunk documentation for more information.

7. Create or update the policy to add the new System Data Engine data stream.

- a) In the Configuration Tool primary window, select the policy that you want to update.
- b) Click the **Add Data Stream** icon  **DATA STREAM** in the **Policy Profile Edit** window.
- c) Find and select the new data stream from the list in the **select data stream** window.
- d) Assign a subscriber for each new data stream.
- e) In the **Policy Profile Edit** window, click **SYSTEM DATA ENGINE** to ensure that values are provided for **USER Concatenation** and **CDP Concatenation** fields, and click **OK**. Complete the field **USER Concatenation** with the data set name of your user concatenation library. Based on previous examples, `USERID` . `LOCAL` . `DEFS` should be specified for the field.
- f) Click **Save** to save the policy.

Important: Each time that the associated update definition or template definition is changed, you must edit and save the policy in the Configuration Tool so that the changes are reflected in the policy.

For more information on how to update a policy, see [“Updating a policy” on page 30](#).

8. Restart the Data Streamer and the System Data Engine.

Results

The records and fields are filtered according to your configuration.

Streaming structured data of your application to analytics platforms by using a data stream definition

In addition to the SMF records that are produced by IBM products and some third-party products, you can use IBM Common Data Provider for z Systems to stream your own SMF records and application data. By using this function, you can stream structured data of your application to analytics platforms.

About this task

If the structured data of your application is not SMF data, you must first write the data to an SMF record. Create custom record definitions, update definitions, and optionally template definitions for this special SMF record by using IBM Common Data Provider for z Systems System Data Engine language to support the structured data of your application. In the Configuration Tool, create data streams for your definitions, and then create or update policies to include the data streams. On the analytics platforms, update the configuration files to support the new data source types.

Procedure

1. If the structured data of your application is available in SMF records, skip this step. If the structured data of your application is not SMF data, write the data to an SMF record as the payload by using the `SMFWTM` or `SMFEWTM` macro, or other methods.

You can use SMF record type 128 through 255, which are for user-written records, to include your structured data.

Because the SMF user exit that is provided by IBM Common Data Provider for z Systems suppresses the recording of the SMF record type 127 subtype 1000, if your SMF is in data set recording mode and you want the records to be processed by IBM Common Data Provider for z Systems only and not recorded to VSAM data sets, you can write your structured data to SMF record type 127 subtype 1000. If you use this record type, you must define the record layout according to the following table. Because this record type is used in IBM Common Data Provider for z Systems by multiple data providers, two additional fields SM127SRC and SM127SRS are defined.

<i>Table 6. SMF record type 127 subtype 1000 layout</i>			
Offset	Data Field	Length	Note
0 (x00)	SM127LEN	2	Record length (maximum size of 32,756) Must be the logical record length including the RDW.
2 (x02)	SM127SEG	2	Segment descriptor Initialize the field with zeros.
4 (x04)	SM127FLG	1	System indicator Turn on bit 1 (x'40') indicating record with subtypes.
5 (x05)	SM127RTY	1	Record type The value must be 127.
6 (x06)	SM127TME	4	Time of record written No need to supply this field if you use SMFWTM or SMFEWTM macro.
10 (x0A)	SM127DTE	4	Date of record written No need to supply this field if you use SMFWTM or SMFEWTM macro.
14 (x0E)	SM127SID	4	System ID No need to supply this field if you use SMFWTM or SMFEWTM macro.
18 (x12)	SM127SSI	4	Subsystem ID The value must be CDP.
22 (x16)	SM127STY	2	Record subtype This value must be 1000.
24 (x18)	SM127SRC	2	User payload type Specify a number between 128 and 255.
26 (x1A)	SM127SRS	2	User payload subtype Use this field to further identify your payload application structured data.
28 (x1C)			Start of user payload data.

Refer to the MVS™ System Management Facilities (SMF) manual for more details on writing SMF records.

- If you do not already have one, create a partitioned data set (PDS) that is used as the user concatenation library for the custom definitions.

For more information about how to create the data set, see step 1a in [“Creating a System Data Engine data stream definition”](#) on page 33.

3. In a data set member of the user concatenation library, create a custom record definition for the SMF records that contain the structured data of your application.

The custom record definition must match the layout of the SMF records. The following code sample creates a member *USERID.LOCAL.DEFS* (USRRS127) to define the record definition for SMF record type 127 subtype 1000 that contains the structured data.

```

/*****
/*
/* SMF Record Type 127 SubType 1000 for User Data
/*
/*
/*****
SET SMF_ABC_RECTYPE = '127' ;
SET SMF_ABC_RECSTYP = '1000' ;
SET SMF_ABC_REC SID = 'CDP' ;
SET SMF_ABC_SRCID = '128' ;

DEFINE RECORD ABC_01
  VERSION 'CDP.110'
  IN LOG SMF
  IDENTIFIED BY SM127RTY = &SMF_ABC_RECTYPE
                AND SM127STY = &SMF_ABC_RECSTYP
                AND SM127SID = &SMF_ABC_REC SID
                AND SM127SRC = &SMF_ABC_SRCID
                AND SM127SRS = 1

  FIELDS (
    -----
    --- Standard SMF record header
    -----
    SM127LEN LENGTH 2 BINARY, -- Record length
    SM127SEG LENGTH 2 BINARY, -- Segment descriptor
    SM127FLG LENGTH 1 HEX, -- System indicator
    SM127RTY LENGTH 1 BINARY, -- Record Type
    SM127TME LENGTH 4 TIME(1/100S), -- Time
    SM127DTE LENGTH 4 DATE(0CYYDDDF), -- Date
    SM127SID LENGTH 4 CHAR, -- System ID
    SM127SSI LENGTH 4 CHAR, -- Subsystem ID
    SM127STY LENGTH 2 BINARY, -- Record subtype
    -----
    --- CDP fields for payload type/subtype
    -----
    SM127SRC LENGTH 2 BINARY, -- Payload Type
    SM127SRS LENGTH 2 BINARY, -- Payload Subtype
    -----
    --- SMF User Data
    -----
    fld1 ,
    fld2 ,
    .....
    fldn
  );

```

SMF_ABC_SRCID

Identifies the payload type of your data. Specify a number between 128 and 225.

DEFINE RECORD

Specifies the name for the custom record definition. The name must be different from other record definitions.

SM127SRC and SM127SRS

Specifies the user data payload type and subtype. Use these fields to further identify the SMF record. SMF record type 127 subtype 1000 uses these two additional fields following the standard SMF record header. If you are using a different SMF record type, ensure that you create the record definition with the correct SMF header.

fld1, fld2, and fldn

Specify the fields for the user data in the SMF record. Fields are separated by commas.

You can define multiple record definitions in the same data set member. Use a different SM127SRS value for each record.

4. In a data set member of the user concatenation library, create a custom update definition to collect the SMF user records.

The following code sample creates a member *USERID.LOCAL.DEFS(USRUS127)* for the update definition.

```
SET IBM_FILE = 'ABC01';

DEFINE UPDATE ABC_01
  VERSION 'CDP.110'
  FROM ABC_01
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT SET(ALL);
```

IBM_FILE

Set a different value of IBM_FILE for each update definition. This value must be a valid DD name.

DEFINE UPDATE

Specifies the update definitions name. The name can be the same as the custom record definition, but it must be different from other update definitions.

FROM

Identifies the source of the update definition, which is the name of the record definition.

You can define multiple update definitions in the same data set member. Also, you can use the WHERE clause to select records to collect. For more information about the WHERE clause, see [“WHERE” on page 171](#).

5. If you want to filter the fields to be streamed, add a DEFINE TEMPLATE statement for the update definition in the same data set member of that update definition.

Verify that the template definition is placed after the update definition and that the template definition name is the same as the update definition name.

In the template definition you must include the SM127TME and SM127DTE fields in the SMF record header for timestamp resolution when you ingest data to analytics platforms.

For the language reference of the DEFINE TEMPLATE statement, see [“DEFINE TEMPLATE statement” on page 174](#).

6. Validate the syntax of the custom record, update, and, optionally, template definitions.

Use the following example job to verify the members for the custom definitions.

```
///HBOJBCOL JOB ( ), 'DUMMY', MSGCLASS=X, MSGLEVEL=(,0),
//          CLASS=A, NOTIFY=&SYSUID
// *
//HBO SMFCB EXEC PGM=HBOPDE, REGION=0M, PARM='SHOWINPUT=YES'
//STEPLIB DD DISP=SHR, DSN=hlq.SHBOLoad
//HBOOUT DD SYSOUT=*
//HBO DUMP DD SYSOUT=*
//HBOIN DD DISP=SHR, DSN=hlq.SHBODEFS(HBOCCSV)
//      DD DISP=SHR, DSN=hlq.SHBODEFS(HBOCCORY)
//      DD DISP=SHR, DSN=hlq.SHBODEFS(HBOLLSMF)
//      DD DISP=SHR, DSN=USERID.LOCAL.DEFS(USRRS127)
//      DD DISP=SHR, DSN=USERID.LOCAL.DEFS(USRUS127)
//      DD *
COLLECT SMF
WITH STATISTICS
BUFFER SIZE 1 M;
// *
//HBO LOG DD DUMMY
```

hlq

Change *hlq* to the high-level qualifier for the IBM Common Data Provider for z Systems SMP/E target data set.

```
// DD DISP=SHR, DSN=USERID.LOCAL.DEFS(USRRS127)
```

```
// DD DISP=SHR, DSN=USERID.LOCAL.DEFS(USRUS127)
```

Specify the data set members for the custom definitions. *USERID.LOCAL.DEFS* is the user concatenation library. *USRRS127* is the member that contains the record definitions. *USRUS127* is the member that contains the update and template definitions. Replace the values based on your

configuration. Verify that the record definition member is included before the update definition member.

Important: Ensure that the definitions are error-free by running the validation job before you create the custom data stream.

Messages are in the output file that is defined by HB00OUT. If there is no syntax error, you see the following messages.

```
HB00125I ABC_01 was successfully defined.
HB00201I Update ABC_01 was successfully defined.
```

If there are syntax errors, correct the errors according to the messages in the output file.

7. Validate the data collection with the custom record, update, and, optionally, template definitions.

Collect data from an SMF data set that contains your SMF records by using a batch System Data Engine job, and validate the data by reviewing the output data set.

Use the following example job to verify the data that is collected with the custom definitions.

```
//HBOJBCOL JOB ( ), 'DUMMY',MSGCLASS=X,MSGLEVEL=(,0),
//          CLASS=A,NOTIFY=&SYSUID
//*
//HBOSMFCB EXEC PGM=HBOPDE,REGION=0M,PARM=' ALLHDRS=YES'
//STEPLIB DD DISP=SHR,DSN=hlq.SHBOLoad
//HB00OUT DD SYSOUT=*
//HB0DUMP DD SYSOUT=*
//HBOIN DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCSV)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOCCORY)
//        DD DISP=SHR,DSN=hlq.SHBODEFS(HBOLLSMF)
//        DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRRS127)
//        DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUS127)
//        DD *
COLLECT SMF
WITH STATISTICS
BUFFER SIZE 1 M;
/*
//HBOLOG DD DISP=SHR,DSN=HLQ.LOCAL.SMFLOGS
//*
//ABC01 DD DSN=USERID.ABC01.CSV,
//        DISP=(NEW,CATLG,DELETE),SPACE=(CYL,(10,10)),
//        DCB=(RECFM=V,LRECL=32756)
```

hlq

Change *hlq* to the high-level qualifier for the IBM Common Data Provider for z Systems SMP/E target data set.

```
// DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRRS127)
```

```
// DD DISP=SHR,DSN=USERID.LOCAL.DEFS(USRUS127)
```

Specify the data set members for the custom definitions. *USERID.LOCAL.DEFS* is the user concatenation library. *USRRS127* is the member that contains the record definitions. *USRUS127* is the member that contains the update and template definitions. Replace the values based on your configuration. Ensure that the record definition member is included before the update definition member.

```
//HBOLOG DD DSN=
```

Specifies the SMF data set that contains your SMF records.

```
//ABC01 DD DSN=
```

Specifies the data set that stores the output data. Ensure that this value is the same as the value of the statement `SET IBM_FILE=` in the corresponding update definition. The output data set is a CSV file which you can download and open with spreadsheet applications for validation.

8. Create a custom System Data Engine data stream in the Configuration Tool based on the update definition and template definition that are created in previous steps.

Create a data stream for each update definition.

Verify that the data stream name, the custom update definition name, and the custom template definition name are the same, and that in the **SHBODEFS data set members** field you specify the

member for the custom record definition before the member for the custom update and template definitions. For more information, see [“Creating a System Data Engine data stream definition”](#) on page 33.

9. Update your analytics platform so that it can process the new data stream.

- If you are ingesting data to the Elastic Stack, for each data stream, create a field name annotation configuration file, and a timestamp resolution configuration file in the Logstash configuration directory.

Field name annotation configuration file

The file is named `H_data_stream_name.lsh`, for example, `H_ABC_01.lsh`. See the following example of the file:

```
# CDPz ELK Ingestion
#
# Field Annotation for stream zOS-ABC_01
#

filter {
  if [sourceType] == "zOS-ABC_01" {

    csv{ columns => [ "Correlator", "SM127LEN", "SM127SEG",
"SM127FLG", "SM127RTY", "SM127TME", "SM127DTE", "SM127SID",
"SM127SSI", "SM127STY", "SM127SRC", "SM127SRS","fld1", "fld2",
"fldn" ]
      separator => "," }

  }
}
```

sourceType

The value of `sourceType` must match the data source type of the data stream. The naming convention is `zOS-data_stream_name`.

```
if [sourceType] == "zOS-ABC_01"
```

fld1, fld2, and fldn

Replace these values with the fields in your custom record definition. If you have a custom template definition, change the column list to match the fields and order in the template definition. Keep `Correlator` as the first column in the list.

Timestamp resolution configuration file

The file is named `N_data_stream_name.lsh`, for example, `N_ABC_01.lsh`. See the following example of the file:

```
# CDPz ELK Ingestion
#
# Timestamp Extraction for stream zOS-ABC_01
#

filter {
  if [sourceType] == "zOS-ABC_01" {
    mutate{ add_field => {
      "[@metadata][timestamp]" => "%{SM127DTE} %{SM127TME}"
    }}

    date{ match => [
      "[@metadata][timestamp]", "yyyy-MM-dd HH:mm:ss:SS"
    ]}

  }
}
```

sourceType

The value of `sourceType` must match the data source type of the data stream. The naming convention is `zOS-data_stream_name`.

```
if [sourceType] == "zOS-ABC_01"
```

add_field =>

For an SMF record, you must specify the date and time fields in the SMF record header. In this example, the fields are SM127DTE and SM127TME.

Restart Logstash after you create the files for the new data stream. Refer to Logstash documentation for more information about the configuration files.

- If you are ingesting data to Splunk, define the layout of the data stream to the Splunk server by creating the `props.conf` file in the `Splunk_Home/etc/apps/ibm_cdpz_buffer/local` directory on the Splunk server. If the `props.conf` file exists, append the following content to the file.

```
#
# ABC_01
#

[zOS-ABC_01]
TIMESTAMP_FIELDS = SM127DTE, SM127TME, timezone
TIME_FORMAT= %F %H:%M:%S:%2Q %z
FIELD_NAMES = "sysplex","system","hostname","","","sourcename","timezone",
"Correlator", "SM127LEN", "SM127SEG", "SM127FLG", "SM127RTY", "SM127TME",
"SM127DTE", "SM127SID", "SM127SSI", "SM127STY", "SM127SRC", "SM127SRS",
"fld1", "fld2", "fldn"
INDEXED_EXTRACTIONS = csv
KV_MODE = none
NO_BINARY_CHECK = true
SHOULD_LINEMERGE = false
category = Structured
disabled = false
pulldown_type = true
```

[zOS-ABC_01]

You must specify the data source name of the data stream. The naming convention is `zOS-data_stream_name`.

TIMESTAMP_FIELDS

For an SMF record, you must specify the date and time fields in the SMF record header. In this example, the fields are SM127DTE and SM127TME.

FIELD_NAMES


Replace `fld1`, `fld2`, and `fldn` with the fields in your custom record definition. If you have a custom template definition, change the column list to match the fields and order in the template definition. If the column `Correlator` exists, do not remove it.

In the Splunk user interface, you must also configure the file to data source type mapping for the new data stream. The file that the Data Receiver saves is named `zOS-data_stream_name-*.cdp`. For example, the data stream `ABC_01` has the file that is named `CDP-zOS-ABC_01-*.cdp`.

Restart the Splunk server after you make the changes.

Refer to Splunk documentation for more information.

10. Create or update the policy to add the new System Data Engine data stream.

- a) In the Configuration Tool primary window, create a new policy or select the policy that you want to update.
- b) Click the **Add Data Stream** icon  **DATA STREAM** in the **Policy Profile Edit** window.
- c) Find and select the new data stream from the list in the **select data stream** window.
- d) Assign a subscriber for each new data stream.
- e) In the **Policy Profile Edit** window, click **SYSTEM DATA ENGINE** to ensure that values are provided for **USER Concatenation** and **CDP Concatenation** fields, and click **OK**. Complete the field **USER Concatenation** with the data set name of your user concatenation library. Based on previous examples, `USERID` . `LOCAL` . `DEFS` should be specified for the field.
- f) Click **Save** to save the policy.

Important: Each time that the associated record definition or update definition is changed, you must edit and save the policy in the Configuration Tool so that the changes are reflected in the policy.

For more information on how to update a policy, see [“Updating a policy” on page 30](#).

11. Restart the Data Streamer and the System Data Engine.

Creating an application data stream definition

From the Common Data Provider tab in the IBM Common Data Provider for z Systems Configuration Tool, you can create an application data stream definition to use in a policy.

Procedure

To create an application data stream, complete the following steps:

1. In the Configuration Tool, click the **MANAGE CUSTOM DATA STREAM DEFINITIONS** button.
2. In the resulting **Manage Custom Data Stream Definitions** window, click the **Create application data stream definition** box.
3. In the resulting **Define Application Data Stream** window, provide values for the following fields:

Name

Specifies the name of the data stream.

This name is converted to uppercase characters when it is saved.

The **Name** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

Group

In the Configuration Tool, the data stream is included in the list of categorized data streams under the main category **Customer Data Streams**. For the hierarchy under **Customer Data Streams**, you must specify the group and subgroup under which you want to include the data stream. The value of **Group** specifies the group. The following example illustrates the hierarchy, and indicates that MYGROUP is specified as the **Group** value:

- **Customer Data Streams**
 - **MYGROUP**

The **Group** value is case-insensitive. For example, if you specify MyGroup as the value, and a group that is named MYGROUP exists under the main category **Customer Data Streams**, the Configuration Tool includes the data stream under MYGROUP, and does not create the category MyGroup.

The **Group** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

Subgroup

In the Configuration Tool, the data stream is included in the list of categorized data streams under the main category **Customer Data Streams**. For the hierarchy under **Customer Data Streams**, you must specify the group and subgroup under which you want to include the data stream. The value of **Subgroup** specifies the subgroup. The following example illustrates the hierarchy, and indicates that MYSUBGROUP is specified as the **Subgroup** value:

- **Customer Data Streams**
 - **MYGROUP**
 - **MYSUBGROUP**

The **Subgroup** value is case-insensitive. For example, if you specify MySubGroup as the value, and a subgroup that is named MYSUBGROUP exists under the group, the Configuration Tool includes the data stream under MYSUBGROUP, and does not create the category MySubGroup.

The **Subgroup** value must contain only alphanumeric characters and underscores. The maximum length is 243 characters.

Tags

Specifies the source or format of the data in the data stream. You can use tags to remind you of the source or format.

If you specify multiple values, you must list one tag per line, as shown in the following example:

```
CSV
Split
```

Any tags that you specify are shown in the Configuration Tool on the data stream and transform nodes.

If the data is being sent in split format, specify the tag `Split`.

If the data is in CSV format, specify the tag `CSV`.

The **Tags** value is case-sensitive and must contain only alphanumeric characters and underscores.

4. Click **OK**.

The data stream is created and available to be included in policies.

What to do next

Add the custom data stream to your policy. For information about creating or updating a policy, see the following information:

- [“Creating a policy” on page 24](#)
- [“Updating a policy” on page 30](#)

Use the Open Streaming API to send your application data to the Data Streamer and stream it to analytics platforms. For more information, see [Chapter 6, “Sending user application data to the Data Streamer,” on page 185](#).

Deleting a custom data stream definition

To delete a custom data stream definition (for either a System Data Engine data stream or an application data stream), you must delete the associated definition file from the Configuration Tool working directory.

Before you begin

Before you delete a custom data stream definition, verify that the associated data stream is not used in a policy. If it is, delete the data stream from the policy.

What happens if you do not delete a data stream before deleting its definition: If the definition for a data stream that is used in a policy is deleted, the data stream remains visible in the policy for the duration of the session with the Configuration Tool. To make the deletion take effect in the user interface, the user must log off the current session, and log on to the Configuration Tool again. If a user then opens a policy that contains the deleted data stream, the following message HB06511W is shown, and the data stream is removed from the policy:

```
HB06511W Definitions are missing for the following data streams:
- SMF_Z30
Please ensure all *.streams.json files used to create this policy
are located within the policy file directory, then restart the
Configuration Tool. All data streams without a matching stream
definition have been removed from the workspace. If you save the
policy now, the unknown data streams will not be included in the
saved policy file.
```

Procedure

To delete a custom data stream definition, complete the following steps:

1. In the Configuration Tool working directory, find the data stream definition file that you want to delete.

The data stream definition file is named `data_stream_name.streams.json`. For example, if the name of the data stream is `SMF_CUST_030`, the data stream definition file is `SMF_CUST_030.streams.json`.

2. To delete the data stream definition file, run the following command:

```
rm data_stream_name.streams.json
```

For example, if you want to delete `SMF_CUST_030.streams.json`, run the following command:

```
rm SMF_CUST_030.streams.json
```

Securing communications between the Data Streamer and its subscribers

To secure communications between the IBM Common Data Provider for z Systems Data Streamer and its subscribers, you must choose a streaming protocol that supports Transport Layer Security (TLS) when you configure a subscriber in a policy. You must also configure the Data Streamer and its subscribers to use TLS.

Before you begin

For more information about the streaming protocols, see “Subscriber configuration” on page 147. The streaming protocols that support TLS contain either SSL or HTTPS in the name. For example, to secure communications between the Data Streamer and the Data Receiver, select the streaming protocol CDP Data Receiver SSL rather than CDP Data Receiver.

Tip: Transport Layer Security (TLS) is the cryptographic protocol that provides secure communications for your connections. Because the Secure Sockets Layer (SSL) protocol is the predecessor to TLS, the term *Secure Sockets Layer*, or *SSL*, is often used generically to refer to TLS encryption.

About this task

The TLS protocol is provided by the IBM Java Runtime Environment that is installed on the z/OS system where the IBM Common Data Provider for z Systems runs, and on the distributed system where the Data Receiver runs. Use Java 8 because by default, it uses the TLS 1.2 protocol, which is the most recent TLS protocol version.

The following scripts for configuring secure communications are provided in the target library `/usr/lpp/IBM/cdpz/v1r1m0/DS/LIB`:

- `setupDataStreamerSSL.sh`
- `setupDataReceiverSSL.sh`
- `setupDataReceiverSSL.bat`
- `importCertificate.sh`

Procedure

To configure the Data Streamer and its subscribers to use TLS, complete the following steps:

1. On each Data Streamer system that must use secure communications with subscribers, set the following environment variables:

JAVA_HOME

The Java installation directory on the Data Streamer system.

CDP_HOME

The Data Streamer working directory that is described in “Configuring the Data Streamer” on page 68.

CDP_DATASTREAMER

The directory that contains the `setupDataStreamerSSL.sh` script and `DataStreamer.jar` file

2. On each Data Streamer system that must use secure communications with subscribers, run the script `setupDataStreamerSSL.sh`, as shown in the following command, where *keystore_password* represents the password that you want to use for the Data Streamer keystore.

This script configures the Data Streamer to use TLS to communicate with subscribers.

```
/usr/lpp/IBM/cdpz/v1r1mo/DS/LIB/setupDataStreamerSSL.sh keystore_password
```

Tip: You are prompted for answers to several questions. For the question `What is your first and last name?`, respond with the fully qualified host name of the Data Streamer system.

The following files are created in the `CDP_HOME` directory:

passStore

Contains a secret key for password encryption.

cdp.properties

Contains the encrypted password for the Data Streamer truststore.

cdp.jks

Keystore to contain the public certificates for the subscribers.

3. For each Data Receiver that must use secure communications with the Data Streamer, complete the following steps on the Data Receiver system.

- a) Set the following environment variables:

JAVA_HOME

The Java installation directory on the Data Receiver system.

CDPDR_HOME

The Data Receiver working directory that is described in [“Setting up a working directory and an output directory for the Data Receiver”](#) on page 65.

The scripts `setupDataReceiverSSL.sh` and `setupDataReceiverSSL.bat` run the Data Receiver during Java key store creation. You must also set up the environment variables `CDPDR_HOME` and `CDPDR_PATH` that are required for running the Data Receiver. For more information, see [“Setting up a working directory and an output directory for the Data Receiver”](#) on page 65.

- b) Download the `setupDataReceiverSSL.sh` (for Linux systems) or `setupDataReceiverSSL.bat` (for Windows systems) file from the IBM Common Data Provider for z Systems system by using a binary protocol.
- c) Move or copy the `setupDataReceiverSSL.sh` or `setupDataReceiverSSL.bat` file into the `CDPDR_HOME` directory where the `DataReceiver.jar` file is located.
- d) Run the script `setupDataReceiverSSL.sh` or `setupDataReceiverSSL.bat`, as shown in the following command.

This script configures the Data Receiver to use TLS to communicate with the Data Streamer. This script requires Java Runtime Environment (JRE) 8.

For Linux systems

```
cd CDPDR_HOME
./setupDataReceiverSSL.sh datareceiver_hostname
datareceiver_ip_address datareceiver_cert_alias
keystore_password
```

For Windows systems

```
cd CDPDR_HOME
setupDataReceiverSSL.bat datareceiver_hostname
datareceiver_ip_address datareceiver_cert_alias
keystore_password
```

The following variables are used in the command:

datareceiver_hostname

The fully qualified host name of the Data Receiver.

datareceiver_ip_address

The IP address of the Data Receiver.

datareceiver_cert_alias

The alias name for the public certificate of the Data Receiver. This name must be used in importing the Data Receiver public certificate to the Data Streamer truststore. The alias is defined when a certificate is imported into a certificate store. When you run the Java **keytool** command to reference a certificate, you must specify the alias. Aliases of all certificates in a key store can be listed using the **keytool** command:

```
keytool -v -list -keystore cdp.jks
```

The **keytool** command will prompt for the Java key store password.

keystore_password

The password that you want to use for the Data Receiver keystore.

Tip: You are prompted for answers to several questions. For the question *What is your first and last name?*, respond with the fully qualified host name of the Data Receiver system.

The following files are created in the *CDPDR_HOME* directory:

passStore

Contains a secret key for password encryption.

cdp.properties

Contains the encrypted password for the Data Receiver keystore.

cdp.jks

Contains the public certificate and private key pair for the Data Receiver.

cdp.cert

The Data Receiver public certificate, which must be imported to the Data Streamer truststore.

4. For any other subscriber that must use secure communications with the Data Streamer, generate a public certificate and private key pair, and configure the subscriber to use them.

For information about how to do this configuration, see the Logstash, Splunk, or other third-party documentation.

5. Transfer the public certificate files from the subscriber systems to the Data Streamer system using a binary mode file transfer. Give each certificate file a unique name.

Important: If a Data Streamer is to send data to more than one subscriber, the public certificate file names must be unique to avoid conflict.

6. On each Data Streamer system, run the script `importCertificate.sh` for each subscriber public certificate, as shown in the following command.

This script imports the public certificate for the subscriber into the Data Streamer truststore.

```
/usr/lpp/IBM/cdpz/v1r1m0/DS/LIB/importCertificate.sh cdp.cert
subscriber_cert_alias
```

The following variables are used in the command:

cdp.cert

The fully qualified path (including the file name) for the file where the public certificate for the subscriber is stored.

subscriber_cert_alias

The alias name for the public certificate of the subscriber. The alias name is used with the **keytool** command to reference certificates and keys in the Java key store.

Preparing the Common Data Provider and the target destinations to stream and receive data

You must prepare your target destinations to receive the z/OS operational data from IBM Common Data Provider for z Systems Data Streamer. The preparation steps differ depending on the target destination.

About this task

For the Data Streamer to stream data to a target destination, you must define the streaming protocol for that target destination in the policy. Table 7 on page 59 lists common target destinations with the required streaming protocols and associated information for preparing the target destination to receive data from the Data Streamer. For more information about defining streaming protocols for sending data from the Data Streamer to its subscribers, see [“Subscribers to a data stream or transform”](#) on page 22.

Table 7. Common target destinations with the required streaming protocols and associated information		
Target destination	Streaming protocol that must be defined in the policy	Steps for preparing target destination to receive data from the Data Streamer
IBM Z Operations Analytics	One of the following protocols: <ul style="list-style-type: none">• IZOA on IOA-LA via Logstash• IZOA on IOA-LA via Logstash SSL	Install the ioaz Logstash output plugin and the Logstash version that are provided with IBM Z Operations Analytics. The Logstash version that is provided with IBM Z Operations Analytics is optimized for use with Linux on z Systems. For more information, see the IBM Z Operations Analytics documentation .

Table 7. Common target destinations with the required streaming protocols and associated information (continued)

Target destination	Streaming protocol that must be defined in the policy	Steps for preparing target destination to receive data from the Data Streamer
Splunk	One of the following protocols: <ul style="list-style-type: none"> • IZOA on Splunk via Data Receiver • IZOA on Splunk via Data Receiver SSL • CDP Splunk via Data Receiver • CDP Splunk via Data Receiver SSL 	Complete the steps that are described in “Preparing to send data to Splunk” on page 61.
	One of the following protocols: <ul style="list-style-type: none"> • CDP Splunk via HEC via HTTP • CDP Splunk via HEC via HTTPS 	Complete the steps that are described in “Preparing to send data to Splunk via the HTTP Event Collector” on page 62.
Elasticsearch	One of the following protocols: <ul style="list-style-type: none"> • IZOA on Elasticsearch via Logstash • IZOA on Elasticsearch via Logstash SSL • CDP Elasticsearch via Logstash • CDP Elasticsearch via Logstash SSL 	Complete the steps that are described in “Preparing to send data to Elasticsearch” on page 64.

Table 7. Common target destinations with the required streaming protocols and associated information (continued)

Target destination	Streaming protocol that must be defined in the policy	Steps for preparing target destination to receive data from the Data Streamer
Other target destinations	One of the following protocols based on your requirements: <ul style="list-style-type: none"> • CDP Logstash • CDP Logstash SSL • CDP Generic HTTP • CDP Generic HTTPS 	Complete the configuration for one of the following subscribers based on the protocol you choose: <ul style="list-style-type: none"> • Logstash receiver, as described in “Configuring a Logstash receiver” on page 67 • Generic HTTP subscriber, as described in “Subscribers to a data stream or transform” on page 22

Preparing to send data to Splunk

To send data from IBM Common Data Provider for z Systems to Splunk, you can use either the IBM Common Data Provider for z Systems Data Receiver, or the HTTP Event Collector (HEC) function in Splunk. Prepare your environment based on the method you choose.

Before you begin

Determine which method to use to send data to Splunk.

- Send data to Splunk by using the Data Receiver.
- Send data by using the HTTP Event Collector.

Sending data by using the Data Receiver has lower CPU usage and smaller data size ingested to Splunk. But you must configure and run an IBM Common Data Provider for z Systems Data Receiver on the system where the Splunk Enterprise server or heavy forwarder is installed. You must also install the IBM Common Data Provider for z Systems Buffered Splunk Ingestion App in Splunk. This method also has slower query and dashboard performance.

Sending data by using the HTTP Event Collector provides quick end-to-end implementation and does not need the Data Receiver and the Buffered Splunk Ingestion App. However, this method will increase the data ingestion size, the cost, and the CPU usage on mainframe.

Preparing to send data to Splunk via the Data Receiver

To send data from IBM Common Data Provider for z Systems to Splunk, configure and run an IBM Common Data Provider for z Systems Data Receiver on the system where the Splunk Enterprise server or heavy forwarder is installed. In Splunk, you must also install the IBM Common Data Provider for z Systems Buffered Splunk Ingestion App.

Procedure

In preparation for sending data to Splunk, complete the following steps:

1. Configure the Data Receiver, as described in [“Configuring the Data Receiver”](#) on page 65.

Important: The Data Receiver environment variables must also be available to Splunk. Verify that the Data Receiver working directory is assigned to the environment variable `CDPDR_HOME`, and that the Data Receiver output directory is assigned to the environment variable `CDPDR_PATH`, as described in [“Setting up a working directory and an output directory for the Data Receiver”](#) on page 65.

2. Start the Data Receiver, as described in [“Running the Data Receiver”](#) on page 177.
3. Define a policy with the Data Receiver as the subscriber.
For more information, see [“Subscribers to a data stream or transform”](#) on page 22.
4. From the IBM Common Data Provider for z Systems /usr/lpp/IBM/cdpz/v1r1m0/DS/LIB directory, download the IBM Common Data Provider for z Systems Buffered Splunk Ingestion App (which is a part of your SMP/E installation package) in binary mode.

The following files contain the App.

Platform on which Splunk runs	File name for Buffered Splunk Ingestion App
UNIX	ibm_cdpz_buffer_nix.spl
Windows	ibm_cdpz_buffer_win.spl

5. To install the Buffered Splunk Ingestion App in Splunk, complete the following steps:
 - a) Log in to Splunk.
 - b) Click the gear icon that is next to the word "Apps."
 - c) Select **Install app from file**.
 - d) Browse for the file that you downloaded in step [“4”](#) on page 62, and select that file.
 - e) When you are prompted, select **Enable now**.

If you are using a Splunk heavy forwarder, you do not have to index the data locally. You can use the system, sysplex, and host attributes to route the data to an appropriate indexer.

If you want to split the indexing locally, you can refine the monitor stanzas in the `inputs.conf` file by extending them to add the sysplex component of the file name. Then, duplicate the monitor stanza for each sysplex from which you want to ingest data, and change the index value on the monitor stanzas to indicate the index in which the data is to be kept. These indexes must be created within Splunk. If you update the IBM Common Data Provider for z Systems Buffered Splunk Ingestion App, this customization is deleted.

Results

You can see the data that is loaded into Splunk by using a simple search. For example, the following search shows you all ingested z/OS SYSLOG events in the zosdex index:

```
index=zosdex sourcetype=zOS-SYSLOG-Console
```

If you expand an event, you can see the individual fields for which extraction rules are set.

The following search example shows you the z/OS SYSLOG messages that are issued by the CICS35 job that is running on your production sysplex and are in the zosdex index:

```
index=zosdex sysplex=PRODPLEX jobname=CICS35 sourcetype=zOS-SYSLOG-Console
```

You can also use Splunk analytics tools to analyze the data, or write your own deep analysis tools.

Preparing to send data to Splunk via the HTTP Event Collector

To stream data to Splunk directly via the HTTP Event Collector (HEC), you must enable HEC in Splunk and create a token that allows an application to communicate with Splunk without using user credentials.

About this task

The following steps are based on the operations in Splunk Enterprise version 7.3.0. For more information, see the topic *Set up and use HTTP Event Collector in Splunk Web* in the Splunk documentation of your version.

Procedure

1. Log on your Splunk server.
2. Go to **Settings > Data Inputs > HTTP Event Collector > Global Settings**.
3. Edit the Global Settings.
 - a) Click the **Enabled** button for the **All Tokens** option.
 - b) If you want to send data via HTTPS, click the **Enable SSL** check box.
To send data via HTTPS, you must configure the Data Streamer to use Transport Layer Security (TLS). For more information, see [“Securing communications between the Data Streamer and its subscribers” on page 56](#).
 - c) In the **HTTP Port Number** field, specify a port number for the HEC to listen on.
 - d) Click **Save**.
4. Go to **Settings > Data Inputs**.
5. Click **+Add New** in the **HTTP Event Collector** row to create a new HEC token.
 - a) In the **Name** field, specify a name for the token.
 - b) If you want to replace the source name for events that this input generates, specify the value in the **Source name override** field.
 - c) Click **Next**.
 - d) In the **Index** section, select the index in which Splunk stores the HEC event data.
It is suggested that you use a test index to verify your data before pushing it to a production index.
Note:
Source type
The source type is determined by the Data Streamer. Any option you choose for Input Settings will be overridden by the Data Streamer.
App context
Application contexts are folders within your Splunk instance that contains configurations for the specific data domain.
 - e) Click **Review** and confirm all settings are correct.
 - f) Click **Submit** to create the HEC token.

Results

The HEC token is created. Take note of the token value for creating policies in the IBM Common Data Provider for z Systems Configuration Tool.

Sample dashboards on Splunk

The IBM Common Data Provider for z Systems provides several sample dashboards in Splunk to visualize mainframe data.

With the sample dashboards you can visualize mainframe operational data like SYSLOG and SMF to view application performance and potential error conditions for CICS, Db2, MQ and z/OS. For more information about the dashboards, see [Dashboards - IBM Common Data Provider for z Systems on Splunkbase](#).

By default these dashboards use an index of `zsdex`, which is the index that the Data Receiver and the IBM Common Data Provider for z Systems Buffered Splunk Ingestion App send data to. To view data that is ingested to another index by the HTTP Event Collector, update the `macro.conf` file under the `ibm_cdpz_dashboards` folder in the Splunk apps directory.

Preparing to send data to Elasticsearch

To send data from IBM Common Data Provider for z Systems to Elasticsearch, configure Logstash by using the Logstash configuration files that are provided by IBM Common Data Provider for z Systems.

About this task

The IBM Common Data Provider for z Systems Elasticsearch ingestion kit contains the Logstash configuration files that are provided by IBM Common Data Provider for z Systems.

Tip: The *Elastic Stack* (formerly known as the *ELK Stack*) is a collection of the popular open source software tools Elasticsearch, Logstash, and Kibana. The IBM Common Data Provider for z Systems Elasticsearch ingestion kit is supported on Elastic Stack Versions 5.1.2 through 5.2.1.

Procedure

In preparation for sending data to Elasticsearch, complete the following steps:

1. From the IBM Common Data Provider for z Systems `/usr/lpp/IBM/cdpz/v1r1m0/DS/LIB` directory, download the IBM Common Data Provider for z Systems Elasticsearch ingestion kit, which is in the `ibm_cdpz_ELK.tar.gz` file, in binary mode.
2. Extract the Elasticsearch ingestion kit to access the Logstash configuration files.
3. Create a new directory under the Logstash installation directory and copy the Logstash configuration files that you need for your environment to the new directory.

Table 8 on page 64 indicates the prefixes that are used in the file names for the Logstash configuration files in the IBM Common Data Provider for z Systems Elasticsearch ingestion kit. The file name prefix is an indication of the configuration file content.

<i>Table 8. Mapping of the prefix that is used in a Logstash configuration file name to the content of the file</i>	
Prefix in file name of Logstash configuration file	Content of configuration file with this prefix
B_	Input stage
E_	Preparation stage
H_	Field name annotation stage
N_	Timestamp resolution stage
Q_	Output stage

The following descriptions further explain the Logstash configuration files in the IBM Common Data Provider for z Systems Elasticsearch ingestion kit:

B_CDPz_Input.1sh file

This file contains the input stage that specifies the TCP/IP port on which Logstash listens for data from the Data Streamer. Copy this file to your Logstash configuration directory. You might need to edit the port number after you copy the file.

E_CDPz_Index.1sh file

This file contains the preparation stage. Copy this file to your Logstash configuration directory.

Files with H_ prefix in file name

Each of these files contains a unique field name annotation stage that maps to a unique data stream that IBM Common Data Provider for z Systems can send to Logstash. To your Logstash configuration directory, copy the H_ files for only the data streams that you want to send to Elasticsearch.

Files with N_ prefix in file name

Each of these files contains a unique timestamp resolution stage that maps to a unique data stream that IBM Common Data Provider for z Systems can send to Logstash. To your Logstash

configuration directory, copy the `N_` files for only the data streams that you want to send to Elasticsearch.

Q_CDPz_Elastic.lsh file

This file contains an output stage that sends all records to a single Elasticsearch server. Copy this file to your Logstash configuration directory.

After you copy the file, edit it to add the name of the host to which the stage is sending the indexing call. The default name is `localhost`, which indexes the data on the server that is running the ingestion processing. Change the value of the **hosts** parameter rather than the value of the **index** parameter. The **index** value is assigned during ingestion so that the data for each source type is sent to a different index. The host determines the Elasticsearch farm in which the data is indexed. The index determines the index in which the data is held.

To split data according to `sysplex`, you can use the `[sysplex]` field in an `if` statement that surrounds an appropriate Elasticsearch output stage.

4. In the script for starting Logstash, specify the directory that you created in step “3” on page 64.
5. Define a policy with the Logstash as the subscriber.
For more information, see “Subscribers to a data stream or transform” on page 22.
6. Start Logstash and Elasticsearch.
If the activation is successful, IBM Common Data Provider for z Systems starts sending data to Elasticsearch.

Configuring the Data Receiver

Before you can use the Data Receiver as a subscriber, you must configure it.

Before you begin

For more information about the Data Receiver, see “Subscribers to a data stream or transform” on page 22.

Setting up a working directory and an output directory for the Data Receiver

You must set up both a working directory and an output directory for the IBM Common Data Provider for z Systems Data Receiver. You must assign the working directory to the environment variable `CDPDR_HOME`, and assign the output directory to the environment variable `CDPDR_PATH`.

About this task

The Data Receiver working directory contains files that are created and used during the operation of the Data Receiver, including the Data Receiver properties and security-related files. The Data Receiver output directory contains output files that the Data Receiver generates based on the data that it receives.

Guidelines for both directories

Use the following guidelines to help you decide which directories to use as the working directory and the output directory:

- The directories must be readable and writable.
- To avoid possible conflicts, do not use the same directory as both the working directory and the output directory for the Data Receiver.
- If you are running multiple Data Receivers in your environment, each Data Receiver must be assigned its own working directory and output directory.

Important: Do not update, delete, or move the files in the `CDPDR_HOME` directory.

Procedure

To set up the working directory and the output directory, complete the step that applies to the platform on which you plan to run the Data Receiver.

Platform on which the Data Receiver runs	Instructions
Linux	<p>Add the following lines either to the system profile or to the profile of the user that starts the Data Receiver:</p> <pre>export CDPDR_HOME=/dr_working_directory export CDPDR_PATH=/dr_output_directory</pre>
Windows	<p>Create system or user environment variables that reference the directories, for example:</p> <pre>CDPDR_HOME=C:\dr_working_directory CDPDR_PATH=C:\dr_output_directory</pre>

Copying the Data Receiver files to the target system

After you set up a working directory for the IBM Common Data Provider for z Systems Data Receiver, you must copy the Data Receiver files to the target system, which is the system on which you plan to run the Data Receiver.

Before you begin

The Data Receiver is in the `DataReceiver.jar` file. The Data Receiver properties are defined in the `cdpdr.properties` sample file. Both of these files are in the `/DS/LIB` directory for IBM Common Data Provider for z Systems.

Procedure

1. Download the `DataReceiver.jar` file and the `cdpdr.properties` sample file from the z/OS system by using a binary protocol.
The `cdpdr.properties` sample file must remain in UTF-8 encoding.
2. Move or copy the `cdpdr.properties` sample file into the Data Receiver working directory (`CDPDR_HOME` directory).

Updating the Data Receiver properties

After you copy the IBM Common Data Provider for z Systems Data Receiver files to the target system, you must update the `cdpdr.properties` sample file in the Data Receiver working directory (`CDPDR_HOME` directory).

About this task

In the `cdpdr.properties` sample file, you can customize the following Data Receiver properties:

port

The port on which the Data Receiver listens for data from the Data Streamer. This port must be the same as the port that is defined for the subscriber in the policy file. For more information, see [“Subscriber configuration” on page 147](#).

cycle

The number of output files that can simultaneously exist in the Data Receiver output directory (`CDPDR_PATH` directory). The minimum value is 3.

The **cycle** property is related to how the Data Receiver manages disk space. For more information about how the Data Receiver manages disk space, see [“Data Receiver process for managing disk space” on page 67](#).

ssl

A y or n specification of whether to use the Transport Layer Security (TLS) protocol for Data Receiver communication with the Data Streamer. If a value other than uppercase Y or lowercase y is used for this property, the Data Receiver disables TLS.

trace

A y or n specification of whether to activate tracing for the Data Receiver. If a value other than uppercase Y or lowercase y is used for this property, the Data Receiver disables tracing. Typically, you activate tracing only at the request of IBM Support.

Procedure

To update the Data Receiver properties, complete the following steps:

1. In the `cdpdr.properties` file in the `CDPDR_HOME` directory, update the property values with your configuration preferences.
2. If you choose to use the TLS protocol for Data Receiver communication with the Data Streamer (`ssl=y` in the `cdpdr.properties` file), also complete the appropriate configuration steps in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

Data Receiver process for managing disk space

The IBM Common Data Provider for z Systems Data Receiver limits the number of output files that can simultaneously exist in its output directory (`CDPDR_PATH` directory). To manage these output files, the Data Receiver uses a cyclic process with rolling files, which are a dynamic, sequential set of files that contain a continuous stream of data.

How the process works

The **cycle** property in the Data Receiver properties file defines the number of output files that can simultaneously exist in the Data Receiver output directory.

When the number of output files in the output directory equals the value of the **cycle** property, and a new file is written, then the oldest file is deleted. Each file contains 1 hour of data. Therefore, if the value of the **cycle** property is set to 3 (3 hours), no more than 3 hours of data (in 3 output files) is on disk at a time.

The following points further illustrate this example of how the Data Receiver manages the output files:

- With the value of the **cycle** property set to 3, the suffixes -0, -1, and -2 are appended to the names of the output files.
- At the beginning of each hour, the Data Receiver erases the old data in the next file in the sequence (if it exists) and writes new data to the file (for example, if it last wrote data to the -0 file, it erases the old data in the -1 file and writes new data to the -1 file).
- One hour later, the Data Receiver erases the old data in the -2 file and writes new data to the -2 file.

Important: The target destination must read the output data in a timely manner. In this example, if the target destination does not read the data within 2 hours of when it is written, the data is lost because it is deleted.

Configuring a Logstash receiver

If you are using a Logstash receiver for target destinations **other than** IBM Z Operations Analytics or Elasticsearch, you must install and configure Logstash on your own. This procedure summarizes how to configure Logstash in this situation.

Before you begin

Remember: If you are sending data to IBM Z Operations Analytics or Elasticsearch, the information in this topic does not apply. Instead, complete the Logstash configuration steps as described in [Table 7](#) on page 59.

About this task

On the distributed Logstash system where you want to send z/OS operational data, you must configure the TCP input plug-in to specify the port on which Logstash listens for data from the Data Streamer.

Procedure

1. To configure the TCP input plug-in, use the following JavaScript Object Notation (JSON) format:

```
input {  
  tcp {  
    port => 8080  
    codec => "json"  
  }  
}
```

2. If you want to configure a secure data connection for streaming operational data from IBM Common Data Provider for z Systems to Logstash, see the [Logstash documentation](#) for information about how to set up Transport Layer Security for the TCP input plug-in.
3. Configure the output plug-in as appropriate for your environment.

Configuring the Data Streamer

The IBM Common Data Provider for z Systems Data Streamer streams operational data to configured subscribers in the appropriate format. It receives the data from the data gatherers (System Data Engine, Log Forwarder, or Open Streaming API), splits it into individual messages when necessary, and sends the data to the subscriber.

Before you begin

The Data Streamer can stream data to both on-platform and off-platform subscribers. To reduce general CPU usage and costs, you can run the Data Streamer on z Systems Integrated Information Processors (zIIPs).

About this task

To configure the Data Streamer, you must create the Data Streamer started task by copying the sample procedure HBODSPRO in the *hlq.SHBOSAMP* library, and updating the copy.

If you want to run the Data Streamer as a job rather than a procedure, use the sample job HBODS001 in the *hlq.SHBOSAMP* library rather than procedure HBODSPRO.

The user ID that is associated with the Data Streamer started task must have the appropriate authority to access the IBM Common Data Provider for z Systems program files, which include the installation files and the policy file. It must also have read/execute permissions to the Java libraries in the UNIX System Services file system.

Procedure

To create the started task, complete the following steps:

1. Copy the procedure HBODSPRO in the *hlq.SHBOSAMP* library to a user procedure library.

Tip: You can rename this procedure according to your installation conventions. When the name HBODSPRO is used in the IBM Common Data Provider for z Systems documentation, including in the messages, it means the Data Streamer started task.

2. In your copy of the procedure HBODSPRO, customize the following parameter values for your environment:

/usr/lpp/IBM/cdpz/v1r1m0/DS/LIB

Replace this value with the directory where the Data Streamer is installed in your environment. This directory contains the *startup.sh* script for the Data Streamer.

/usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/Sample.policy

Replace this value with the policy file path and name for your environment.

nnnnn

Replace this value with the port number on which the Data Streamer listens for data from the data gatherers. The default port on which the Data Streamer listens for data is 51401.

Important: All data gatherers must send data to the Data Streamer through this port. If you update this port in the Data Streamer configuration, you must also update it in the configuration for all data gatherers. For more information, see [“Data Streamer port definition” on page 7](#).

start=w and start=c

This parameter is optional. When stopping, the Data Streamer saves data that have not been sent to the subscriber in a file buffer so that the data can be sent when the Data Streamer is started again. If you want the Data Streamer to load the data in the file buffer on startup and then send it to the subscriber, use `start=w`. Otherwise, use `start=c`. If this parameter is absent, the default value `start=w` takes effect.

3. In your copy of the procedure HBODSPRO, set the following environment variables for your environment:

JAVA_HOME

Specify the Java installation directory.

CDP_HOME

Specify the location of the Data Streamer working directory. The Data Streamer working directory contains files that are created and used during the operation of the Data Streamer, including the Data Streamer truststore and file buffers.

Guidelines for the working directory

Use the following guidelines to help you decide which directory to use as the working directory:

- The directory must be readable and writable by the user ID that runs the Data Streamer.
- To avoid possible conflicts, do not use a directory that is defined as the Configuration Tool working directory.

Important: Do not update, delete, or move the files in the `CDP_HOME` directory.

TZ

Specify the time zone offset for the Data Streamer. For more information, see the information about the format of the `TZ` environment variable in the [z/OS product documentation in the IBM Knowledge Center](#).

Important: If the value of the `TZ` environment variable is incorrect, the time interval that is set in the **Time Filter** transform is directly affected. For more information about the **Time Filter** transform, see [“Time Filter transform” on page 147](#).

RESOLVER_CONFIG

If a TCP/IP resolver must be explicitly provided, uncomment the `RESOLVER_CONFIG` environment variable, and specify the correct TCP/IP resolver. The Data Streamer must have access to a TCP/IP resolver. For more information, see [“Verifying the search order for the TCP/IP resolver configuration file” on page 91](#).

_BPXK_SETIBMOPT_TRANSPORT

If you want the Data Streamer to have affinity to a certain TCP/IP stack, uncomment the `_BPXK_SETIBMOPT_TRANSPORT` environment variable, and specify that TCP/IP stack.

DEFAULT_HEAP

The heap value that is used by the Data Streamer java application by default. `DEFAULT_HEAP=4g` means that the heap size is 4 GB by default. If you want to change the default heap size, uncomment the parameter **DEFAULT_HEAP** and set a new size.

MAXIMUM_HEAP

The maximum heap value that is available to the Data Streamer Java application. `MAXIMUM_HEAP=4g` means that the maximum heap size is 4 GB. If you want to change the maximum heap size, uncomment the parameter **MAXIMUM_HEAP** and set a new size. This value

must be no less than the value of **DEFAULT_HEAP**. For more information about the maximum heap size, see “The Data Streamer maximum heap size” on page 70.

4. Update your security software, such as the Resource Access Control Facility (RACF), to permit the Data Streamer started task to run in your environment.

File buffer function in the Data Streamer

The Data Streamer can buffer data when its TCP/IP connections with the analytics platforms are not stable. When the Data Streamer is stopped, any data that is buffered and not sent to the analytics platforms is written to one or more files in the UNIX System Services file system. When the Data Streamer is restarted, the UNIX System Services files are read by the new instance of the Data Streamer, and the buffered data is sent to the analytics platforms.

The Data Streamer maximum heap size

To handle large volumes of data, the Data Streamer uses 64-bit virtual storage. You can set the limit of the storage amount when you configure the Data Streamer by specifying the value for the parameter **MAXIMUM_HEAP**. This maximum heap size value is found under the STDENV DD statement in the JCL procedure that you use to start the Data Streamer address space. See the following example:

```
//STDENV DD *  
JAVA_HOME=/Java/J8.0_64  
CDP_HOME=/u/dhods/SYSG/cdpz/V1R1M0/DS/data  
TZ=EST5EDT  
DEFAULT_HEAP=4g  
MAXIMUM_HEAP=4g
```

The value for the parameter **MAXIMUM_HEAP** must be no less than the value for the parameter **DEFAULT_HEAP**. In this example, **MAXIMUM_HEAP=4g** means that the limit of virtual storage for the Data Streamer address space is 4 GB.

If your mainframe environment has sufficient real storage to support large usage of virtual storage, you can set the parameter **MAXIMUM_HEAP** to a value higher than 4g. Otherwise, a large **MAXIMUM_HEAP** value might cause high amounts of paging activities between virtual, real, and auxiliary storage.

The data buffer and UNIX System Services file system size

In general, if the TCP/IP connection between the Data Streamer and the analytics platform is stable, set the maximum heap size to 4 GB. However, if the connection is unstable or lost, the Data Streamer might require a larger amount of virtual storage to buffer data.

If the Data Streamer address space is stopped, any data that is buffered and not sent to the analytics platforms is written to one or more files in the UNIX System Services file system on your z/OS LPAR. The amount of data can be as large as the maximum heap size that is specified for the Data Streamer address space. Therefore, you must ensure that the UNIX System Services file system has at least the same available storage as the maximum heap size of the Data Streamer address space. For example, if the value of **MAXIMUM_HEAP** is 4g, the UNIX System Services file system must have at least 4 GB available space to hold the buffered data from the Data Streamer.

Specifying the timestamp format information for streaming custom data types to Splunk via HEC

If you are streaming custom data types to Splunk via HEC, you must add the timestamp format information in the `timestamp.json` file.

Procedure

1. Go to the Data Streamer working directory where the `timestamp.json` file is located.
2. Edit the following parameters.

name

The name of the custom data type.

date

Key for the date field. If **TIMESTAMP** is used, leave this parameter blank.

time

Key for the time field. If **TIMESTAMP** is used, specify **TIMESTAMP** here.

dateFormat

The date format.

timeFormat

The time format.

splunkTimeFormat

The Splunk Time Format.

See the following sample:

```
{
  "name": "SMF_00",
  "date": "SMF0DTE",
  "time": "SMF0TME",
  "dateFormat": "yyyy-MM-dd",
  "timeFormat": "HH:mm:ss:SS",
  "splunkTimeFormat": "%F %H:%M:%S:%2Q %z"
},
Example with time being "TIMESTAMP"
{
  "name": "z/OS SYSLOG",
  "date": "",
  "time": "TIMESTAMP",
  "dateFormat": "yyD",
  "timeFormat": "HH.mm.ss.SSS Z",
  "splunkTimeFormat": "%y%j %H.%M.%S.%3N %z"
},
```

Configuring the data gatherer components

The Log Forwarder and the System Data Engine are the primary data gatherer components of IBM Common Data Provider for z Systems.

About this task

The z/OS File System (zFS) or systems that contain the IBM Common Data Provider for z Systems program files (installation files, configuration files, and working files) can be shared among multiple instances of IBM Common Data Provider for z Systems.

If a single directory contains the Log Forwarder configuration files for more than one system, or logical partition (LPAR), each configuration file name must include the names of the sysplex and the system (LPAR) to which the file applies. The file names must use the following conventions, where *SYSNAME* is the name of the system (LPAR) where the Log Forwarder runs, and *SYSPLEX* is the name of the sysplex (or monoplex) in which that system is located. The values of both *SYSPLEX* and *SYSNAME* must be in all uppercase.

- *SYSPLEX.SYSNAME.zlf.conf*
- *SYSPLEX.SYSNAME.config.properties*

If one file system contains the working directories for multiple instances of IBM Common Data Provider for z Systems, the working directory for each Data Streamer or Log Forwarder instance must be uniquely named.


Configuring the Log Forwarder

Before you run the IBM Common Data Provider for z Systems Log Forwarder to gather z/OS log data, you must configure it.

Before you begin

Before you configure the Log Forwarder, the following policy definition tasks, which are done in the Configuration Tool, must be complete:

1. In the Configuration Tool, create one or more policies that include one or more data streams for z/OS log data.

In the Configuration Tool, when you click the **Configure** icon  on a data stream node for data that is gathered by the Log Forwarder, the "**Configure data stream**" window is shown. [“Data stream configuration for data gathered by Log Forwarder” on page 116](#) lists the configuration values that you can update in this window.

2. After you configure the data streams for z/OS log data, click the **LOG FORWARDER** button, which is in the Global Properties section of the **Policy Profile Edit** window, to set the configuration values for your Log Forwarder environment, as described in [“LOG FORWARDER properties: Defining your Log Forwarder environment” on page 93](#).
3. [“Output from the Configuration Tool” on page 20](#) describes the output from the Configuration Tool, which includes the following Log Forwarder files:

.zlf.conf file

Contains environment variables for the Log Forwarder.

.config.properties file

Contains configuration information for the Log Forwarder.

About this task

To configure the Log Forwarder, you must complete the following tasks:

1. Create the Log Forwarder started task, as described in [“Creating the Log Forwarder started task” on page 72](#).
2. Copy the Log Forwarder configuration files to the *ENVDIR* directory, as described in [“Copying the Log Forwarder configuration files to the ENVDIR directory” on page 74](#).
3. If appropriate for your environment, install the user exit for collecting z/OS SYSLOG data, as described in [“Installing the user exit for collecting z/OS SYSLOG data” on page 75](#).
4. If appropriate for your environment, configure the z/OS NetView message provider for collecting NetView messages, as described in [“Configuring the z/OS NetView message provider for collecting NetView messages” on page 78](#).

Creating the Log Forwarder started task

You must create the started task for the IBM Common Data Provider for z Systems Log Forwarder by copying the sample procedure GLAPROC in the *h1q.SGLASAMP* library, and updating the copy.

Procedure

To create the started task, complete the following steps:

1. Copy the procedure GLAPROC in the *h1q.SGLASAMP* library to a user procedure library.

Tip: You can rename this procedure according to your installation conventions. When the name GLAPROC is used in the IBM Common Data Provider for z Systems documentation, including in the messages, it means the Log Forwarder started task.

2. Update your copy of the GLAPROC procedure, according to the comments in the sample.

Update the following variables:

ENVDIR procedure variable

Specifies the directory where the Log Forwarder configuration files are located. To indicate the variable, the option identifier `-e` precedes the directory specification, as shown in the following example:

```
'-e /etc/IBM/zscala/V3R1'
```

The following directory is the default directory that is used if the *ENVDIR* procedure variable is not specified:

```
/usr/lpp/IBM/zscala/V3R1/samples
```

Important: You must copy the Log Forwarder configuration files to this *ENVDIR* directory, as described in [“Copying the Log Forwarder configuration files to the ENVDIR directory”](#) on page 74.

GLABASE procedure variable

Specifies the directory where the `startup.sh` script is located.

The following directory is the default installation directory for the `startup.sh` script:

```
/usr/lpp/IBM/zscala/V3R1/samples
```

Change the value if a different installation directory was used during the SMP/E installation.

3. Verify that the user ID that is associated with the Log Forwarder started task has the required authorities, as described in [“Requirements for the Log Forwarder user ID”](#) on page 73.
4. Update the SAF product that is protecting your system, such as the Resource Access Control Facility (RACF), to permit the Log Forwarder started task to run in your environment.

Requirements for the Log Forwarder user ID

The user ID that is associated with the Log Forwarder started task must have the required authorities for file access and for issuing z/OS console messages.

The following information further describes the required authorities:

- [“File access authority”](#) on page 73
- [“Authority to issue z/OS console messages”](#) on page 74

Tip: The Log Forwarder user ID does *not* require any special MVS authority to run the Log Forwarder.

File access authority

The Log Forwarder user ID must have the appropriate authority to access the Log Forwarder program files, which include the installation files, the configuration files, and the files in the working directory.

Installation file access

The Log Forwarder user ID must have read and execute permissions to the Log Forwarder installation files in the UNIX System Services file system.

Configuration file access

The Log Forwarder user ID must have read permission to the Log Forwarder configuration files in the UNIX System Services file system.

Important: The user ID that configures the Log Forwarder must have read/write permission to the configuration files.

Working directory access

The Log Forwarder user ID must have read and write permissions to the Log Forwarder working directory, which is described in [“Log Forwarder properties configuration”](#) on page 93. The Log Forwarder user ID must also have permission to change the permission bits for a file in the Log Forwarder working directory.

Authority to issue z/OS console messages

The Log Forwarder user ID must have the authority to issue z/OS console messages.

If you are using the Resource Access Control Facility (RACF) as your System Authorization Facility (SAF) product, complete one of the following options to assign this authority:

Option 1 if you are using RACF

You can use the GLARACF procedure in the SGLASAMP library to create a user ID for the Log Forwarder started task (GLAPROC procedure) and associate that user ID with the started task. The documentation that is provided in the GLARACF sample includes more information, and the following steps outline this process:

1. Copy the GLARACF procedure to a user job library.
2. To define a user ID and associate it with the Log Forwarder started task (GLAPROC procedure), update your copy of the GLARACF procedure according to the comments in the sample and the following instructions:
 - If the user ID exists, comment out the ADDUSER statement.
 - If a user ID other than GLALGF is to be associated with the GLAPROC procedure, change the USER value on the **STDATA** parameter.
3. Submit your updated copy of the GLARACF procedure.

Option 2 if you are using RACF

Complete the following steps:

1. In RACF, add the BPX.CONSOLE resource to the class FACILITY by using the General Resource Profiles option in the RACF Services Option Menu.
2. In the BPX.CONSOLE profile that was created (or updated) in the preceding step, add the user ID that the Log Forwarder started task is associated with, and assign read access to the user ID.
3. Issue the following command to activate your changes:

```
SETROPTS RACLIST(FACILITY) REFRESH
```

Tips:

- The user ID that the GLARACF procedure creates is named GLALGF. The Log Forwarder started task does not require the user ID to be GLALGF. This user ID is provided only as a convenience.
- If the SAF product for your environment is not RACF, use the GLARACF sample procedure and the SAF product documentation to create the appropriate definitions in the SAF product.

Copying the Log Forwarder configuration files to the *ENVDIR* directory

After you create the started task for the IBM Common Data Provider for z Systems Log Forwarder, copy the Log Forwarder configuration files from the Configuration Tool working directory to the directory that is specified by the *ENVDIR* procedure variable in the Log Forwarder started task.

Procedure

Copy the `.zlf.conf` and `.config.properties` files from the Configuration Tool working directory to the *ENVDIR* directory. The names of the files in the *ENVDIR* directory must be `zlf.conf` and `config.properties`, or `SYSPLEX.SYSTEM.zlf.conf` and `SYSPLEX.SYSTEM.config.properties`, where *SYSPLEX* is the name of the sysplex and *SYSTEM* is the name of the system.

For more information about these files, see [“Output from the Configuration Tool” on page 20](#).

Installing the user exit for collecting z/OS SYSLOG data

If you configure a **z/OS SYSLOG** data stream for gathering z/OS SYSLOG data from a user exit, you must install either the GLASYSYG or GLAMDBG user exit. If you configure only a **z/OS SYSLOG from OPERLOG** data stream for gathering z/OS SYSLOG data, the installation of a user exit is not necessary.

About this task

The GLASYSYG and GLAMDBG user exits, and other modules that are used by these user exits, are provided with IBM Common Data Provider for z Systems and are in the SGLALPA product library.

All modules in the SGLALPA library must be added to the system link pack area (LPA). For more information about the LPA, see the *z/OS MVS Initialization and Tuning Guide*.

The following modules are in the SGLALPA library:

- GLADSRW (a program call module)
- GLAGDSDL (a program call module)
- GLAGLMSG (a program call module)
- GLAMDBG
- GLASYSYG
- GLAUERQ (a program call module)

You must install the GLASYSYG or GLAMDBG user exit on the appropriate MVS installation exit. [Table 9 on page 75](#) indicates the MVS installation exit on which to install each user exit and describes how to choose which user exit to install.

Both user exits allocate a data space with a minimum size of 100 MB and a maximum size of 500 MB. The data space is used to store z/OS SYSLOG data for retrieval by the z/OS SYSLOG.

Table 9. User exits for collecting z/OS SYSLOG data, with associated MVS installation exits and usage notes

User exit	MVS installation exit on which to install the user exit	Usage note for user exit
GLASYSYG	CNZ_MSGTOSYSLOG	If your z/OS system is not running JES3 with the DLOG option enabled, install this user exit.
GLAMDBG	CNZ_WTOMDBEXIT	If your z/OS system is running JES3 with the DLOG option enabled, install this user exit.

Procedure

To install the user exit, complete the following steps:

1. To add the load modules to an LPA, complete one of the following actions:

Action	Instruction
Add the SGLALPA library to the pageable link pack area (PLPA) at system IPL	<p>Add the following statement to an LPALSTxx member:</p> <pre>zscala.v3r1.SGLALPA(volume)</pre> <p>Replace <code>zscala.v3r1</code> with the target library high-level qualifier that is used to install IBM Common Data Provider for z Systems, and replace <code>volume</code> with the volume identifier of the data set.</p>
Add the individual modules in the SGLALPA library to the	<p>Issue the following MVS system commands:</p> <pre>SETPROG LPA,ADD,MODNAME=GLASYSYG,DSNAME=zscala.v3r1.SGLALPA SETPROG LPA,ADD,MODNAME=GLAMDBG,DSNAME=zscala.v3r1.SGLALPA</pre>

Action	Instruction
dynamic LPA after the system IPL	<pre> SETPROG LPA,ADD,MODNAME=GLADSRW,DSNAME=zscala.v3r1.SGLALPA SETPROG LPA,ADD,MODNAME=GLAGDSDL,DSNAME=zscala.v3r1.SGLALPA SETPROG LPA,ADD,MODNAME=GLAGLMSG,DSNAME=zscala.v3r1.SGLALPA SETPROG LPA,ADD,MODNAME=GLAUERQ,DSNAME=zscala.v3r1.SGLALPA </pre>

2. To install the exit, complete one of the following actions:

Action	Instruction
Install the user exit on an MVS installation exit at system IPL	<p>Add one of the following statements to a PROGxx member:</p> <ul style="list-style-type: none"> EXIT ADD EXITNAME(CNZ_MSGTOSYSLOG) MODNAME(GLASYSYG) EXIT ADD EXITNAME(CNZ_WTOMDBEXIT) MODNAME(GLAMDBG)
Dynamically install the user exit after the system IPL	<p>Issue one of the following MVS commands:</p> <ul style="list-style-type: none"> SETPROG EXIT,ADD,EXITNAME=CNZ_MSGTOSYSLOG,MODNAME=GLASYSYG SETPROG EXIT,ADD,EXITNAME=CNZ_WTOMDBEXIT,MODNAME=GLAMDBG

manageUserExit utility for managing the installed user exit

The GLASYSYG and GLAMDBG user exits create system resources that might need to be managed while they are in operation. The manageUserExit utility is a shell script that can be used to manage the system resources. The utility is included in the product samples directory in the hierarchical file system.

The following system resources might need to be managed:

- A data space, which is used to store z/OS SYSLOG data for retrieval by the Log Forwarder.
- Program call modules, which are loaded by the user exit and made available to other programs (such as the Log Forwarder and the manageUserExit utility) for interacting with the data space.

manageUserExit.sh description

This utility manages the data space and program call modules that are controlled by the user exit. For example, you can use the utility to complete the following management actions:

- Refresh the data space.
- Refresh the program call modules.
- Delete the data space, unload the program call modules, and uninstall the user exit from the MVS installation exit.

Important: Before you run the manageUserExit.sh utility, stop any instances of the Log Forwarder that are gathering z/OS SYSLOG data. This action prevents the Log Forwarder from trying to access or call a system resource that is being deleted. An abend might occur if the Log Forwarder accesses a non-existent data space or calls a non-existent program call module.

manageUserExit.sh details

Format

```
manageUserExit.sh -p[d] [environment_configuration_directory]
```

```
manageUserExit.sh -d[p] [environment_configuration_directory]
```

```
manageUserExit.sh -u [environment_configuration_directory]
```

Parameters

-d

Refreshes the data space by deleting and re-creating it.

For normal operations, refreshing the data space is not needed. However, for example, if you are requested to refresh the data space by IBM Software Support, use this parameter to delete and re-create the data space. All z/OS SYSLOG data that is in the data space before deletion is lost.

-p

Refreshes the program call modules by unloading and reloading from the LPA.

Refreshing the program call modules might be necessary when maintenance is applied. Updates to the modules in the SGLALPA library must be reloaded by the user exit. Use this parameter to unload the previously loaded program call modules and load the new program call modules.

Tips:

1. Before you refresh the program call modules, the modules must be loaded dynamically into the system LPA. If the program call modules are currently in the dynamic LPA, the user exit must be uninstalled, and the old program call modules must be deleted from the dynamic LPA before the new modules can be reloaded. The user exit must then be reinstalled on the MVS installation exit.
2. If the application of maintenance requires a refresh of the program call modules, the maintenance information specifies that a refresh is necessary.

-u

Deletes the data space, unloads the program call modules, and uninstalls the user exit.

Examples

manageUserExit.sh -pd /etc/IBM/zscala/V3R1

This command refreshes both the data space and program call modules. In this example, the directory /etc/IBM/zscala/V3R1 contains the environment configuration file.

manageUserExit.sh -u

This command uses the *ZLF_CONF* environment variable to find the directory that contains the environment configuration file. It also deletes the data space, unloads the program call modules, and uninstalls the user exit.

Exit values

0

Successful completion

-1

Did not complete successfully

Messages

The utility issues messages to standard output. The messages have the prefix GLAK.

manageUserExit.sh usage notes

The following information describes some tips for using the `manageUserExit.sh` utility:

- To run the `manageUserExit.sh` utility, you must specify at least one parameter.
- Specification of the environment configuration directory is optional. However, if this directory is not specified, the *ZLF_CONF* environment variable must be set, and its value must be the working directory that contains the `zlf.conf` file that is used by the Log Forwarder.

For example, if the `zlf.conf` file is in /etc/IBM/zscala/V3R1, either the environment configuration directory or the value of the *ZLF_CONF* environment variable must be this directory.

- The -p and -d parameters cannot be used with the -u parameter.

- The utility requests operations by using a system common storage area. The requested operation does not complete until the user exit is called by a system console message. The requested operations are not run synchronously by the utility.
- The utility can be run even if the user exit is not active or installed. The requested operations are completed when the user exit is activated and is called by a system console message.
- When the utility completes successfully, it indicates only that it made a request of the user exit. A system console message is issued by the user exit when it performs the requested operations.

Configuring the z/OS NetView message provider for collecting NetView messages

If you configure a **NetView Netlog** data stream for gathering NetView for z/OS message data, you must also configure the NetView message provider to monitor and forward NetView for z/OS messages to the Log Forwarder. The NetView message provider is defined in the REXX module GLANETV in the SGLACLS data set.

About this task

The NetView message provider must be associated with a NetView autotask and can be run as a long-running command to get NetView for z/OS messages. The NetView autotask to which you associate the NetView message provider must have the following permissions:

- Permission to access and edit the configuration directory for the NetView message provider by using the queued sequential access method (QSAM).
- Permission to issue CZR messages by using the **PIPE** and **CNMECZFS** commands
- Permission to use the **LISTVAR** and **PPI** commands

Procedure

To prepare the NetView message provider for use, complete the following steps:

1. Ensure that the GLANETV module is placed in the DSICLD data set that is defined in the NetView procedure. Also, ensure that the NetView autotask has access to the GLANETV module.
2. In the CNMSTYLE member, specify the following information by using common variables:

Information to specify	How to specify	Example entry in CNMSTYLE member
Indication of whether to start the NetView message provider in cold or warm start mode	Specify either of the following values for the <i>COMMON.GLANETV.START</i> variable: <ul style="list-style-type: none"> • C for cold start mode • W for warm start mode, which is the default mode 	<i>COMMON.GLANETV.START = W</i>
Configuration directory for the NetView message provider	For the configuration directory, specify a partitioned data set (PDS) where the Log Forwarder can store some information to keep track of its progress in reading log data. Specify the data set as the value of the <i>COMMON.GLANETV.CONFIG.DIR</i> variable. The default value is <i>USER.CLIST</i> .	<i>COMMON.GLANETV.CONFIG.DIR = USER.CLIST</i>

Configuring the System Data Engine


If you want to gather System Management Facilities (SMF) data, you must authorize the IBM Common Data Provider for z Systems System Data Engine with the authorized program facility (APF), and configure

the System Data Engine to run either as a started task for streaming SMF data, or as a job for loading SMF data in batch.

Before you begin

Before you configure the System Data Engine, the following policy definition tasks, which are done in the Configuration Tool, must be complete:

1. In the Configuration Tool, create one or more policies that include one or more data streams for SMF data.

In the Configuration Tool, when you click the **Configure** icon  on a data stream node for data that is gathered by the System Data Engine, the "**Configure System Data Engine data stream**" window is shown. [“Data stream configuration for data gathered by System Data Engine” on page 143](#) lists the configuration values that you can update in this window.

2. After you configure the data streams for SMF data, click the **SYSTEM DATA ENGINE** button, which is in the Global Properties section of the **Policy Profile Edit** window, to set the configuration values for your System Data Engine environment, as described in [“SYSTEM DATA ENGINE properties: Defining your System Data Engine environment” on page 95](#).
3. [“Output from the Configuration Tool” on page 20](#) describes the output from the Configuration Tool, which includes the following System Data Engine file:

.sde file

Contains configuration information for the System Data Engine.

Procedure

To configure the System Data Engine, complete the following steps:

1. Authorize the System Data Engine with the authorized program facility (APF), as described in [“Authorizing the System Data Engine with APF” on page 80](#).
2. Complete the following steps, depending on whether you want to stream the SMF data, or load the SMF data in batch:

Option	Description
Stream SMF data	<ol style="list-style-type: none">a. Decide which method to use for collecting SMF data, as described in “Deciding which method to use for collecting SMF data” on page 80.b. Create the System Data Engine started task, as described in “Creating the System Data Engine started task for streaming SMF data” on page 80.c. If you want to collect SMF data from the SMF user exit, install the user exit, as described in “Installing the SMF user exit” on page 82.d. If you want to collect IMS data, you must write IMS records to SMF for processing by the System Data Engine. For more information, see “Collecting IMS records by using IMS User Exit” on page 85.
Load SMF data in batch	Create the job for loading SMF data in batch, as described in “Creating the job for loading SMF data in batch” on page 89 .

Tip: You must use the z/OS SYS1.PARMLIB member SMFPRMxx (or its equivalent) to enable the collection of each SMF record type that you want to gather.

Authorizing the System Data Engine with APF

For the System Data Engine to gather System Management Facilities (SMF) data, the SHBOLoad library must be authorized with the authorized program facility (APF).

About this task

To authorize the SHBOLoad library, a library name and volume ID must be in the list of authorized libraries in the PROGxx member of the SYS1.PARMLIB library.

Procedure

Use one of the following methods to authorize the SHBOLoad library:

- To include the SHBOLoad library in APF at system IPL, add the following statement to a PROGxx member:

```
APF ADD DSNAME(hlq.SHBOLoad) VOLUME(volname)
```

- To dynamically add the SHBOLoad library to APF after system IPL, issue the following MVS command:

```
SETPROG APF,ADD,DSNAME=hlq.SHBOLoad,VOLUME=volname
```

Deciding which method to use for collecting SMF data

IBM Common Data Provider for z Systems can collect System Management Facilities (SMF) data from any one of the following three sources: an SMF in-memory resource (by using the SMF real-time interface), the SMF user exit HBOSMFEX, or the SMF log stream. You must decide which method you want to use, and do the appropriate configuration for that method.

Before you begin

For more information about the SMF user exit, see [“Installing the SMF user exit” on page 82](#).

About this task

Review the following tips, and decide which method you want to use for collecting SMF data:

- If SMF is running in log stream recording mode, collect SMF data from an SMF in-memory resource by using the SMF real-time interface.

If you are running z/OS V2R1 or V2R2, APAR OA49263 must be applied to use the SMF real-time interface.

If you cannot apply APAR OA49263 to z/OS V2R1 or V2R2, use the SMF user exit to collect SMF data.
- If SMF is running in data set recording mode, consider changing the mode to log stream recording mode and collecting SMF data from an SMF in-memory resource by using the SMF real-time interface.

If you cannot run SMF in log stream recording mode, use the SMF user exit to collect SMF data.

Creating the System Data Engine started task for streaming SMF data

To have the IBM Common Data Provider for z Systems System Data Engine stream SMF data to the Data Streamer, you must create the started task for the System Data Engine by copying the sample procedure HBOSMF in the hlq.SHBOCNTL library, and updating the copy.

Procedure

To create the started task, complete the following steps:

1. Copy the procedure HBOSMF in the hlq.SHBOCNTL library to a user procedure library.

Tip: You can rename this procedure according to your installation conventions. When the name HBOSMF is used in the IBM Common Data Provider for z Systems documentation, including in the messages, it means the System Data Engine started task.

2. Update the high-level qualifier to the one for your IBM Common Data Provider for z Systems target libraries that were installed by using SMP/E.
3. To enable the zIIP offloading function to run eligible code on zIIPs, specify **ZIIPOFFLOAD=YES** in the **PARM** parameter in the EXEC statement.

```
//HBOSMFCL EXEC PGM=HBOPDE,REGION=0M,TIME=1440,
//          PARM='SHOWINPUT=NO,ZIIPOFFLOAD=YES'
```

For more information about the zIIP offloading function, see [“Offloading the System Data Engine code to z Systems Integrated Information Processors”](#) on page 90.

4. If appropriate for your environment, update the interval value (in minutes or seconds) for IBM_SDE_INTERVAL, which controls how often the System Data Engine processes data.

At regular intervals, the System Data Engine queries the appropriate sources for new data. For example, it queries one of the following sources:

- SMF in-memory resource
- Shared storage to which the SMF user exit writes
- SMF log stream

The default interval for this querying is 1 minute, and the minimum interval is 1 second. After each interval, the System Data Engine sends the new SMF records to the Data Streamer.

This collect processing interval is set on the EVERY clause of the COLLECT statement.

Guidelines for determining the interval value: Changing the interval value can affect the resource consumption of IBM Common Data Provider for z Systems. Either use the default value, which is 1 minute, or use the following guidelines to help you determine an appropriate interval value:

- Use a large interval value to minimize overhead.
- Use a small interval value to minimize memory.
- The interval value must be small enough to produce data as often as it is required by the subscriber.
- Use an interval value that is a factor of the total time in one day. [Table 10 on page 81](#) lists some example values.
- If you want to use an interval value that is equal to or greater than 60 seconds, specify that value as a whole number, and specify the time unit in minutes. For example, if you want to set the interval value to 120 seconds, instead set it to 2 minutes. [Table 10 on page 81](#) lists some example values.

<i>Table 10. Example System Data Engine interval values that are a factor of the total time in one day</i>	
Time unit	Example values
Seconds	1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20, 24, 25, 27, 30, 32, 36, 40, 45, 48, 50, 54
Minutes	1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15

5. Update the port value for IBM_UPDATE_TARGET to specify the TCP/IP port that is configured for the Data Streamer.

Tip: For more information about the Data Streamer port, see [“Configuring the Data Streamer”](#) on page 68.

6. Replace the value `/u/userid/cdpConfig/Sample1.sde` with the policy file path and name for your environment.
7. Update the value for IBM_RESOURCE to specify only one of the following values as the source from which SMF data is to be gathered.
 - The SMF in-memory resource name

- The keyword EXIT, which indicates that SMF data is to be gathered from the SMF user exit HBOSMFEX.
- The SMF log stream name

Remember: If you want to collect SMF data from the SMF user exit, you must install the user exit, as described in [“Installing the SMF user exit”](#) on page 82.

8. Verify that the user ID that is associated with the System Data Engine started task has the required authorities, as described in [“Requirements for the System Data Engine user ID”](#) on page 82.
9. Update the SAF product that is protecting your system, such as the Resource Access Control Facility (RACF), to permit the System Data Engine started task to run in your environment.

Requirements for the System Data Engine user ID

If you are collecting SMF data from an in-memory resource or log stream, the user ID that is associated with the System Data Engine started task must have authority to read the SMF in-memory resource or log stream. Also, if you are collecting SMF data from a log stream, the user ID must have update access to the RACF profile MVS.SWITCH.SMF in the OPERCMDS RACF class.

If you are collecting SMF data from the SMF user exit, there are no other requirements for the user ID.

The following information further describes the required authorities:

Authority to read the SMF in-memory resource or log stream

For example, if you are using the Resource Access Control Facility (RACF) as your System Authorization Facility (SAF) product, you must give the System Data Engine user ID read authority to the profile that you set up to secure your SMF in-memory resource or log stream. In the following examples, *IFASMF.resource* represents the name of the SMF in-memory resource or log stream that is being used to gather SMF records, and *userid* represents the System Data Engine user ID.

Tip: *IFASMF.resource* is also described in step “7” on page 81 of [“Creating the System Data Engine started task for streaming SMF data”](#) on page 80.

In-memory resource example

```
PERMIT IFA.IFASMF.resource CLASS(FACILITY) ACCESS(READ) ID(userid)
```

Log stream example

```
PERMIT IFASMF.resource CLASS(LOGSTRM) ACCESS(READ) ID(userid)
```

Update access to the RACF profile MVS.SWITCH.SMF in the OPERCMDS RACF class (only if you are collecting SMF data from a log stream)

This authority is **not** required to process data from an SMF in-memory resource.

Update access to the RACF profile MVS.SWITCH.SMF in the OPERCMDS RACF class is required only if you are collecting SMF data from a log stream so that the user ID can issue the **MVS SWITCH SMF** command. The System Data Engine periodically issues the **MVS SWITCH SMF** command to verify that it is accessing the most up-to-date data from the log stream. To grant the user ID update access to this RACF profile, issue the following commands:

```
PERMIT MVS.SWITCH.SMF CLASS(OPERCMDS) ACCESS(UPDATE) ID(userid)
SETROPTS RACLIST(OPERCMDS) REFRESH
```

Installing the SMF user exit

You can configure the IBM Common Data Provider for z Systems System Data Engine to collect System Management Facilities (SMF) data from the SMF user exit HBOSMFEX, which is provided with IBM Common Data Provider for z Systems. By using the SMF user exit, you can collect streaming SMF data independently of whether SMF is running in log stream recording mode or data set recording mode.

Before you begin

Important: The SMF types that you define on the **SYS** parameter in z/OS SYS1.PARMLIB member SMFPRMxx (or its equivalent) do not take effect if you also have **SUBSYS** parameter definitions. Therefore, if you define any subsystems, you must define the associated SMF types, and the MVS installation exits IEFU83, IEFU84, and IEFU85, for each subsystem that is specified by a **SUBSYS** parameter.

About this task

If you want to use the user exit to collect SMF data, install the HBOSMFEX user exit on the following MVS installation exits:

- IEFU83
- IEFU84
- IEFU85

For more information about MVS installation exits, see the z/OS MVS installation exits documentation.

An MVS installation exit does not receive control for records when the writing of the record is suppressed either because of a system failure or because of options that were selected at IPL time or by using the **SET SMF** command.

The HBOSMFEX module is required by the HBOSMFEX user exit and is in the SHBOLPA library.

All modules in the SHBOLPA library must be added to the system link pack area (LPA). For more information about the LPA, see the z/OS MVS initialization and tuning documentation.

Procedure

To install the SMF user exit HBOSMFEX, complete the following steps:

1. To add the load modules to an LPA, complete one of the following actions:

Action	Instruction
Add the SHBOLPA library to the pageable link pack area (PLPA) at system IPL	Add the following statement to an LPA1STxx member, but replace <i>hlq</i> with the target library high-level qualifier that is used to install IBM Common Data Provider for z Systems, and replace <i>volume</i> with the volume identifier of the data set: <pre>hlq.SHBOLPA(volume)</pre>
Add the individual modules in the SHBOLPA library to the dynamic LPA after the system IPL	Issue the following MVS system command: <pre>SETPROG LPA,ADD,MODNAME=HBOSMFEX,DSNAME=hlq.SHBOLPA</pre>

2. To install the exit, complete one of the following actions:

Action	Instruction
Install the user exit on an MVS installation exit at system IPL	Add the following statements to a PROGxx member of library SYS1.PARMLIB: <pre>EXIT ADD EXITNAME(SYS.IEFU83) MODNAME(HBOSMFEX) EXIT ADD EXITNAME(SYS.IEFU84) MODNAME(HBOSMFEX) EXIT ADD EXITNAME(SYS.IEFU85) MODNAME(HBOSMFEX)</pre>

Action	Instruction
	<p>If you specified any subsystems in an SMFPRMxx member, you must add the exit to those subsystems. For example, for the subsystem JES2, you must add the following statements:</p> <pre>EXIT ADD EXITNAME(SYSJES2.IEFU83) MODNAME(HBOSMFEX) EXIT ADD EXITNAME(SYSJES2.IEFU84) MODNAME(HBOSMFEX) EXIT ADD EXITNAME(SYSJES2.IEFU85) MODNAME(HBOSMFEX)</pre>
Dynamically install the user exit after the system IPL	<p>Issue the following MVS commands:</p> <ul style="list-style-type: none"> SETPROG EXIT,ADD,EXITNAME=SYS.IEFU83,MODNAME=HBOSMFEX SETPROG EXIT,ADD,EXITNAME=SYS.IEFU84,MODNAME=HBOSMFEX SETPROG EXIT,ADD,EXITNAME=SYS.IEFU85,MODNAME=HBOSMFEX <p>If you specified any subsystems in an SMFPRMxx member, you must define the exit to those subsystems. For example, for the subsystem JES2, you must issue the following commands:</p> <pre>SETPROG EXIT,ADD,EXITNAME=SYSJES2.IEFU83,MODNAME=HBOSMFEX SETPROG EXIT,ADD,EXITNAME=SYSJES2.IEFU84,MODNAME=HBOSMFEX SETPROG EXIT,ADD,EXITNAME=SYSJES2.IEFU85,MODNAME=HBOSMFEX</pre>

Troubleshooting tips:

- To display the status of the SMF user exit, use the following commands:

```
- D PROG,EXIT,EXITNAME=SYS.IEFU83
- D PROG,EXIT,EXITNAME=SYS.IEFU84
- D PROG,EXIT,EXITNAME=SYS.IEFU85
```

- If you need to uninstall the user exit, see [“Uninstalling the SMF user exit” on page 84](#).

Uninstalling the SMF user exit

To uninstall the SMF user exit HBOSMFEX from a system, complete this procedure.

Procedure

1. Stop the System Data Engine.
2. Remove the SMF user exit from the MVS installation exits by issuing the following MVS commands:

```
SETPROG EXIT,DELETE,EXITNAME=SYS.IEFU83,MODNAME=HBOSMFEX
SETPROG EXIT,DELETE,EXITNAME=SYS.IEFU84,MODNAME=HBOSMFEX
SETPROG EXIT,DELETE,EXITNAME=SYS.IEFU85,MODNAME=HBOSMFEX
```

3. Remove the SMF user exit from the system link pack area (LPA) by issuing the following MVS command:

```
SETPROG LPA,DELETE,MODNAME=HBOSMFEX,FORCE=YES
```

4. Stop the Data Streamer.
5. To free the 2G above-the-bar storage, and other storage spaces that are used by the SMF user exit, run the sample job HBODSPCE.

Configuring the System Data Engine for collecting IMS records

To collect IMS records, you can either write IMS records to System Management Facilities (SMF) for processing by the System Data Engine, or configure the System Data Engine to collect IMS records from IMS log data sets.

About this task

Determine which method you want to use for collecting IMS records.

If you want to stream the data in near real-time, you can install user exit and write IMS records to SMF to be processed by the System Data Engine.

If the transaction rate of your IMS system is high, and you do not want to install the user exit, you can configure the System Data Engine to collect IMS records directly from IMS log data sets. This method collects the IMS records when log switching occurs.

If you want to collect IMS Performance Analyzer Transaction Index records, use the HBOPIMS utility provided by IBM Common Data Provider for z Systems.

Collecting IMS records by using IMS User Exit

To collect IBM Information Management System (IMS) log data, you can write IMS records to System Management Facilities (SMF) for processing by the System Data Engine.

Before you begin

Before you complete the steps in this procedure, you must complete the configuration steps for SMF data collection. For example, in the System Data Engine started task, verify that the COLLECT statement specifies the correct source of the SMF records (for example, the SMF in-memory resource or the SMF user exit). Also, if you are collecting SMF data by using the SMF user exit, install the SMF user exit, as described in [“Installing the SMF user exit” on page 82](#).

About this task

By using IMS User Exit, all IMS log records **except for** IMS Performance Analyzer Transaction Index records can be written to SMF. For more information about collecting IMS Performance Analyzer Transaction Index records, see [“Collecting IMS Performance Analyzer Transaction Index records” on page 88](#).

Procedure

To write IMS records to SMF for processing by the System Data Engine, complete the following configuration steps:

1. Update the z/OS SYS1.PARMLIB member SMFPRMxx (or its equivalent) to enable the collection of SMF record type 127.
2. If you are collecting SMF data from the SMF in-memory resource, create a new, or update an existing, SMF in-memory resource to include SMF record type 127.

Important: Do not include SMF record type 127 in any SMF log stream definitions.

3. Depending on the type of IMS log records that you want to collect, choose one or both of the following methods for writing IMS log records to SMF, and complete the associated configuration steps:

Option	Description
IMS LOGWRT user exit	For writing all IMS log records, except for IMS Performance Analyzer Transaction Index records, install the IMS LOGWRT user exit, as described in “Installing the IMS LOGWRT user exit” on page 86 .
HBOPIMS utility	For writing IMS Performance Analyzer Transaction Index records, run the HBOPIMS utility, as described in “Collecting IMS Performance Analyzer Transaction Index records” on page 88 .

Installing the IMS LOGWRT user exit

IBM Common Data Provider for z Systems provides the IMS LOGWRT user exit to write IMS log records to SMF. The System Data Engine reads the IMS log records either from an SMF in-memory resource or from storage that is created by the SMF user exit.

Before you begin

The IMS LOGWRT user exit supports IMS Version 13 or later.

Procedure

To install the user exit, complete the steps that apply for your installation option.

Installation option	Steps
IMS multi-user exit	<ul style="list-style-type: none">a. Add the <i>hlq</i>.SHBOLOAD data set to the STEPLIB concatenation of the IMS Control Region.b. Add the following LOGWRT user exit definition to the IMS PROCLIB member DFSDFxxx:<div style="background-color: #f0f0f0; padding: 5px;"><pre>EXITDEF=(TYPE=LOGWRT,EXITS=(HBOFLGX0))</pre></div>c. After the IMS Control Region JCL is updated, recycle the IMS system to activate the LOGWRT user exit.
IMS tools	<p>If IMS tools are implemented for the IMS environment, install the LOGWRT user exit by using the distributed module HBOFLGX0 that is in the SHBOLOAD library. This module is specified as EXITNAME (HBOFLGX0). IMS Tools does not require the load library to be inserted into the IMS Control Region STEPLIB JCL.</p> <ul style="list-style-type: none">a. Add an IMS tools user exit definition to the IMS PROCLIB member GLXEXIT0, as shown in the following example:<div style="background-color: #f0f0f0; padding: 5px;"><pre>EXITDEF (TYPE (LOGR) EXITNAME (HBOFLGX0) LOADLIB (hlq.SHBOLOAD))</pre></div>b. To activate the LOGWRT user exit, recycle the IMS system.
Stand-alone exit	<p>The SHBOLOAD library contains member DFSFLGX0, which is the member name that IMS searches for during startup. The DFSFLGX0 module loads HBOLGX?0, which writes IMS log records to SMF.</p> <ul style="list-style-type: none">a. Add the SHBOLOAD library to the STEPLIB concatenation of the IMS Control Region, and verify that the DFSFLGX0 module in this library is concatenated before any other module of the same name.b. After the IMS Control Region JCL is updated, recycle the IMS system to activate the LOGWRT user exit.

When the LOGWRT user exit initializes successfully, the following message is written to the z/OS console:

```
HB08101I CDP IMS LOGWRT EXIT ACTIVATED FOR IMSID=iii
```

Collecting IMS records without using IMS User Exit

If the transaction rate of your IMS system is high, and you do not want to install the IMS LOGWRT user exit, you can use the System Data Engine to collect IMS records directly from IMS log data sets.

About this task

You can collect IMS log records directly from IMS log data sets by using the following methods:

A separate System Data Engine started task

You can use a separate System Data Engine started task to monitor the IMS online log data set (OLDS) status and collect the IMS log records when log switching occurs.

A System Data Engine batch job

You can use a System Data Engine batch job to load IMS log records from the IMS log data sets that are specified in the HBOLG DD statement.

Creating the System Data Engine started task for collecting IMS records

To have the IBM Common Data Provider for z Systems System Data Engine stream IMS data to the Data Streamer without using the IMS User Exit, you must create the started task for the System Data Engine by copying the sample procedure HBOIMS in the *hlq.SHBOCNTL* library, and updating the copy.

Procedure

1. Copy the procedure HBOIMS in the *hlq.SHBOCNTL* library to a user procedure library.
2. Update *HB0vrm* to the high-level qualifier for your IBM Common Data Provider for z Systems target libraries that were installed by using SMP/E.
3. Update *IMSvrm* to the high-level qualifier for your IMS target libraries that were installed by using SMP/E.

Tip: If multiple IMS versions are running in the same LPAR, update *IMSvrm* to the high-level qualifier for the target libraries of the lowest IMS version.

4. To enable the zIIP offloading function to run eligible code on zIIPs, specify **ZIIPOFFLOAD=YES** in the **PARM** parameter in the EXEC statement.

```
//HBOSMFCL EXEC PGM=HBOPDE,REGION=0M,TIME=1440,  
//          PARM='SHOWINPUT=NO,ZIIPOFFLOAD=YES'
```

For more information about the zIIP offloading function, see [“Offloading the System Data Engine code to z Systems Integrated Information Processors”](#) on page 90.

5. If appropriate for your environment, update the interval value (in minutes or seconds) for *IBM_SDE_INTERVAL*, which controls how often the System Data Engine processes data.
At regular intervals, the System Data Engine queries the OLDS data set status from the IMS system. Align the interval with the time when IMS OLDS data set switching usually occurs.
This collect processing interval is set on the EVERY clause of the COLLECT statement.
6. Update the port value for *IBM_UPDATE_TARGET* to specify the TCP/IP port that is configured for the Data Streamer.
7. Replace the value */etc/cdpConfig/hboin.sde* with the policy file path and name for your environment.
8. Verify that the user ID that is associated with the System Data Engine started task has the required authorities as described in [“Requirements for the System Data Engine user ID for collecting IMS records”](#) on page 87.

Requirements for the System Data Engine user ID for collecting IMS records

If you are collecting data from IMS, the user ID that is associated with the System Data Engine started task must have authority to read the IMS RECON and online log data sets (OLDS). Also, if authorization control for DBRC API requests is established, the user ID must have read access to the security resource profiles for the STARTDBRC, STOPDBRC, and QUERY TYPE=OLDS API requests.

The following information further describes the required authorities:

Authority to read the RECON data sets and Online Log data sets (OLDS)

For example, if you are using the Resource Access Control Facility (RACF) as your System Authorization Facility (SAF) product, you must give the System Data Engine user ID read authority to the profiles for the IMS RECON and online log data sets.

```
PERMIT hlq.RECON* CLASS(DATASET) ACCESS(READ) ID(userid)  
PERMIT hlq.OLP* CLASS(DATASET) ACCESS(READ) ID(userid)
```

hlq is the high-level qualifier of the RECON and online log data sets.

Authority to issue the DBRC API requests

For example, if you are using the Resource Access Control Facility (RACF) to protect the DBRC API requests, you must give the System Data Engine user ID read authority to the following security resource profiles.

```
PERMIT hlq.STDBRC CLASS(FACILITY)
ACCESS(READ) ID(userid)
PERMIT hlq.LIST.LOG.ALLOLDS CLASS(FACILITY)
ACCESS(READ) ID(userid)
```

hlq is the high-level qualifier of the resource name.

Creating the System Data Engine batch job for loading IMS log data

Create a batch job for the System Data Engine to run in batch mode so that the output is written to a file instead of being streamed to the Data Streamer. You can create this job based on the sample job HBOJBIMS in the *hlq*.SHBOCNTL library.

Procedure

1. Copy the sample job HBOJBIMS from *hlq*.SHBOCNTL to a user job library.
2. Update the job card according to your environment.
3. To enable the zIIP offloading function to run eligible code on zIIPs, specify **ZIIPOFFLOAD=YES** in the **PARM** parameter in the EXEC statement.

```
//HBOBMFCL EXEC PGM=HBOPDE,REGION=0M,TIME=1440,
//          PARM='SHOWINPUT=NO,ZIIPOFFLOAD=YES'
```

For more information about the zIIP offloading function, see [“Offloading the System Data Engine code to z Systems Integrated Information Processors”](#) on page 90.

4. Update the following STEPLIB DD statement to specify the *hlq*.SHBOLOAD data set.

```
//STEPLIB DD DISP=SHR,DSN=HBOvrm.SHBOLOAD
```

5. Update the following HBOIN DD statements to specify the *hlq*.SHBODEFS data set members.

```
//HBOIN DD DISP=SHR,DSN=HBOvrm.SHBODEFS(HBOCCSV)
//      DD DISP=SHR,DSN=HBOvrm.SHBODEFS(HBOLLSMF)
//      DD DISP=SHR,DSN=HBOvrm.SHBODEFS(HBORSIMS)
//      DD DISP=SHR,DSN=HBOvrm.SHBODEFS(HBOUSIMS)
```

6. For each IMS log record type that you want to collect, create a DD statement to receive the output. Change *IMSxxxx* to the DD name of the IMS log record type. Refer to the SET IBM_FILE statements in the *hlq*.SHBODEFS(HBOUSIMS) member for the output DD names of IMS log record types.

```
//IMSxxxx DD SYSOUT=*,RECFM=V,LRECL=32756
```

The following example shows the DD statements for receiving the output for IMS log record types x07 and x08.

```
/* Sample COLLECT statement for processing log stream data
/*
// DD *
COLLECT IMS
  COMMIT AFTER END OF FILE;
/*
//HBOLOG DD DISP=SHR,DSN=stored.imsdata
//HBOOUT DD SYSOUT=*
//HBODUMP DD SYSOUT=*
//IMS07 DD SYSOUT=*, RECFM=V,LRECL=32756 for record type 07
//IMS08 DD SYSOUT=*, RECFM=V,LRECL=32756 for record type 08
```

Collecting IMS Performance Analyzer Transaction Index records

IMS Performance Analyzer batch reporting can create specialized extract files for IMS Transaction Index and IMS Connect Transaction Index records. IBM Common Data Provider for z Systems provides the

HBOPIMS utility for reading IMS Transaction Index and IMS Connect Transaction Index records from the extract files and writing the records to SMF for processing by the System Data Engine.

Procedure

To write the IMS Transaction Index records (x'CA01') or IMS Connect Transaction Index records (x'CA20') to SMF record type 127, subtype 1000, customize and run the HBOJIMS JCL in the *hlq.SHB0CNTL* data set on the system where the System Data Engine is running.

The comments in the JCL job include instructions for customizing and running the job.

Creating the job for loading SMF data in batch

To run the IBM Common Data Provider for z Systems System Data Engine in batch mode so that it writes its output to a file, rather than streaming it to the Data Streamer, you must create the job for loading SMF data in batch. You can create this job by using the sample job HBOJBCOL in the *hlq.SHB0CNTL* library, and updating the copy.

Procedure

To create the job, complete the following steps:

1. Copy the job HBOJBCOL in the *hlq.SHB0CNTL* library to a user job library.
2. Update the job card according to your site standards.
3. To enable the zIIP offloading function to run eligible code on zIIPs, specify **ZIIPOFFLOAD=YES** in the **PARM** parameter in the EXEC statement.

```
//HBOSMFCL EXEC PGM=HBOPDE,REGION=0M,TIME=1440,  
//          PARM='SHOWINPUT=NO,ZIIPOFFLOAD=YES'
```

For more information about the zIIP offloading function, see [“Offloading the System Data Engine code to z Systems Integrated Information Processors”](#) on page 90.

4. Update the following STEPLIB DD statement to refer to the *hlq.SHB0LOAD* data set:

```
//STEPLIB DD DISP=SHR,DSN=HBOvrm.LOAD
```

5. For each SMF record type that you want to collect, update the following control statements, which are provided by the HBOIN DD statement, and change the variable *nnn* to the appropriate SMF record type value, for example, 030, 080, or 110.

```
//HBOIN DD DISP=SHR,DSN=HBOvrm.SHB0DEFS(HBOCCSV)  
// DD DISP=SHR,DSN=HBOvrm.SHB0DEFS(HBOLLSMF)  
// DD DISP=SHR,DSN=HBOvrm.SHB0DEFS(HBORSnnn)  
// DD DISP=SHR,DSN=HBOvrm.SHB0DEFS(HBOUSnnn)
```

Most of the control statements that are required to run the System Data Engine are provided in the *hlq.SHB0DEFS* data set and must not be changed.

Each member in the HBOIN DD concatenation specifies a task that the System Data Engine must do. The last statement in the HBOIN DD concatenation must be a COLLECT control statement, which initiates the processing of the input data by the System Data Engine.

The following example shows the control statements for SMF record types 80 and SMF_110_1_KPI:

```
//* CONTROL STATEMENTS  
/*  
//HBOIN DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBOCCSV)  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBOLLSMF)  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBOTCIFI) for type 110_1_KPI  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBORS110) for type 110_1_KPI  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBOU110I) for type 110_1_KPI  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBORS080) for type 80  
// DD DISP=SHR,DSN=hlq.m1q.SHB0DEFS(HBOUS080) for type 80
```

6. For each SMF record type that you specify for collection, add a DD statement, such as the following statement, to receive the output, and change the variable *nnn* to the appropriate SMF record type value, for example, 030, 080, or 110.

```
//SMFnnn DD SYSOUT=*,RECFM=VB,LRECL=32756
```

The following example shows the DD statements for receiving the output for SMF record types 80 and SMF_110_1_KPI:

```
/* Sample COLLECT statement for processing log stream data
/*
/* DD *
COLLECT SMF
  COMMIT AFTER END OF FILE;
/*
//HBOLOG DD DISP=SHR,DSN=stored.smfdata
//HBOOUT DD SYSOUT=*
//HBODUMP DD SYSOUT=*
//SMF080 DD SYSOUT=* for type 80
//SMF110 DD SYSOUT=* for type 110_1_KPI
//SMF11001 DD SYSOUT=* for type 110_1_KPI
//SMF110FC DD SYSOUT=* for type 110_1_KPI
//SMF110TX DD SYSOUT=* for type 110_1_KPI
//SMF1101I DD SYSOUT=* for type 110_1_KPI
```

Offloading the System Data Engine code to z Systems Integrated Information Processors

The System Data Engine can offload most of the code to run on z Systems Integrated Information Processors (zIIPs). This operation frees the general-purpose processors (GCPs) for other work. It can also reduce software licensing costs.

About this task

Not all System Data Engine code is eligible to run on zIIPs. The System Data Engine must switch to task mode for the portion of code that must run in task mode, and switch back to Service Request Block (SRB) when that portion of code finishes running. The synchronization between task and enclave SRB creates additional overhead at the address space level. As a result, the total CPU time (GCPs plus zIIPs) when zIIP offloading is enabled is slightly higher than the total CPU time when zIIP offloading is not enabled. Therefore, if the zIIP capacity on the logical partition is insufficient, do not enable the zIIP offloading function.

Procedure

- To activate the zIIP offloading function, specify **ZIIPOFFLOAD=YES** in the **PARM** parameter in the EXEC statement of the System Data Engine started task procedure or batch job JCL as shown in the following example.

```
//HBOSMFCL EXEC PGM=HBOPDE,REGION=0M,TIME=1440,
//          PARM='SHOWINPUT=NO,ZIIPOFFLOAD=YES'
//STEPLIB DD DISP=SHR,DSN=HBOvsm.LOAD
//HBOIN DD *
SET IBM_SDE_INTERVAL = '1 MINUTES';
SET IBM_UPDATE_TARGET = 'PORT 61001';
SET IBM_FILE_FORMAT = 'CSV';
SET IBM_RESOURCE = 'IFASMF.SYS01.PERF';
//          DD PATH='/etc/cdpConfig/SYS01.sde',
//          PATHDISP=(KEEP),RECFM=V,LRECL=255,FILEDATA=RECORD
//          DD *
COLLECT SMF FROM &IBM_RESOURCE
  EVERY &IBM_SDE_INTERVAL;
/*
//HBOOUT DD SYSOUT=*
//HBODUMP DD SYSOUT=*
```

If **ZIIPOFFLOAD=YES** is specified but no zIIPs are online when the System Data Engine address space is started, no work is offloaded to zIIPs.

The Resource Measurement Facility (RMF) provides information on zIIP usage to help you identify when to add more zIIPs to the logical partition. Also, fields in SMF Type 30 records allow you to know how much time is spent on zIIPs, and how much time is spent on running zIIP eligible work on GCPs. A high CPU time consumed on GCPs by work that is eligible for a zIIP indicates high contention on the zIIP processors. In this case, you must add more zIIPs to the logical partition, or specify **ZIIPOFFLOAD=NO** to disable the zIIP offloading function of the System Data Engine.

Verifying the search order for the TCP/IP resolver configuration file

Before you start IBM Common Data Provider for z Systems, verify that the z/OS environment is set up correctly so that IBM Common Data Provider for z Systems can access the TCP/IP resolver configuration file.

About this task

The IBM Common Data Provider for z Systems Data Streamer and Log Forwarder are z/OS UNIX System Services programs. They use TCP/IP functions that require access to the TCP/IP resolver configuration file. This access is provided by using a resolver search order. The resolver search order for z/OS UNIX System Services programs is documented in the topic about resolver configuration files in the *z/OS Communications Server: IP Configuration Guide*.

The following list summarizes the resolver search order:







1. GLOBALTCPIPDATA statement
 2. The *RESOLVER_CONFIG* environment variable in the Data Streamer procedure or job and in the Log Forwarder properties (which are part of the global properties that you can define for data streams in a policy).
- Tip:** For information about this environment variable configuration, see the following topics:
- [“Configuring the Data Streamer” on page 68](#)
 - [“Log Forwarder properties configuration” on page 93](#)
3. */etc/resolv.conf* file
 4. SYSTCPD DD statement in the Log Forwarder and Data Streamer started tasks
 5. *userid.TCPIP.DATA*, where *userid* is the user ID that is associated with the Log Forwarder and Data Streamer started tasks
 6. SYS1.TCPPARMS(TCPDATA)
 7. DEFAULTTCPIPDATA
 8. TCPIP.TCPIP.DATA

Procedure

Verify that the resolver configuration file is available to the Data Streamer and the Log Forwarder by using one of the search order mechanisms.

Configuration reference for managing policies

This reference contains information that is useful in creating and updating policies. It includes information about the global properties that you can define for a policy, the icons on each node in a policy, the correlation between SMF record types and the associated SMF data stream names, and the configuration values that you can update for each data stream, transform, or subscriber.

Table 11. Configuration reference information for managing policies	
Reference information	Area of Configuration Tool where the relevant configuration is done
“Global properties that you can define for all data streams in a policy” on page 92	Policy Profile Edit window in the Global Properties section
“Icons on each node in a policy” on page 115	Policy Profile Edit window in the graph
“SMF data stream reference” on page 97	Window that is shown when you click the Add Data Stream icon  DATA STREAM in the Policy Profile Edit window
“SMF_110_1_KPI data stream content” on page 111	Window that is shown when you click the Add Data Stream icon  DATA STREAM in the Policy Profile Edit window
“Data stream configuration for data gathered by Log Forwarder” on page 116	Window that is shown when you click the Configure icon  on a data stream node for data that is gathered by the Log Forwarder
“Data stream configuration for data gathered by System Data Engine” on page 143	Window that is shown when you click the Configure icon  on a data stream node for data that is gathered by the System Data Engine
“Transform configuration” on page 143	Window that is shown when you click the Transform icon  on a data stream or transform node
“Subscriber configuration” on page 147	Window that is shown when you click a Subscribe icon  on a data stream or transform node

Global properties that you can define for all data streams in a policy

When you create or edit a policy in the IBM Common Data Provider for z Systems Configuration Tool, the buttons **SYSTEM**, **LOG FORWARDER**, **SYSTEM DATA ENGINE**, and **SCHEDULES** are shown in the Global Properties section of the **Policy Profile Edit** window. You can use these buttons to define properties that apply to all data streams (or all data streams from a certain type of data gatherer) in the policy.

Tips:

- The **LOG FORWARDER** button is available only after you define a data stream for z/OS log data, which is gathered by the Log Forwarder. Use this button to set, or verify, the Log Forwarder properties.
- The **SYSTEM DATA ENGINE** button is available only after you define a data stream for SMF or IMS data, which is gathered by the System Data Engine. Use this button to set, or verify, the System Data Engine properties.

SYSTEM properties: Defining alternative host names for source systems

When you create or edit a policy in the IBM Common Data Provider for z Systems Configuration Tool, you can use the **SYSTEM** button to define alternative host names for the source systems from which IBM Common Data Provider for z Systems collects data. The Data Streamer then uses these alternative host names in the associated data records that it sends to subscribers.

About this task

Example of using alternative host names

If the host name for a source system is `abc.host.com`, and you define an alternative host name of `def.host.com` for this source system, the Data Streamer changes the host name in the associated data records to `def.host.com` before it sends the records to the subscriber.

Reasons why you might want to define alternative host names

The host name for a source system can sometimes change. If you know, for example, that a source system interchangeably uses `ghi.host.com` and `jkl.host.com` as its host name, you can define `ghi.host.com` to be the alternative host name for `jkl.host.com`. Then, the host name is always reported as `ghi.host.com` so that the data at the target destination can easily be correlated to the correct source system.

You might also have other reasons for defining alternative host names.

Procedure

To define alternative host names, complete the following steps:

1. In the Global Properties section of the **Policy Profile Edit** window, click **SYSTEM**.
2. Click **ADD SYSTEM**.
3. In the **System name** field, type the current host name.
4. In the **Remapped host name** field, type the alternative host name.
5. Repeat steps “2” on page 93 to “4” on page 93 for each source system for which you want to define alternative host names.
6. Click **OK**.

LOG FORWARDER properties: Defining your Log Forwarder environment

In the IBM Common Data Provider for z Systems Configuration Tool, after you define a data stream for z/OS log data (which is gathered by the Log Forwarder), use the **LOG FORWARDER** button to set the configuration values for your Log Forwarder environment.

About this task

For more information about the Log Forwarder configuration values, see [“Log Forwarder properties configuration” on page 93](#).

For more information about configuring the Log Forwarder, see [“Configuring the Log Forwarder” on page 72](#).

Procedure

To define your Log Forwarder environment, complete the following steps:

1. In the Global Properties section of the **Policy Profile Edit** window, click **LOG FORWARDER**.
2. In the **"Configure Log Forwarder properties"** window, update the configuration values for your environment, and click **OK**.

Log Forwarder properties configuration

This reference lists the configuration values that you can update in the **"Configure Log Forwarder properties"** window of the IBM Common Data Provider for z Systems Configuration Tool.

Port

The port on which the Data Streamer listens for data from the Log Forwarder.

Tip: For more information about the Data Streamer port, see [“Configuring the Data Streamer” on page 68](#).

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams.

The value must be an integer in the range 1 - 5. The default value is 1.

Pattern Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for new data sources that match wildcard specifications. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams.

The value must be an integer in the range 0 - 60. The default value is 1.

JRELIB

The fully qualified path to a set of native libraries that are required by the Java Runtime Environment (31-bit). The default value is `/usr/lib/java_runtime`.

JRELIB64

The fully qualified path to a set of native libraries that are required by the Java Runtime Environment (64-bit). The default value is `/usr/lib/java_runtime64`.

REGJAR

The fully qualified path to the `ifaedjreg.jar` file, which provides access to z/OS product registration services. The default value is `/usr/include/java_classes/ifaedjreg.jar`.

RESOLVER_CONFIG

The TCP/IP resolver configuration file that the Log Forwarder must use.

The Log Forwarder is a z/OS UNIX System Services program. It uses TCP/IP functions that require access to the TCP/IP resolver configuration file.

For more information, see [“Verifying the search order for the TCP/IP resolver configuration file” on page 91](#).

TZ

The time zone offset for the Log Forwarder and all data streams from the Log Forwarder.

ZLF_JAVA_HOME

The Java installation directory.

ZLF_HOME

The Log Forwarder installation directory.

ZLF_WORK

The Log Forwarder working directory, which contains files that are created and used during the operation of the Log Forwarder. For example, it includes files that contain information about the state of the Log Forwarder and its progress in collecting data.

Guidelines for the working directory

Use the following guidelines to help you decide which directory to use as the working directory:

- The working directory must be in a different physical location from the working directory for any other Log Forwarder instance.
- The directory must be readable and writable by the user ID that runs the Log Forwarder.
- To avoid possible conflicts, do not use a directory that is defined as the Configuration Tool working directory.

Important: Do not update, delete, or move the files in the Log Forwarder working directory.

ZLF_LOG

The directory for the `logging.properties` file.

ZLF_WAS_PLUGINS_ROOT

The IBM WebSphere Application Server installation root directory for Web Server Plug-ins. This directory contains the `com.ibm.hpel.logging.jar` file that is used to retrieve log data from High Performance Extensible Logging (HPEL).

ZLF_GATHERER

The directory for use by data gatherers from a third party organization.

Transport Affinity (environment variable `_BPXK_SETIBMOPT_TRANSPORT`)

The TCP/IP stack to which the Log Forwarder must have affinity. If no value is specified, the Log Forwarder has affinity to the default TCP/IP stack.

SYSTEM DATA ENGINE properties: Defining your System Data Engine environment

In the IBM Common Data Provider for z Systems Configuration Tool, after you define a data stream for SMF data (which is gathered by the System Data Engine), use the **SYSTEM DATA ENGINE** button to set the configuration values for your System Data Engine environment.

About this task

For more information about configuring the System Data Engine, see [“Configuring the System Data Engine”](#) on page 78.

Procedure

1. In the Global Properties section of the **Policy Profile Edit** window, click **SYSTEM DATA ENGINE**.
2. In the **"Configure System Data Engine properties"** window, update the following configuration values for your environment, and click **OK**.

USER Concatenation

This value is relevant only if you are using custom System Data Engine data streams. It is required as part of enabling the Configuration Tool to support custom System Data Engine data streams. For more information, [“Creating a System Data Engine data stream definition”](#) on page 33.

The value must be the name of the USERDEFS data set that contains the custom System Data Engine definitions. This data set is also referenced in the `concat.s.json` file, which is in the working directory for the IBM Common Data Provider for z Systems Configuration Tool.

CDP Concatenation

This value must be the name of the SHBODEFS data set that is installed with IBM Common Data Provider for z Systems in your environment. This data set is also referenced in the `concat.s.json` file, which is in the working directory for the IBM Common Data Provider for z Systems Configuration Tool.

IZOA Concatenation

This value is relevant only if you are using IBM Z Operations Analytics. It is required as part of enabling the Configuration Tool to support SMF data that is destined for IBM Z Operations Analytics. For more information, see the [IBM Z Operations Analytics documentation](#).

The value must be the name of the SGLASAMP data set that is installed with IBM Z Operations Analytics in your environment. This data set is also referenced in the `concat.s.json` file, which is in the working directory for the IBM Common Data Provider for z Systems Configuration Tool.

Tip: The `concat.s.json` file is created in the Configuration Tool working directory when you save the first policy that you create. By default, any new policies that are created use the same `concat.s.json` file.

SCHEDULES properties: Defining time intervals for filtering operational data

When you create or edit a policy in the IBM Common Data Provider for z Systems Configuration Tool, you can use the **SCHEDULES** button to define time intervals for filtering the operational data that IBM Common Data Provider for z Systems collects. For example, you might want to define time intervals to filter data according to the expected peak demand for your applications.

About this task

To define a time interval for filtering the data, you must first define a schedule, which can contain one or more time interval definitions. You can define multiple schedules.

Important: The schedules that you define are used in filtering data streams only if, when you configure the data streams, you select the **Time Filter** transform in the "**Transform data stream**" window. For more information about transform types, see [“Transform configuration” on page 143](#).

Procedure

- In the Global Properties section of the **Policy Profile Edit** window, click **SCHEDULES**, and complete one or more of the following actions, depending on what you want to do.

Any previously defined schedules are shown in the **Schedule** list.

Action	Instruction
Create or edit a schedule	<p>To edit a schedule, select it from the Schedule list.</p> <p>To define a new time interval in a schedule, click ADD, and complete the following steps:</p> <ol style="list-style-type: none">1. In the Edit name field, type the name for the schedule that you want to contain this time interval.2. To set the time interval for this schedule, either type the time information in the From and to fields, or use the slider to adjust the time.3. To add another time interval for this schedule, click ADD WINDOW, and repeat the previous step.4. To save the schedule, click APPLY.
Delete a schedule	<p>Select the schedule from the Schedule list, and click DELETE.</p> <p>Restriction: The DELETE button is not available if a schedule is assigned to a transform for a data stream.</p>

Groups of data streams in the Configuration Tool

This reference lists and describes the data stream groups in the "**Select data stream**" window. In the window, you can expand and select data streams from these groups: Starter Sets, Common Data Provider for z Systems, IBM Z Operations Analytics and Custom Data Streams.

Starter Sets

Starter Sets includes commonly used data streams. These data streams are categorized into various data stream units based on some z/OS components and subsystems. This group includes one subgroup: Common Data Provider for z Systems. You can select basic sets of data streams as a unit.

Table 12. Subgroup and data streams of Starter Sets		
Subgroup	Data stream unit	Description of data stream unit
Common Data Provider for z Systems	General z/OS system monitoring	z/OS system log data, common address space work and RMF CPU activity
	Security	z/OS system log data and RACF processing
	CICS	CICS MSGUSR and EYULOG log information and information about key performance indicators (KPIs) for CICS Transaction Server for z/OS monitoring
	WebSphere Application Server	WAS Request Activity and Data from the SYSOUT and SYSPRINT job log
	Db2	Db2 Statistics - system services, database services, Dynamic ZPARMS, Buffer Manager Group Buffer Pool, System Storage Usage and aggregated accounting statistics
	IMS	IMS and IMS CPI-CI program start and termination

Common Data Provider for z Systems

This group includes all the data streams that IBM Common Data Provider for z Systems supports.

IBM Z Operations Analytics

This group includes all the data streams that IBM Z Operations Analytics supports.

Custom Data Streams

This group includes the data streams that are customized through System Data Engine language.

SMF data stream reference

For each System Management Facilities (SMF) record type, this reference lists the name of the data stream that IBM Common Data Provider for z Systems uses to collect the data and includes a brief description of the data stream content. In the Configuration Tool, these SMF data stream names are shown in the "**Select data stream**" window, which opens when you click the **Add Data Stream** icon

 **DATA STREAM** in the **Policy Profile Edit** window.

Table 13 on page 98 provides the following information:

Column 1

The SMF record type

Column 2

The subtype of the SMF record. In either of the following situations, no subtype is indicated:

- The stream applies to all subtypes of the respective SMF record.
- The respective SMF record has no subtypes.

Column 3

The name of the data stream to which the SMF data is written

Column 4

A brief description of the content of the SMF data stream

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data			
Type	Subtype	Data stream name	Description of data stream content
0		SMF_000	IPL
2		SMF_002	Dump header
3		SMF_003	Dump trailer
4		SMF_004	Step termination
		SMF_004_DEVICE	Step termination device data
5		SMF_005	Job termination
		SMF_005_ACCOUNTING	Job termination accounting data
6		SMF_006	JES2/JES3/PSF/External writer
7		SMF_007	SMF data lost
8		SMF_008	I/O configuration at IPL
		SMF_008_ONLINE	Data for online devices at IPL
9		SMF_009	VARY device ONLINE
		SMF_009_DEVICE	Data for each device varied online
10		SMF_010	Allocation recovery
		SMF_010_DEVICE	Data for each device made available
11		SMF_011	VARY device OFFLINE
		SMF_011_DEVICE	Data for each device varied offline
14		SMF_014	INPUT or RDBACK data set activity
		SMF_014_UCB	UCB information
15		SMF_015	OUTPUT, UPDAT, INOUT, or OUTIN data set activity
		SMF_015_UCB	UCB information
16		SMF_016	DFSORT statistics
		SMF_016_SORTIN	SORTIN data set information
		SMF_016_SORTOUT	SORTOUT data set information
		SMF_016_OUTFIL	OUTFIL data set information
17		SMF_017	Scratch data set status
		SMF_017_VOLUMEXT	Volume information
18		SMF_018	Rename data set status
		SMF_018_VOLUMEXT	Volume information
19		SMF_019	Direct access volume
20		SMF_020	Job initiation
		SMF_020_ACCOUNTING	Job accounting information
21		SMF_021	Error statistics by volume
22		SMF_022	Configuration
23		SMF_023	SMF status
24		SMF_024	JES2 spool offload
		SMF_024_PRODUCT	JES2 product information
		SMF_024_SPOOLOFF	Statistics for spool offload devices
		SMF_024_JOBSEL	Job selection criteria
		SMF_024_SYSEL	SYSOUT selection criteria
		SMF_024_SYSAFF	System affinity information
25		SMF_025	JES3 device allocation
26		SMF_026	JES2/JES3 job purge

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
28		SMF_028	NPM statistics
30		SMF_030	Common address space work
		SMF_030_EXCP	I/O information for a specific DD Name/Device address pair for the address space
		SMF_030_ACCOUNTING	User accounting information for the address space
		SMF_030_OPENMVS	z/OS UNIX process information
		SMF_030_ARM	Information related to a batch job or started task that registers as an element of automatic restart management
		SMF_030_USAGE	Product ID information and usage data
		SMF_030_ENCLAVE	Remote system data for each system that executed work under a multisystem enclave
		SMF_030_COUNTER	Hardware Instrumentation Services (HIS) counters
		SMF_030_ZEDC	zEDC usage statistics section
31		SMF_031	TIOC initialization
32		SMF_032	TSO user work accounting
		SMF_032_IDENTIF	Identification section
		SMF_032_TSOCOMMAND	TSO/E command segment
33	1	SMF_033	APPC/MVS TP accounting
		SMF_033_ACS	TP usage accounting
		SMF_033_TPS	TP usage scheduler data
34		SMF_034	TS-step termination
		SMF_034_DEVICE	EXCP section
35		SMF_035	LOGOFF
		SMF_035_ACCOUNT	Accounting information
36		SMF_036	Integrated Catalog Facility Catalog
37		SMF_037_HW	NetView Hardware Monitor
		SMF_037_ETHERNET	Ethernet LAN data
		SMF_037_TEXT	Text message data
39	1 - 7	SMF_039_1_TO_7	NetView Session Monitor
	8	SMF_039_8	NetView Session Monitor
40		SMF_040	Dynamic DD
		SMF_040_DEVICE	EXCP section
41	1 - 3	SMF_041	Data-in-virtual Access/Unaccess
		SMF_041_VLF	VLF statistics

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
42	1	SMF_042_1	DFSMS - BMF performance statistics
		SMF_042_STOR_CLASS	Storage class summary
	2	SMF_042_2	DFSMS - DFP cache control unit statistics
		SMF_042_UNIT_CACHE	Control unit cache section
		SMF_042_VOL_STATUS	Volume status section
	3	SMF_042_3	DFSMS - DFP SMS configuration statistics
		SMF_042_EVNT_AUDIT	Event audit section
	4	SMF_042_4	DFSMS - DFP concurrent copy session statistics
		SMF_042_CONC_COPY	Concurrent copy session section
		SMF_042_EAVCC_VOL	EAV concurrent copy volume section
	5	SMF_042_5	DFP Storage Class statistics
		SMF_042_STOR_RESP	Storage class response time
	6	SMF_042_6	DFP Data Set statistics
		SMF_042_6_X	DFP Data Set statistics from record procedure
43	11	SMF_042_11	DFP Extended Remote Copy (XRC) Session Statistics
	14	SMF_042_14	ADSM Server statistics
43	2	SMF_043_JES2	JES2 start
	5	SMF_043_JES3	JES3 start
45	2	SMF_045_JES2	JES2 withdrawal
	5	SMF_045_JES3	JES3 stop
47	2	SMF_047_JES2	JES2 SIGNON/start line (BSC only)
	5	SMF_047_JES3	JES3 SIGNON/start line/LOGON
48	2	SMF_048_JES2	JES2 SIGNOFF/stop line (BSC only)
	5	SMF_048_JES3	JES3 SIGNOFF/stop line/LOGOFF
49	2	SMF_049_JES2	JES2 integrity (BSC only)
	5	SMF_049_JES3	JES3 integrity
50		SMF_050	ACF/VTAM® tuning statistics
52		SMF_052	JES2 LOGON/start line (SNA only)
53		SMF_053	JES2 LOGOFF/stop line (SNA only)
54		SMF_054	JES2 integrity (SNA only)
55		SMF_055	JES2 network SIGNON
56		SMF_056	JES2 network integrity
57	2	SMF_057_JES2	JES2 network SYSOUT transmission
	5	SMF_057_JES3	JES3 networking transmission
58		SMF_058	JES2 network SIGNOFF
59		SMF_059	MVS/BDT file-to-file transmission
60		SMF_060	VSAM volume data set updated
61		SMF_061	Integrated Catalog Facility Define Activity
62		SMF_062	VSAM component or cluster opened
		SMF_062_ONLINE	Online volume information
63		SMF_063	VSAM entry defined
64		SMF_064	VSAM component or cluster status
		SMF_064_EXTENT	Extent information section
65		SMF_065	Integrated Catalog Facility Delete Activity
66		SMF_066	Integrated Catalog Facility Alter Activity
67		SMF_067	VSAM entry delete
68		SMF_068	VSAM entry renamed

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
69		SMF_069	VSAM data space: defined, extended, or deleted
70	1	SMF_070	RMF CPU activity
		SMF_070_CPU	CPU data section
		SMF_070_BCT	PR/SM partition data section
		SMF_070_BPD	PR/SM logical processor data section
		SMF_070_INS	CPU identification section
		SMF_070_LCD	Logical core data section
		SMF_070_TRG	Tenant Resource Group data section
	2	SMF_070_2	RMF Cryptographic Hardware Activity
		SMF_070_PCICA	Cryptographic Accelerator Data Section.
		SMF_070_PCICC	Cryptographic CCA coprocessor data section
		SMF_070_PKCS11	Cryptographic PKCS11 coprocessor data section
71	1	SMF_071	RMF paging activity
		SMF_071_SWAP	Swap placement section
72	1	SMF_072_1	RMF workload activity
		SMF_072_PGP	Performance Group Period data section
	2	SMF_072_2	RMF storage data
		SMF_072_2_DATA	Performance Group data section
		SMF_072_2_SWAP_RSN	Swap reason data section
	3	SMF_072_3	RMF goalmode workload activity
		SMF_072_SSS	Service class served data section
		SMF_072_SCS	Service/Report Class period data section
		SMF_072_WRS	Work Manager/Resource Manager state section
		SMF_072_DNS	Resource delay type names section
	4	SMF_072_4	RMF Goalmode delay and storage frame data
		SMF_072_4_DATA	Service class period data section
		SMF_072_4_SWAP_RSN	Swap reason data section
	5	SMF_072_5	RMF system suspend locks and GRS data
		SMF_072_CMS_LOCK	CMS lock data section
		SMF_072_ENQ_LOCK	CMS Enqueue/Dequeue lock data section
		SMF_072_LATCH_LOCK	CMS latch lock data section
		SMF_072_SMF_LOCK	CMS SMF lock data section
		SMF_072_LOCK_TYPE	Local lock data section
		SMF_072_LOCK_OWNER	CML lock owner data section
		SMF_072_LOCK_RQSTR	CML lock requestor data section
		SMF_072_LATCH_CRTR	Latch creator data section
		SMF_072_OFFSET_LR	Latch requestor data section
		SMF_072_GRS_ENQ	GRS Enqueue step data section
		SMF_072_ENQ_SYS	GRS Enqueue system data section
		SMF_072_ENQ_SYSS	GRS Enqueue systems data section
		SMF_072_GRS_QSCAN	GRS QScan statistics data section
73	1	SMF_073	RMF channel path activity
		SMF_073_CHAN_PATH	Channel path data section
		SMF_073_EXT_CHAN	Extended channel path data section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
74	1	SMF_074_1	RMF device activity
		SMF_074_DEV_DATA	Device data section
	2	SMF_074_2	RMF XCF activity
		SMF_074_SYS_DATA	System data section
		SMF_074_PATH_DATA	Path data section
		SMF_074_MBR_DATA	Member data section
	3	SMF_074_3	RMF OPENMVS kernel activity
		SMF_074_OMVS_DATA	Control data section
	4	SMF_074_4	RMF XES/CF activity
		SMF_074_CONN_DATA	Connectivity data section
		SMF_074_STRUC_DATA	Structure data section
		SMF_074_RQST_DATA	Request data section
		SMF_074_PROC_DATA	Processor utilization data section
		SMF_074_CACHE_DATA	Cache data section
		SMF_074_REMOTE_FAC	Remote facility data section
		SMF_074_CHAN_PATH	Channel path data section
		SMF_074_SC_MEMDATA	Storage class memory data section
		SMF_074_ACFDS	Asynchronous CF Duplexing Data Section
	5	SMF_074_5	RMF Cache activity
		SMF_074_CACHE_DEV	Cache device data section
		SMF_074_XDEV	Cache device data section extension
		SMF_074_CCU_STATUS	Cache control unit status section
		SMF_074_RAID_RANK	RAID Rank/Extent Pool data section
	6	SMF_074_6	RMF Hierarchical file system activity
		SMF_074_HFS_GLOBAL	HFS global data section
		SMF_074_HFS_BUFFER	HFS global buffer section
		SMF_074_HFS	HFS file system section
	7	SMF_074_7	RMF FICON® Director Statistics
		SMF_074_FCD_GLOBAL	FCD global data section
		SMF_074_FCD_PORT	FCD port data section
		SMF_074_FCD_CONN	FCD connector data section
	8	SMF_074_8	RMF Enterprise Storage Server® (ESS) Link Statistics
		SMF_074_ESS_LINK	Link statistics section
		SMF_074_EXT_POOL	Extent pool statistics section
		SMF_074_RANK_STATS	Rank statistics section
		SMF_074_RANK_ARRAY	Rank array data section
		SMF_074_SILS_ARRAY	Synchronous I/O Link Statistics Section
	9	SMF_074_9	RMF Monitor III PCIE Statistics
		SMF_074_PCIE_FUNC	PCIE function data section
		SMF_074_DMA_00	PCIE function type data section for format x'00'
		SMF_074_DMA_01	PCIE function type data section for format x'01'
		SMF_074_DMA_02	PCIE function type data section for format x'02'
		SMF_074_DMA_03	PCIE function type data section for format x'03'
		SMF_074_DMA_04	PCIE function type data section for format x'04'
		SMF_074_HWAC	Hardware accelerator data section
		SMF_074_HWAC_COMP	Hardware accelerator compression data section
		SMF_074_SIOL	Synchronous I/O Link data section
		SMF_074_SIOR	Synchronous I/O response time distribution data section
		SMF_074_SCM	Storage Class Memory (SCM) Statistics
	10	SMF_074_EADM_DEV	SCM device (subchannel) information section
		SMF_074_SCMC	SCM configuration measurement section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
75	1	SMF_075	RMF page/swap data set activity
		SMF_075_PAGE_SWAP	Page Data Set data section
76		SMF_076	RMF trace activity
		SMF_076_PRODUCT	RMF product section
		SMF_076_TRCCTRL	Trace control section
		SMF_076_TRCDATA	Trace data section
		SMF_076_VARDATA	Variable trace data section
77	1	SMF_077	RMF enqueue activity
		SMF_077_ENQ	Enqueue data section
78	1	SMF_078_1	RMF I/O queuing activity for the 308x, 908x, and 4381 processors
		SMF_078_IOQDATA	I/O Queuing data section for 308x
	2	SMF_078_2	RMF virtual storage activity
		SMF_078_VSPA	Virtual Storage Private Area data section
		SMF_078_VSPASS	Virtual Storage Private Area subpool section
	3	SMF_078_3	RMF I/O queuing activity for the 3090, 9021, 9121, and 9221 processors
		SMF_078_HYPERPAV	HyperPAV data section
		SMF_078_IOQDATA3	I/O Queuing data section
79		SMF_079	RMF Monitor II activity
		SMF_079_ASDDATA	ASD and ASDJ data section
		SMF_079_ARDDATA	ARD and ARDJ data section
		SMF_079_SRCSDATA	SRCS data section
		SMF_079_SPAGDATA	SPAG data section
		SMF_079_ASRMDATA	ASRM and ASRMJ data section
		SMF_079_SENQRDATA	SENQR data section
		SMF_079_SENQDATA	SENQ data section
		SMF_079_TRXDATA	TRX data section
		SMF_079_DEVICEDATA	Device data section
		SMF_079_DDMNDATA	DDMN data section
		SMF_079_PGSPDATA	PGSP control section
		SMF_079_PGSP_DATA	PGSP data set section
		SMF_079_CHANNELCTL	Channel path control section
		SMF_079_IOCONFIG	I/O Queuing configuration control section for 308x
		SMF_079_IOQ_DATA	I/O Queuing configuration data section for 308x
		SMF_079_IOQUEDTA	I/O Queuing data section for 308x
		SMF_079_IOCONFIGQ	I/O Queuing global section
		SMF_079_IOQ_DATAS	I/O Queuing data section
		SMF_079_LONG_LOCK	IMS long lock data section
80 (for RACF)		SMF_080	RACF processing
		SMF_080_RELOCATE	RACF relocate section
		SMF_080_XRELOCATE	RACF extended relocate section
80 (for CA Top Secret)		SMF_080_CA_16	CA Top Secret security-related activity
		SMF_080_CA_REL	CA Top Secret security-related audit and logging information
81		SMF_081	RACF initialization
		SMF_081_RELOCATE	RACF relocate section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
82	1	SMF_082_1	PCF (Programmed Cryptographic Facility) record
		SMF_082_1_GENERAL	PCF repeated section
	2	SMF_082_2	CUSP (Cryptographic Unit Support Program) record
		SMF_082_2_GENERAL	CUSP repeated section
83		SMF_083	RACF audit record for data sets
84	1	SMF_084_1	JES3 monitoring facility (JMF) FCT (Function Control Table) analysis
	2	SMF_084_2	JES3 monitoring facility (JMF) FCT summary and highlight
		SMF_084_JES3_WAIT	JES3 wait analysis section
	3	SMF_084_3	JES3 monitoring facility (JMF) Spool data management
	4	SMF_084_4	JES3 monitoring facility (JMF) Resqueue cellpool, JCT, and control block utilization
	5	SMF_084_5	JES3 monitoring facility (JMF) Job analysis
	6	SMF_084_6	JES3 monitoring facility (JMF) JES3 hot spot analysis
	7	SMF_084_7	JES3 monitoring facility (JMF) JES3 internal reader DSP analysis
	8	SMF_084_8	JES3 monitoring facility (JMF) JES3 SSI response time analysis
	9	SMF_084_9	JES3 monitoring facility (JMF) JES3 SSI destination queue analysis
	10	SMF_084_10	JES3 monitoring facility (JMF) JES3 Workload Manager Analysis
85		SMF_085	OAM record
		SMF_085_ARRAY	Volume array section
88		SMF_088	System logger
89		SMF_089	Product Usage Data
		SMF_089_USAGE_DATA	Usage data section
		SMF_089_PROD_ISECT	Product intersection data section
		SMF_089_STATE_DATA	State data section
90		SMF_090	System status
		SMF_090_SMFDATASET	SMF data set section
		SMF_090_SUBSYSTEM	Subsystem record section
		SMF_090_SUBPARM	Subsystem parameter section
92		SMF_092	OpenMvs File System Activity
94	1	SMF_094	34xx tape library data server statistics
	2	SMF_094_2	Volume Pool Statistics

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
99		SMF_099	System resource manager decisions
		SMF_099_REASM_INFO	Reassembly area information
		SMF_099_AAT	Trace table entry section
		SMF_099_SS	System state information section
		SMF_099_PP	System paging plot information section
		SMF_099_PT	Priority table entry section
		SMF_099_RG	Resource group entry section
		SMF99_S1_GENRES	Generic resource entry section
		SMF99_S1_SL	Software licensing information
		SMF99_S1_SLT	Software licensing table information
		SMF99_S1_ZE	ZE information section
		SMF99_S1_BP	Buffer pool section
		SMF99_S2_CLS	Class data section
		SMF99_S2_XMEM	Cross memory delay entry section
		SMF99_S2_SERVER	Server data entry section
		SMF99_S2_SDATA	Server sample data entry section
		SMF99_S2_QDATA	Queue server data entry section
		SMF99_S2_ASESP	Address space expanded storage access policy section
		SMF99_S3_CLS	Class data section
		SMF99_S3_PPRP	Period paging rate plot section
		SMF99_S3_RUA	Ready user average plot section
		SMF99_S3_SWP	Swap delay plot section
		SMF99_S3_PAS	Proportional aggregate speed plot section
		SMF99_S3_QMPLP	Queue delay plot section
		SMF99_S3_QRUAP	Queue ready user average plot section
		SMF99_S3_AINS	Active server instances plot section
		SMF99_S3_ASTR	VS plot for active server instances section
		SMF99_S3_TSTR	VS plot for total server instances section
		SMF99_S3_QSTP	Queue service time plot section
		SMF99_S4_IOPT	Device cluster priority table section
		SMF99_S4_IOPLOT	I/O plot information section
		SMF99_S5_MON	Monitored address space information
		SMF99_S6_PDS	Period data section
		SMF99_S7_PAV	PAV device section
		SMF99_S8_LPAR	LPAR data entry section
		SMF99_S8_PT	Priority table entry section
		SMF99_S8_IOSUB	I/O subsystems samples data section
		SMF99_S8_ICPU	LPAR CPU data for a partition in an LPAR cluster section
		SMF99_S8_SYSH	SYSH CPU plot section
		SMF99_S9_SUBS	Channel path data entry section
		SMF99_S9_PLOT	I/O subsystem plot section
		SMF99_S9_CHAN	Channel path data entry section
		SMF99_SA_CPUD	CPU data section
		SMF99_SA_PCHGO	Processor speed change (old)
		SMF99_SA_PCHGN	Processor speed change (new)
		SMF99_SB_DATA	Capacity group data section
		SMF99_SB_CECS	CEC service data section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
100	0	SMF_100_0	Db2 statistics, system services
		SMF_100_ADDR_SPACE	Address space data section
		SMF_100_DEST	Instrumentation destination data section
		SMF_100_INST	Instrumentation data section
		SMF_100_LATCH_MGR	Latch manager data section
		SMF_100_STRGE_MGR9	Storage manager data section (Db2 V9 and below)
		SMF_100_STRGE_MGR	Storage manager data section (Db2 V10 and above)
		SMF_100_DDF9	Distributed data facility section (Db2 V9 and below)
		SMF_100_DDF	Distributed data facility section (Db2 V10 and above)
	1	SMF_100_1	Db2 statistics - database services
		SMF_100_BIND	Bind data section (DSNDQTST)
		SMF_100_BUFF_MGR9	Buffer manager data section (DSNDQBST - Db2 V9 and below)
		SMF_100_BUFF_MGR	Buffer manager data section (DSNDQBST - Db2 V10 and above)
		SMF_100_DATA_MGR9	Data manager data section (DSNDQIST - Db2 V9 and below)
		SMF_100_DATA_MGR	Data manager data section (DSNDQIST - Db2 V10 and above)
		SMF_100_BUFF_POOL9	Buffer manager group buffer pool (Db2 V9 and below)
		SMF_100_BUFF_POOL	Buffer manager group buffer pool (Db2 V10 and above)
		SMF_100_SERV_CNTL	Service controller locking statistics
		SMF_100_IDAA_DATA	IDAA data section
		SMF_100_SIMUL_BP	Simulated Buffer Pool section
	2	SMF_100_2	Db2 statistics - Dynamic ZPARMS
	3	SMF_100_3	Db2 statistics - Buffer Manager Group Buffer Pool
		SMF_100_3BUFF_POOL	Buffer manager group buffer pool
	4	SMF_100_4	Db2 System Storage® Usage
		SMF_100_DB2_STRGE9	Db2 system storage usage (Db2 V9 and below)
		SMF_100_DB2_STRGE	Db2 system storage usage (Db2 V10 and above)
		SMF_100_STOR_THRD	Thread information (QW02252)
		SMF_100_STOR_CMN	Shared and common storage summary (QW02253)
		SMF_100_STOR_STMT	Statement cache and shareable statement detail (QW02254)
		SMF_100_STOR_POOL	Pool details (QW02255)
		SMF_100_STOR_IRLM	IRLM storage information (QW02256)
	5	SMF_100_5	DB2 aggregated accounting statistics
		SMF_100_QW0369_2	Connection types(QW0369_2)
		SMF_100_INSTRMET	Instrumentation data
		SMF_100_CLASSOVF	Class 3 overflow data

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
101	0	SMF_101	Db2 accounting
		SMF_101_BUFFER_31	Buffer manager accounting block (DSNDQBAC)
		SMF_101_DIST9	Distributed data facility statistics (DSNDQLAC - Db2 V9 and below)
		SMF_101_DIST	Distributed data facility statistics (DSNDQLAC - Db2 V10 and above)
		SMF_101_QMDA	Distributed QMD accounting data (DSNDQMDA)
		SMF_101_IFI	Distributed IFI accounting data (DSNDQIFA)
		SMF_101_PACKAGE	Package accounting data (DSNDQPAC)
		SMF_101_QWAR	Rollup accounting correlation block (DSNDQWAR)
		SMF_101_BUFFER_MGR	Buffer manager group buffer pool accounting information (DSNDQBGA)
		SMF_101_GLBL_LOCK	Service controller global locking accounting block (DSNDQTGA)
		SMF_101_DATA_SHARE	Data sharing accounting data (DSNDQWDA)
		SMF_101_IDAA_ACCT	Accelerator services accounting block (DSNDQ8AC)
	1	SMF_101_1	Db2 accounting IFCID 239
		SMF_101_1_PACKAGE	Package accounting data (DSNDQPAC)
		SMF_101_SQL_ACC	SQL accounting data (DSNDQXPK)
		SMF_101_BUFMGR_ACC	Buffer manager accounting block (DSNDQBAC)
		SMF_101_LOCK_ACC	Lock manager accounting block (DSNDQTXA)
102		SMF_102	Db2 system initialization parameters
		SMF_102_SYS_PARM	System initialization parameters (DSNDQWPZ)
		SMF_102_INI_PARM	Log initialization parameters (DSNDQWPZ)
		SMF_102_ARCH_PARM	Archive initialization parameters (DSNDQWPZ)
		SMF_102_SYS_PARM8	System parameters (DSNDQWPZ - Db2 V8)
		SMF_102_SYS_PARM9	System parameters (DSNDQWPZ - Db2 V9)
		SMF_102_SYS_PARMA	System parameters (DSNDQWPZ - Db2 V10)
		SMF_102_SYS_PARMB	System parameters (DSNDQWPZ - Db2 V11)
		SMF_102_SYS_PARMC	System parameters (DSNDQWPZ - Db2 V12)
		SMF_102_DDF_START	DDF start control information (DSNDQWPZ)
		SMF_102_DATA_SHARE	Group initialization parameters for data sharing (DSNDQWPZ)
		SMF_102_DSNHDECP	DSNHDECP parameters (DSNDQWPZ)
103	1	INT_103_01	Internet Connection Secure Server configuration Record
	2	INT_103_02	Internet Connection Secure Server performance record
104		SMF_104	RMF Distributed Platform Performance Data
		SMF_104_METRICS	Metric section
108	1	SMF_108_01	Domino® Statistics Server Load
		SMF_108_TRAN	Transaction section
		SMF_108_PORT_ACT	Port activity section
	2	SMF_108_02	Domino Statistics User Activity
		SMF_108_USER_ACT	User activity server load section
	3	SMF_108_03	Domino Statistics Monitoring and Tuning
	6	SMF_108_06	Domino Statistics Data Base Activity
		SMF_108_DB_ACT	Database activity data section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
110	0	SMF_110_0	CICS/ESA monitoring record
	1	SMF_110_1	CICS Transaction Server for z/OS monitoring data
		SMF_110_1_FIELD	Field connectors
		SMF_110_1_DICT	Dictionary data
		SMF_110_1_KPI	Information about key performance indicators (KPIs) for CICS Transaction Server for z/OS monitoring
		SMF_110_1_5	CICS/MVS transaction data
		SMF_110_1_6	CTS Monitoring identity data
		SMF_110_E	Monitoring exception data
	2	SMF_110_2	CICS statistics
	3	SMF_110_3	CICS statistics
	4	SMF_110_4	CICS/TS Coupling Facility statistics
	5	SMF_110_5	CICS/TS Named server sequence statistics
111		SMF_111	CICS TS for z/OS Statistics
112	203	SMF_112_203	OMEGAMON® CICS
113	1	SMF_113_1	Hardware capacity delta statistics
		SMF_113_1_SCDS	Short counters section
		SMF_113_1_LCDS	Long counters section
	2	SMF_113_2	Hardware capacity reporting and statistics
		SMF_113_2_CSS	Counter set section
		SMF_113_2_CDS	Counter data section
114	1	SMF_114_1	System Automation Tracking
115	1	MQS_115_1	MQSeries® log manager statistics
		MQS_115_QSST	Storage manager statistics section
		MQS_115_QJST	Log manager statistics section
	2	MQS_115_2	MQSeries information statistics
		MQS_115_QMST	Message manager statistics section
		MQS_115_QPST	Buffer manager statistics section
		MQS_115_QLST	Lock manager statistics section
		MQS_115_Q5ST	Db2 manager statistics section
		MQS_115_QTST	Data manager statistics section
		MQS_115_QESD	Shared message data sets section
	215	MQS_115_215	Buffer manager
		MQS_115_QPST215	Buffer Manager Buffer Pool Statistics section
	231	MQS_115_231	Channel initiator statistics data
		MQS_115_QCCT	CHINIT Control info section
		MQS_115_QCT_DSP	Dispatcher tasks
		MQS_115_QCT_ADP	Adapter tasks
		MQS_115_QCT_SSL	SSL tasks
		MQS_115_QCT_DNS	DNS task

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
116	0	MQS_116	MQSeries accounting statistics
		MQS_116_QWHS	Message manager section
		MQS_116_QMAC	Message manager accounting section
	1	MQS_116_1	MQSeries thread and queue level accounting statistics
		MQS_116_WTAS	Task-related statistics section
		MQS_116_WQST1	Queue-level accounting statistics section
	2	MQS_116_2	MQSeries queue level accounting statistics
		MQS_116_QWHS2	Common MQSeries SMF Header
		MQS_116_WQST2	Queue-level accounting statistics section
	10	MQS_116_10	Channel statistics
		MQS_116_QCST	Channel Statistics section
117		SMF_117	WebSphere Message Broker
		SMF_117_T1_THREAD	Thread data
		SMF_117_T2_NODE	Node data
		SMF_117_T2_TERM	Terminal data
118	1 - 2	SMF_118_1	TCP/IP API calls
	3	SMF_118_3	TCP/IP FTP Client Calls
	4	SMF_118_4	TCP/IP TELNET Client Calls record
	5	SMF_118_5	TCP/IP General Stats Record
		SMF_118_5_2	TCP/IP General Stats Record
	20 - 21	SMF_118_20	TCP/IP TELNET Server Record
	70 - 75	SMF_118_70	TCP/IP FTP Server

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)

Type	Subtype	Data stream name	Description of data stream content
119	1	SMF_119_1	TCP/IP Connection Initiation Record
	2	SMF_119_2	TCP/IP Connection Termination Record
	3	SMF_119_3	TCP/IP Client Transfer Completion
	4	SMF_119_4	TCP/IP Profile Information Record
	5	SMF_119_5	TCP/IP Statistics
	6	SMF_119_6	TCP/IP Interface Statistics
		SMF_119_INTERFACE	Interface statistics
		SMF_119_HOME_IP	Home IP address section
	7	SMF_119_7	TCP/IP Server Port Statistics
		SMF_119_TCP_PORT	TCP server port statistics section
		SMF_119_UDP_PORT	UDP server port statistics section
	8	SMF_119_8	TCP/IP Stack Start/Stop
	10	SMF_119_10	UDP Socket Close Record
	20	SMF_119_20	TN3270 Server SNA Session Initiation
	21	SMF_119_21	TN3270 Server SNA Session Termination
	22	SMF_119_22	TSO Telnet Client Connection Initiation
	23	SMF_119_23	TSO Telnet Client Connection Termination
	32	SMF_119_32	DVIPA Status Change Record
	33	SMF_119_33	DVIPA Removed Record
	34	SMF_119_34	DVIPA Target Added Record
	35	SMF_119_35	DVIPA Target Removed Record
	36	SMF_119_36	DVIPA Target Server Started Record
	37	SMF_119_37	DVIPA Target Server Ended Record
	48	SMF_119_48	CSSMTP Configuration record
	49	SMF_119_49	CSSMTP Connection Record
	50	SMF_119_50	CSSMTP Mail Record
	51	SMF_119_51	CSSMTP Spool File Record
	52	SMF_119_52	CSSMTP Statistical Record
	70	SMF_119_70	FTP Server Transfer Completion
	72	SMF_119_72	FTP Server Logon Failure
	73	SMF_119_73	IPSec IKE Tunnel Activation/Refresh
	74	SMF_119_74	IPSec IKE Tunnel Deactivation/Expire
	75 - 80	SMF_119_75_80	IPSec Dynamic/Manual Tunnel Activation/Refresh/Deactivate Add/Remove
120	9	SMF_120_9	WAS Request Activity Record
		SMF_120_9_CLA	Classification data section
		SMF_120_9_SEC	Security data section
		SMF_120_9_CPU	CPU usage breakdown section
		SMF_120_9_USR	User data section
	10	SMF_120_10	Outbound request record
	11	SMF_120_11	HTTP requests in liberty
		SMF_120_11_USD	User data section
		SMF_120_11_CLS	Classification section
	12	SMF_120_12	Batch job in liberty
		SMF_120_12_SC5	Accounting section
		SMF_120_12_SC7	Reference names section
		SMF_120_12_SC8	User data section

Table 13. Data stream names that IBM Common Data Provider for z Systems uses to collect SMF data (continued)			
Type	Subtype	Data stream name	Description of data stream content
123	1	SMF_123_01	The z/OS Connect EE audit interceptor records request activity to the SMF data store on z/OS operating systems.
127	1000	SMF_IMS_07	IMS program termination
		SMF_IMS_08	IMS program start
		SMF_IMS_0A07	IMS CPI-CI program termination
		SMF_IMS_0A08	IMS CPI-CI program start
		SMF_IMS_10	IMS security violation
		SMF_IMS_56FA	IMS transaction level statistics
		SMF_IMS_CA01	IMS Transaction Index
		SMF_IMS_CA20	IMS Connect Transaction Index
194		SMF_194	Definition of TS7700 Virtualization Engine Statistics Record
230		SMF_230_CA_16	CA ACF2 security-related activity
		SMF_230_CA_16_T1	For CA ACF2 record version 0 or 1, part of command trace information
		SMF_230_CA_16_T2	For CA ACF2 record version 2, part of command trace information
		SMF_230_CA_16_SNEN	Part of CA ACF2 distributed database sense information
231		SMF_231_CA_16	CA Top Secret security events for UNIX System Services
		SMF_231_CA_EXT	CA Top Secret audit records for UNIX System Services
245		SMF_245_3_CACHE	Cache RMF Reporter (3990 model 03)
		SMF_245_3_DEV	Caching subsystem device status entries
		SMF_CACHE_06	Cache RMF Reporter (3990 model 06)
		SMF_245_6_DEV	Caching subsystem device status entries
		SMF_CACHE_13	Cache RMF Reporter (3880 model 13)
		SMF_CACHE_23	Cache RMF Reporter (3880 model 23)

SMF_110_1_KPI data stream content

SMF_110_1_KPI records in the **SMF_110_1_KPI** data stream contain information about key performance indicators (KPIs) for CICS Transaction Server for z/OS monitoring.

Data stream definition in Configuration Tool

To select the **SMF_110_1_KPI** data stream in the IBM Common Data Provider for z Systems Configuration Tool, complete the following steps:

1. In the "Select data stream" window, expand **KPI Data**.
2. Expand **Base Streams**.
3. Expand **CICS**.
4. Select the **SMF_110_1_KPI** check box.

Fields in the SMF_110_1_KPI data stream

In the following table, the column that is titled "Corresponding SMF field" indicates the name of the SMF field that corresponds to the field name in the data stream.

Table 14. Fields in the **SMF_110_1_KPI** data stream

Field name	Description	Corresponding SMF field
Time	The time that the record was written to SMF	SMFMNTME
Date	The date that the record was written to SMF	SMFMNDTE
MVS_SYSTEM_ID	The system ID, which is also known as the SMF ID	SMFMNSID
START_TIMESTAMP	The start time of the transaction. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	START
STOP_TIMESTAMP	The stop time of the transaction. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	STOP
ELAPSED_TIME	The elapsed time of the transaction, which is derived by the System Data Engine (the stop time minus the start time)	Not applicable
CICS_SPEC_APPLID	CICS Transaction Server for z/OS specific application ID	SMFMNSPN
CICS_GEN_APPLID	CICS Transaction Server for z/OS generic application ID	SMFMNPRN
JOB_NAME	CICS Transaction Server for z/OS job name	SMFMNJBN
PGM_NAME	CICS Transaction Server for z/OS program name. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	PGMNAME
TRANSACTION_ID	CICS Transaction Server for z/OS transaction ID. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	TRAN
TRANSACTION_NUM	CICS Transaction Server for z/OS transaction number. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	TRANNUM
ORIG_ABEND_CODR	Original CICS Transaction Server for z/OS abend code. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	ABCODEO
CURR_ABEND_CODE	Current CICS Transaction Server for z/OS abend code. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	ABCODEC
CICS_USER	Current CICS Transaction Server for z/OS user ID. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	USERID
SYNCPPOINTS	The total number of syncpoint requests that are issued by the user task. The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.	SPSYNCCT

Table 14. Fields in the **SMF_110_1_KPI** data stream (continued)

Field name	Description	Corresponding SMF field
TERM_WAIT	<p>After the user task issued a RECEIVE request, the elapsed time during which the user task waited for input from the terminal operator.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	TCIOWTT
DISPATCH_TIME	<p>The total elapsed time during which the user task was dispatched on each CICS task control block (TCB) under which the task ran. The TCB modes that are managed by the CICS dispatcher are: QR, RO, CO, FO, SZ, RP, SL, SP, SO, EP, J8, J9, L8, L9, S8, TP, T8, X8, X9, JM, and D2.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	USRDISPT
CPU_TIME	<p>The total processor time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task ran.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	USRCPUT
RLS_CPU_TIME	<p>The amount of CPU time in which the transaction was processing record-level sharing (RLS) file requests.</p> <p>Tip: For a measurement of the total CPU time that is used by a transaction, add this RLS_CPU_TIME value to the CPU_TIME value (SMF field USRCPUT).</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	RLSCPUT
SUSP_TIME	<p>The time in which the user task was suspended by the CICS dispatcher.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	SUSPTIME
SYNCTIME	<p>The time in which the user task was dispatched and was processing syncpoint requests.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	SYNCTIME
DISP_CICS_USER	<p>The dispatch time that CICS Transaction Server for z/OS gives to a user task, which is the total elapsed time during which the user task is dispatched by the CICS dispatcher domain on a CICS Key 8 mode TCB.</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	KY8DISPT




Table 14. Fields in the **SMF_110_1_KPI** data stream (continued)

Field name	Description	Corresponding SMF field
JAVA_CPU_TIME	<p>This field is a composite field that indicates one of the following elements:</p> <ul style="list-style-type: none"> • The amount of CPU time that this task used when it was dispatched on the J8 TCB Mode • The number of times that this task was dispatched on the J8 TCB Mode • An indication that the mode is used by Java applications <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	J8CPUT
L8_TCB_DISP_TIME	<p>This field is a composite field that indicates one of the following elements:</p> <ul style="list-style-type: none"> • The amount of CPU time that this task used when it was dispatched on the L8 TCB Mode • The number of times that this task was dispatched on the L8 TCB Mode • An indication that the mode is used by programs that are defined with CONCURRENCY=THREADSAFE when they issue Db2 requests <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	L8CPUT
S8_TCB_DISP_TIME	<p>This field is a composite field that indicates one of the following elements:</p> <ul style="list-style-type: none"> • The amount of CPU time that this task used when it was dispatched on the S8 TCB Mode • The number of times that this task was dispatched on the S8 TCB Mode • An indication that the mode is used for making secure socket calls <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	S8CPUT
RMI_TIME	<p>The time in which the task was external to CICS Transaction Server for z/OS (for example, in Db2 or MQ).</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	RMITIME
RMI_SUSP_TIME	<p>The time in which the user task was suspended by the CICS dispatcher while it was in the CICS Resource Manager Interface (RMI).</p> <p>The field name corresponds to the CICS Transaction Server for z/OS Dictionary nickname.</p>	RMISUSP




Icons on each node in a policy

This reference describes the icons that are shown on each data stream, transform, and subscriber node that you define in a policy. It also indicates where you can find more information about configuring data streams, transforms, and subscribers.




Data stream node

Table 15. Icons on each data stream node in a policy		
Icon	Window that opens when you click icon	More information
Configure icon 	One of the following windows opens, depending on whether the data is gathered by the Log Forwarder or the System Data Engine: <ul style="list-style-type: none"> Configure Log Forwarder data stream Configure System Data Engine data stream 	<ul style="list-style-type: none"> “Data stream configuration for data gathered by Log Forwarder” on page 116 “Data stream configuration for data gathered by System Data Engine” on page 143
Transform icon 	<ul style="list-style-type: none"> Transform data stream 	<ul style="list-style-type: none"> “Transform configuration” on page 143
Subscribe icon 	<ul style="list-style-type: none"> Subscribe to a data stream 	<ul style="list-style-type: none"> “Adding a subscriber for a data stream or transform” on page 31 “Subscriber configuration” on page 147

Transform node

Table 16. Icons on each transform node in a policy		
Icon	Window that opens when you click icon	More information
Configure icon 	One of the following windows opens: <ul style="list-style-type: none"> Configure Transcribe transform Configure Splitter transform Configure Filter transform 	<ul style="list-style-type: none"> “Transform configuration” on page 143
Transform icon 	<ul style="list-style-type: none"> Transform data stream 	<ul style="list-style-type: none"> “Transform configuration” on page 143
Subscribe icon 	<ul style="list-style-type: none"> Subscribe to a transform 	<ul style="list-style-type: none"> “Adding a subscriber for a data stream or transform” on page 31 “Subscriber configuration” on page 147

Subscriber node

Table 17. Icons on each subscriber node in a policy		
Icon	Window that opens when you click icon	More information
Configure icon 	<ul style="list-style-type: none">• Configure subscriber	<ul style="list-style-type: none">• “Adding a subscriber for a data stream or transform” on page 31• “Subscriber configuration” on page 147
Export icon 	<ul style="list-style-type: none">• Export	<ul style="list-style-type: none">• “Exporting and importing subscribers” on page 32
Subscribe icon 	<ul style="list-style-type: none">• Update the subscriptions of this subscriber	<ul style="list-style-type: none">• “Adding a subscriber for a data stream or transform” on page 31• “Subscriber configuration” on page 147

Data stream configuration for data gathered by Log Forwarder

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window. The fields that are shown in this window are based on the source from which the Log Forwarder collects data for the data stream.

The Log Forwarder gathers z/OS log data from the following sources:

- Job log, which is output that is written to a data definition (DD) by a running job
- z/OS UNIX log file, including the UNIX System Services system log (syslogd)
- Entry-sequenced Virtual Storage Access Method (VSAM) cluster
- z/OS system log (SYSLOG)
- IBM Tivoli NetView for z/OS messages
- IBM WebSphere Application Server for z/OS High Performance Extensible Logging (HPEL) log
- IBM Information Management System (IMS) log

[Table 18 on page 117](#) summarizes which data streams come from which sources.

Table 18. Correlation between the sources from which the Log Forwarder gathers data and the data streams that can be defined for those sources

Source	Data streams
Job log	<ul style="list-style-type: none"> • “Generic z/OS Job Output data stream” on page 119 • “CICS EYULOG data stream” on page 121 • “CICS EYULOG DMY data stream” on page 123 • “CICS EYULOG YMD data stream” on page 125 • “CICS User Messages data stream” on page 127 • “CICS User Messages DMY data stream” on page 129 • “CICS User Messages YMD data stream” on page 131 • “WebSphere SYSOUT data stream” on page 135 • “WebSphere SYSPRINT data stream” on page 137
z/OS UNIX log file	<ul style="list-style-type: none"> • “Generic ZFS File data stream” on page 118 • “USS Syslogd data stream” on page 133 • “WebSphere USS Sysprint data stream” on page 139
Entry-sequenced VSAM cluster	<ul style="list-style-type: none"> • “Generic VSAM Cluster data stream” on page 117
z/OS SYSLOG	<ul style="list-style-type: none"> • “z/OS SYSLOG data stream” on page 139
IBM Tivoli NetView for z/OS messages	<ul style="list-style-type: none"> • “NetView Netlog data stream” on page 133
IBM WebSphere Application Server for z/OS HPEL log	<ul style="list-style-type: none"> • “WebSphere HPEL data stream” on page 134

Generic VSAM Cluster data stream

This reference lists the configuration values that you can update in the **"Configure Log Forwarder data stream"** window for the **Generic VSAM Cluster** data stream. It also describes why you might want to define paired data sets for this data stream.

Data collection from paired data sets

For the **Generic VSAM Cluster** data stream, the Log Forwarder can gather log data from a logical pair of data sets, called *paired data sets*. The use of paired data sets prevents an individual data set from getting too large and makes the process of pruning old log data from the system much easier.

With paired data sets, data is logged to only one data set in the pair at a time. When that data set exceeds some threshold (for example, the data set surpasses a specified size, or a specified time interval passes), the data in the other data set is deleted, and logging switches to that other data set. This switching between each data set in the pair is repeated continuously as each threshold is exceeded.

When you define a **Generic VSAM Cluster** data stream, you can specify either a single data set (in the **Data Set Name** field) or two data sets (one in the **Data Set Name** field, and the other in the **Paired Data Set Name** field) that are logically paired. If you specify two data sets, the contents of both data sets are associated with the same data stream. Both data sets must be entry-sequenced Virtual Storage Access Method (VSAM) clusters. At least one of the data sets must be allocated before the Log Forwarder is started.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Data Set Name

The name of the entry-sequenced VSAM cluster that contains the data to be gathered. This name must be in the format *x.y.z*.

Paired Data Set Name

The name of the entry-sequenced VSAM cluster that, together with the cluster that is specified in the **Data Set Name** field, contains the data to be gathered. This name must be in the format *x.y.z*.

Tip: For more information about the use of paired data sets, see [“Data collection from paired data sets”](#) on page 117.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration”](#) on page 147.

Data Source Type

A value that the subscriber can use to uniquely identify the type and format of the streamed data.

File Path

A unique identifier that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Generic ZFS File data stream

This reference lists the configuration values that you can update in the **"Configure Log Forwarder data stream"** window for the **Generic ZFS File** data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

File Path

A unique identifier that represents the data origin. The identifier must be the absolute path, including the file name, of a log file that contains the relevant data.

Tip: If you are gathering log data from a rolling z/OS UNIX log, see [“Data collection from a rolling z/OS UNIX log” on page 140](#) for more information, including how to specify this file path value for a rolling log.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration” on page 147](#).

Data Source Type

A value that the subscriber can use to uniquely identify the type and format of the streamed data.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Generic z/OS Job Output data stream

This reference lists the configuration values that you can update in the **“Configure Log Forwarder data stream”** window for the **Generic z/OS Job Output** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see [“Use of wildcard characters in the Job Name field” on page 120](#).

DD Name

The data definition (DD) name for the job log.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration”](#) on page 147.

Data Source Type

A value that the subscriber can use to uniquely identify the type and format of the streamed data.

File Path

A unique identifier, such as *jobName/ddName*, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is ABCD????, and the JES spool contains the following jobs, two data streams are created, one for job name ABCD1234 and one for job name ABCDE567:

JOBNAME	JobID
ABCD1234	STC00735
DEFG1234	STC00746

JOBNAME	JobID
ABCDE567	STC00798
DEFG5678	STC00775
ABCD123	STC00772
DEFG456	STC00794
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_ddName</code> appended to that value. The <code>jobName</code> is the discovered job name, and the <code>ddName</code> is the DD name for the job log.
File Path	The value of the File Path field in the template, with <code>/jobName/ddName</code> appended to that value. The <code>jobName</code> is the discovered job name, and the <code>ddName</code> is the DD name for the job log.

CICS EYULOG data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS EYULOG** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS EYULOG** data stream uses the date format "month day year" (MDY) in the time stamp.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 122.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration”](#) on page 147.

File Path

A unique identifier, such as *jobName/ddName*, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587

JOBNAME	JobID
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_EYULOG</code> appended to that value. The <i>jobName</i> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/EYULOG</code> appended to that value. The <i>jobName</i> is the discovered job name.

CICS EYULOG DMY data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS EYULOG DMY** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS EYULOG DMY** data stream uses the date format "day month year" (DMY) in the time stamp.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 124.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier, such as *jobName/ddName*, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600

JOBNAME	JobID
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_EYULOG</code> appended to that value. The <code>jobName</code> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/EYULOG</code> appended to that value. The <code>jobName</code> is the discovered job name.

CICS EYULOG YMD data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS EYULOG YMD** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS EYULOG YMD** data stream uses the date format "year month day" (YMD) in the time stamp.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 126.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier, such as `jobName/ddName`, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600
HBODSPRO	STC00623

JOBNAME	JobID
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_EYULOG</code> appended to that value. The <code>jobName</code> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/EYULOG</code> appended to that value. The <code>jobName</code> is the discovered job name.

CICS User Messages data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS User Messages** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS User Messages** data stream uses the date format "month day year" (MDY) in the time stamp.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 128.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier, such as `jobName/ddName`, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which

is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.

CICS User Messages DMY data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS User Messages DMY** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS User Messages DMY** data stream uses the date format "day month year" (DMY) in the time stamp.

Configuration values that you can update**Name**

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 130.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier, such as `jobName/ddName`, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in ["Log Forwarder properties configuration"](#) on page 93.

The value must be in the format `plus_or_minusHHMM`, where `plus_or_minus` represents the + or - sign, `HH` represents two digits for the hour, and `MM` represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on

other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.

CICS User Messages YMD data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **CICS User Messages YMD** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream. The source for the **CICS User Messages YMD** data stream uses the date format "year month day" (YMD) in the time stamp.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see ["Use of wildcard characters in the Job Name field"](#) on page 132.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier, such as `jobName/ddName`, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in ["Log Forwarder properties configuration"](#) on page 93.

The value must be in the format `plus_or_minusHHMM`, where `plus_or_minus` represents the + or - sign, `HH` represents two digits for the hour, and `MM` represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500

If you want this time zone	Specify this value
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is CMAS5*, and the JES spool contains the following jobs, two data streams are created, one for job name CMAS53 and one for job name CMAS5862:

JOBNAME	JobID
CMAS43	STC00586
CMAS482	STC00588
CMAS53	STC00587
CMAS5862	STC00589
CMAS61	STC00590
CMAS62	STC00600
HBODSPRO	STC00623
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.
File Path	The value of the File Path field in the template, with <code>/jobName/MSGUSR</code> appended to that value. The <code>jobName</code> is the discovered job name.

NetView Netlog data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **NetView Netlog** data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Domain Name

The name of the NetView domain from which to gather data.

Important: If you define multiple **NetView Netlog** data streams, do not define the same NetView domain name for multiple streams. Each stream must reference a unique domain name.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier that represents the data origin.

USS Syslogd data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **USS Syslogd Admin**, **USS Syslogd Debug**, and **USS Syslogd Error** data streams.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier that represents the data origin. The identifier must be the absolute path, including the file name, of a log file that contains the relevant data.

Tip: If you are gathering log data from a rolling z/OS UNIX log, see [“Data collection from a rolling z/OS UNIX log” on page 140](#) for more information, including how to specify this file path value for a rolling log.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

WebSphere HPEL data stream

This reference lists the configuration values that you can update in the **"Configure Log Forwarder data stream"** window for the **WebSphere HPEL** data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Log Directory

The HPEL log directory for an application server that you are collecting data from. This HPEL log directory must have a `logdata` subdirectory, and the HPEL log files must be present in the `logdata` subdirectory.

If you are collecting only trace data, do not specify a value in the **Log Directory** field.

If no value is specified in the **Trace Directory** field, a **Log Directory** value is required. Otherwise, the **Log Directory** value is not required.

Trace Directory

The WebSphere Application Server for z/OS HPEL trace directory for an application server that you are collecting data from. This HPEL trace directory must have a `tracedata` subdirectory, and the HPEL trace files must be present in the `tracedata` subdirectory.

If you are collecting only log data, do not specify a value in the **Trace Directory** field.

If no value is specified in the **Log Directory** field, a **Trace Directory** value is required. Otherwise, the **Trace Directory** value is not required.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration” on page 147](#).

File Path

A unique identifier that represents the data origin. The identifier must be a virtual or physical path that represents the HPEL log data.

WebSphere SYSOUT data stream

This reference lists the configuration values that you can update in the **"Configure Log Forwarder data stream"** window for the **WebSphere SYSOUT** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see [“Use of wildcard characters in the Job Name field” on page 136](#).

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration” on page 147](#).

File Path

A unique identifier, such as *jobName/ddName*, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration” on page 93](#).

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is BBOS???S, and the JES spool contains the following jobs, two data streams are created, one for job name BBOSABCS and one for job name BBOSDEFS:

JOBNAME	JobID
BBODMGR	STC00586
BBODMGRS	STC00588
BBODMNC	STC00587
BBON001	STC00589
BBOSABC	STC00590
BBOSABC	STC00600
BBOSABCS	STC00592
BBOSABCS	STC00602
BBOSDEF	STC00594
BBOSDEFS	STC00596
BBOSDEFS	STC00598
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_SYSOUT</code> appended to

Template field	Value
	that value. The <i>jobName</i> is the discovered job name.
File Path	The value of the File Path field in the template, with <i>/jobName/SYSOUT</i> appended to that value. The <i>jobName</i> is the discovered job name.

WebSphere SYSPRINT data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **WebSphere SYSPRINT** data stream. It also describes how to use wildcard characters in the **Job Name** field for this data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Job Name

The name of the server job from which to gather data. This value can contain wildcard characters.

For information about the use of wildcard characters, see [“Use of wildcard characters in the Job Name field”](#) on page 138.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration”](#) on page 147.

File Path

A unique identifier, such as *jobName/ddName*, that represents the data origin.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

Discovery Interval

In the process of streaming data, the number of minutes that the Log Forwarder waits before it checks for a new log file in the data stream. This value applies to all data streams from the Log Forwarder, although it can be overridden on some individual streams, such as this one.

The value must be an integer in the range 1 - 5. The default value is the value that is defined in the Log Forwarder properties, as described in [“Log Forwarder properties configuration”](#) on page 93.

Use of wildcard characters in the Job Name field

In the **Job Name** field for this data stream, you can use the following wildcard characters:

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters, including an empty sequence

If you use wildcard characters in the job name, the job name value becomes a pattern, and the data stream definition becomes a template. When the Log Forwarder starts, it searches the Job Entry Subsystem (JES) spool for job names that match the pattern, and it creates a separate data stream for each unique job name that it discovers. After the Log Forwarder initialization is complete, the Log Forwarder continues to monitor the job names on the JES spool. As it discovers new job names that match the pattern, it uses the same template to create more data streams.

For example, if the job name value is BBOS???S, and the JES spool contains the following jobs, two data streams are created, one for job name BBOSABCS and one for job name BBOSDEFS:

JOBNAME	JobID
BBODMGR	STC00586
BBODMGRS	STC00588
BBODMNC	STC00587
BBON001	STC00589
BBOSABC	STC00590
BBOSABC	STC00600
BBOSABCS	STC00592
BBOSABCS	STC00602
BBOSDEF	STC00594
BBOSDEFS	STC00596
BBOSDEFS	STC00598
GLAPROC	STC00661
SYSLOG	STC00552

Tips:

- To avoid gathering data from job logs that you do not intend to gather from, use a job name pattern that is not too broad.
- The Log Forwarder might discover jobs from other systems if spool is shared between systems or if JES multi-access spool is enabled. Although the data stream does not include data for the jobs that run on other systems, the Log Forwarder creates a data stream for that data. Therefore, ensure that the wildcard pattern does not match jobs that run on other systems.

Each resulting data stream is based on the template and has the same configuration values as the template, with the exception of the following values:

Template field	Value
Job Name	The discovered job name
Data Source Name	The value of the Data Source Name field in the template, with <code>_jobName_SYSPRINT</code> appended to

Template field	Value
	that value. The <i>jobName</i> is the discovered job name.
File Path	The value of the File Path field in the template, with <i>/jobName/SYSPRINT</i> appended to that value. The <i>jobName</i> is the discovered job name.

WebSphere USS Sysprint data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **WebSphere USS Sysprint** data stream.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see ["Subscriber configuration"](#) on page 147.

File Path

A unique identifier that represents the data origin. The identifier must be the absolute path, including the file name, of a log file that contains the relevant data.

Tip: If you are gathering log data from a rolling z/OS UNIX log, see ["Data collection from a rolling z/OS UNIX log"](#) on page 140 for more information, including how to specify this file path value for a rolling log.

Time Zone

If the time stamps in the collected data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone, which is defined in the Log Forwarder properties, as described in ["Log Forwarder properties configuration"](#) on page 93.

The value must be in the format *plus_or_minusHHMM*, where *plus_or_minus* represents the + or - sign, *HH* represents two digits for the hour, and *MM* represents two digits for the minute.

Examples:

If you want this time zone	Specify this value
Coordinated Universal Time (UTC)	+0000
5 hours west of UTC	-0500
8 hours east of UTC	+0800

z/OS SYSLOG data stream

This reference lists the configuration values that you can update in the "**Configure Log Forwarder data stream**" window for the **z/OS SYSLOG** (from user exit) and **z/OS SYSLOG from OPERLOG** data streams.

Configuration values that you can update

Name

The name that uniquely identifies the data stream to the Configuration Tool. If you want to add more data streams of the same type, you must first rename the last stream that you added.

Data Source Name

The name that uniquely identifies the data source to subscribers.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration” on page 147](#).

File Path

A unique identifier that represents the data origin.

Data collection from a rolling z/OS UNIX log

For data streams that come from z/OS UNIX log file sources, IBM Common Data Provider for z Systems can gather log data from rolling z/OS UNIX logs. The use of a rolling log prevents any one log file from getting too large and simplifies the process of pruning older log data from the system.

Tip: The following data streams come from z/OS UNIX log file sources:

- [“Generic ZFS File data stream” on page 118](#)
- [“USS Syslogd data stream” on page 133](#)
- [“WebSphere USS Sysprint data stream” on page 139](#)

Use of a rolling log

A *rolling log* is a dynamic, sequential set of files that contains a continuous stream of log data. A new file is added whenever a previous file exceeds some threshold (for example, the file surpasses a specified size, or a specified time interval passes). Sometimes, older files are pruned (automatically or manually) so that only a defined number of files is retained.

For example, with a rolling log, a new file might be created once a day, or at specified times. The log is a set of logically grouped log files, rather than only one log file. Individual files are differentiated by an index or a time stamp in the file name.

Important: IBM Common Data Provider for z Systems does not gather log data from a rolling log if the following events occurred when the log was rolled:

- The name of a log file was changed to a name that does not match the configured file pattern.
- The contents of a log file were removed.

File path pattern for a rolling log

IBM Common Data Provider for z Systems uses a file path pattern with one or more wildcard characters to identify the log files that must be logically grouped into one logical log (a rolling log) and mapped to the same data source name.

You must determine the appropriate file path pattern for each set of log files that are gathered, and specify this pattern in the **File Path** field when you configure a data stream that comes from a z/OS UNIX log file source. The file path pattern must be as specific as possible so that only the appropriate log files are included.

The following wildcard characters are valid in a file path pattern (in the **File Path** field for a data stream that comes from a z/OS UNIX log file source):

Wildcard character	What the character represents
?	Any single character
*	Any sequence of characters

Example of how to specify the file path pattern: Assume that a rolling log uses the following file naming scheme, where the integer *n* is incremented for each new log file:

- /u/myLogDir/myLogFile.*n*.log

For example, n is 1 for the first file, 2 for the second file, and 3 for the third file.

In this example, the following file path pattern matches all of the file path names:

- `/u/myLogDir/myLogFile.*.log`

The following scenarios provide more examples:

- [“Sample scenario that uses date and time substitution in the JCL cataloged procedure” on page 141](#)
- [“Sample scenario that uses the `redirect_server_output_dir` environment variable” on page 142](#)

File path pattern utility for verifying file path values for rolling logs

IBM Common Data Provider for z Systems includes a file path pattern utility to help you verify the file path values for any rolling logs. The utility determines which files on the current system are included by each file path pattern.

To run the utility, issue the following command in the logical partition (LPAR) where IBM Common Data Provider for z Systems runs:

```
checkFilePattern.sh configuration_directory
```

The variable `configuration_directory` represents the directory that contains both the data configuration file and the environment configuration file.

The following example further illustrates how to issue the command and includes sample values:

```
/usr/lpp/IBM/zscala/V3R1/samples/checkFilepattern.sh /usr/lpp/IBM/zscala/V3R1
```

Optionally, a data stream identifier can be specified so that the file path for only the specified data stream is checked. The following example shows that the data stream identifier 9 is specified:

```
/usr/lpp/IBM/zscala/V3R1/samples/checkFilepattern.sh /usr/lpp/IBM/zscala/V3R1 9
```

The command response is written to standard output (STDOUT). As shown in the following example, it contains a list of all files that match each file path value:

```
INFO: GLAB021I The file path pattern
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.???????.SR.?????.?????.SYSPRINT.txt
for data gatherer identifier 5 resolves to the following files:
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00036.SR.140929.170703.SYSPRINT.txt
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00158.SR.140929.193451.SYSPRINT.txt
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00252.SR.141006.134949.SYSPRINT.txt
INFO: GLAB021I The file path pattern
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.???????.SR.?????.?????.SYSOUT.txt
for data gatherer identifier 7 resolves to the following files:
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00036.SR.140929.170703.SYSOUT.txt
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00158.SR.140929.193451.SYSOUT.txt
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.STC00252.SR.141006.134949.SYSOUT.txt
```

The following example shows the command response that is written for the data stream if no files match a pattern:

```
WARNING: GLAB022W The file path pattern
/u/myLogDir/BB0CELL.BBONODE.BBOSAPP.BBOSAPPS.???????.SR.?????.?????.SYSPRINT.txt
for data gatherer identifier 6 resolves to no files.
```

Sample scenario that uses date and time substitution in the JCL cataloged procedure

Job logs can be redirected to z/OS UNIX files. They can then be rolled by using date and time substitution in the JCL cataloged procedure that is used to start the job. Each time that the job is restarted, a new file is created.

In this scenario, the following SYSOUT DD statement is from a JCL cataloged procedure is used to start a job:

```
//SYSOUT DD PATH='/u/myLogDir/myLog.&LYMMDD..&LHHMMSS..log',
//          PATHOPTS=(OWRONLY,OCREAT),PATHMODE=SIRWXU
```

The variable `&LYYMMDD`. is replaced by the local date on which the job was started, and the date is in `YYMMDD` format. Similarly, the variable `&LHHMMSS`. is replaced by the local time in which the job was started, and the time is in `HHMMSS` format.

To convert a path with date and time variables into a file path pattern for IBM Common Data Provider for z Systems configuration, replace the date and time variables with one or more wildcard characters.

For example, in this scenario, replace `&LYYMMDD`. with `??????` because the date format `YYMMDD` is always six characters. Similarly, replace `&LHHMMSS`. with `??????` because the time format `HHMMSS` is always six characters.

File path pattern for this scenario

Use the following file path pattern for this scenario:

```
/u/myLogDir/myLog.?????.?????.log
```

Sample scenario that uses the `redirect_server_output_dir` environment variable

WebSphere Application Server for z/OS `SYSOUT` and `SYSPRINT` logs can also be redirected to z/OS UNIX files and rolled by using the WebSphere environment variable `redirect_server_output_dir`.

A new set of files for `SYSOUT` and `SYSPRINT` is created for each server region at the following times:

- Each time that the server job is restarted.
- Each time that the modify command is issued with the **ROLL_LOGS** parameter.

The new files are created in the directory that is specified by the `redirect_server_output_dir` environment variable.

The following file naming conventions are used for the redirected files:

```
cellName.nodeName.serverName.jobName.jobId.asType.date.time.SYSOUT.txt  
cellName.nodeName.serverName.jobName.jobId.asType.date.time.SYSPRINT.txt
```

For each server region, the cell name, node name, server name, job name, and address space type are constant. Only the job ID, date, and time are variable.

To convert one of these file naming convention into a file path pattern for IBM Common Data Provider for z Systems configuration, complete the following steps:

1. Add the absolute path, which is specified in the WebSphere environment variable `redirect_server_output_dir`, to the beginning of the file path pattern.
2. Replace `cellName`, `nodeName`, `serverName`, and `jobName` with the appropriate values.
3. Replace `asType` with CTL (for controller), SR (for servant), or CRA (for adjunct).
4. If you are using JES2, replace `jobId` with `????????`, which matches any eight characters.

If you are using JES3, replace `jobId` with `*`, which matches any sequence of characters. In JES3, `jobId` is sometimes incorrectly populated with the job name rather than the job ID.

5. Replace `date` with `??????`, which matches any six characters.
6. Replace `time` with `??????`, which matches any six characters.

File path pattern for this scenario

The following file path pattern is an example of the pattern to use for `SYSPRINT` files for the BBOSAPP server that is using JES2:

```
/u/myLogDir/BB0CELL.BB0NODE.BB0SAPP.BB0SAPPS.????????SR.?????.?????.SYSPRINT.txt
```


Data stream configuration for data gathered by System Data Engine

This reference lists the configuration values that you can update in the "**Configure System Data Engine data stream**" window.

dataSourceName

The data source name. This value is sent to a Logstash receiver as the `Source Name` field.

Tip: If you use the **Auto-Qualify** field in the subscriber configuration to fully qualify the data source name, this **dataSourceName** value is automatically updated with the fully qualified data source name. For more information about the values that you can select in the **Auto-Qualify** field, see [“Subscriber configuration” on page 147](#).

Flavor

This field is not available for all SMF record types. If it is available, it specifies a filter (which you can select in the field) to restrict the forwarded records to those for a specific set of applications, such as CICS Transaction Server for z/OS or Db2 for z/OS applications.

Transform configuration

This reference lists and describes the transforms that you can select in the "**Transform data stream**" window. For each transform, it also lists and describes the field values that you can update in the "**Configure transform**" window.

The two categories of transform are splitter transforms and filter transforms.

Splitter transforms

Based on specified criteria, a splitter transform splits data that is received as one message into multiple messages.

Transforms in this category

- [“CRLF Splitter transform” on page 143](#)
- [“FixedLength Splitter transform” on page 144](#)

Filter transforms

Based on specified criteria, a filter transform discards messages from the data stream.

Transforms in this category

- [“Regex Filter transform” on page 145](#)
- [“Time Filter transform” on page 147](#)

CRLF Splitter transform

The **CRLF Splitter** transform splits a single message in a packet into multiple messages, based on occurrences of a carriage return (CR) character, a line feed (LF) character, or any contiguous string of these two characters. The transform also considers the packet encoding as it determines whether characters in the message are carriage return or line feed characters.

The transform splits data according to the following delimiters, among others:

- CR
- LF
- CRLF
- LFCR
- CRCR
- LFLF
- CRLFCRLF
- LFCRLFCR

Configuration values that you can update

For the **CRLF Splitter** transform, you can update the following field values in the "**Configure Splitter transform**" window:

Inspect

Specifies whether, and at what stage, data packets in the data stream are to be inspected. For example, during transform processing, the data packets can be inspected by printing them to the z/OS console at the input stage, the output stage, or both stages.

You can choose any of the following values. The default value is None. To prevent the sending of large volumes of data to the z/OS console and to the IBM Common Data Provider for z Systems Data Streamer job log, use the default value, unless you are instructed by IBM Software Support to change this value for troubleshooting purposes.

None

Specifies that data packets are not inspected.

Input

Specifies that data packets are printed to the z/OS console before they are processed by the transform.

Output

Specifies that data packets are printed to the z/OS console after they are processed by the transform.

Both

Specifies that data packets are printed to the z/OS console both before and after they are processed by the transform.

Emit

Specifies the maximum number of messages to be included in an outgoing data packet. For example, if an incoming data packet has 15 messages, and this **Emit** value is set to 6, each outgoing data packet must have no more than 6 messages. In this example, the incoming data packet must be split into three outgoing data packets, where two data packets have 6 messages, and one data packet has 3 messages.

The default value is 0, which indicates that all messages in an incoming data packet are included in the outgoing data packet.

Ignore Character

Specifies a character that, if found at the beginning of a data record, causes the record to be ignored and not included in the outgoing data packet.

This field is optional and is blank by default.

FixedLength Splitter transform

The **FixedLength Splitter** transform splits data records that have a fixed record length into multiple messages, based on configuration values that you provide.

Configuration values that you can update

For the **FixedLength Splitter** transform, you can update the following field values in the "**Configure Splitter transform**" window:

Inspect

Specifies whether, and at what stage, data packets in the data stream are to be inspected. For example, during transform processing, the data packets can be inspected by printing them to the z/OS console at the input stage, the output stage, or both stages.

You can choose any of the following values. The default value is None. To prevent the sending of large volumes of data to the z/OS console and to the IBM Common Data Provider for z Systems Data Streamer job log, use the default value, unless you are instructed by IBM Software Support to change this value for troubleshooting purposes.

None

Specifies that data packets are not inspected.

Input

Specifies that data packets are printed to the z/OS console before they are processed by the transform.

Output

Specifies that data packets are printed to the z/OS console after they are processed by the transform.

Both

Specifies that data packets are printed to the z/OS console both before and after they are processed by the transform.

Start Offset

Specifies the starting point of each data record. This value is required.

Fixed Length

Specifies the expected length of the incoming data record. This value is required.

Skip

Specifies the number of bytes from the incoming data record to skip, which means that these bytes are excluded from the output message. This value is required.

Emit

Specifies the maximum number of messages to be included in an outgoing data packet. For example, if an incoming data packet has 15 messages, and this **Emit** value is set to 6, each outgoing data packet must have no more than 6 messages. In this example, the incoming data packet must be split into three outgoing data packets, where two data packets have 6 messages, and one data packet has 3 messages.

The default value is 0, which indicates that all messages in an incoming data packet are included in the outgoing data packet.

Ignore Character

Specifies a character that, if found at the beginning of a data record, causes the record to be ignored and not included in the outgoing data packet.

This field is optional and is blank by default.

Regex Filter transform

The **Regex Filter** transform filters messages in the data stream according to a regular expression (regex) pattern, which you can define. You also define the filter to either accept or deny incoming messages based on the regular expression. For example, if an incoming message contains the regular expression, and you define the filter to deny incoming messages based on the regular expression, the filter then discards any incoming messages that contain the regular expression.

To use this transform, you must know how to use regular expressions. The [Oracle documentation about regular expressions](#) is one source of reference information.

Important: The use of complex regular expressions can result in increased usage of system resources.

Configuration values that you can update

For the **Regex Filter** transform, you can update the following field values in the "**Configure Filter transform**" window:

Inspect

Specifies whether, and at what stage, data packets in the data stream are to be inspected. For example, during transform processing, the data packets can be inspected by printing them to the z/OS console at the input stage, the output stage, or both stages.

You can choose any of the following values. The default value is None. To prevent the sending of large volumes of data to the z/OS console and to the IBM Common Data Provider for z Systems Data

Streamer job log, use the default value, unless you are instructed by IBM Software Support to change this value for troubleshooting purposes.

None

Specifies that data packets are not inspected.

Input

Specifies that data packets are printed to the z/OS console before they are processed by the transform.

Output

Specifies that data packets are printed to the z/OS console after they are processed by the transform.

Both

Specifies that data packets are printed to the z/OS console both before and after they are processed by the transform.

Regex

Specifies one or more valid regular expressions. At least one regular expression must be defined for this transform. You can also select the check box for any of the following expression flags:

Case Insensitive

Enables case-insensitive matching, in which only characters in the US-ASCII character set are matched.

To enable Unicode-aware, case-insensitive matching, select both the **Unicode Case** flag and the **Case Insensitive** flag.

Comments

Permits white space and comments in the regular expression. In this mode, white space is ignored, and any embedded comment that starts with the number sign character (#) is ignored.

Dotall

Enables dotall mode in which the "dot" expression (.) matches any character, including a line terminator.

Multi Line

Enables multiline mode in which the caret expression (^) and the dollar sign expression (\$) match immediately after, or immediately before, a line terminator or the end of the message.

Unicode Case

Enables Unicode-aware case folding.

To enable Unicode-aware, case-insensitive matching, select both the **Unicode Case** flag and the **Case Insensitive** flag.

Unix Lines

Enables UNIX lines mode in which the "dot" expression (.), the caret expression (^), and the dollar sign expression (\$) are interpreted only as the line feed (LF) line terminator.

To define one or more regular expressions in the **Regex** field, complete the following steps:

1. Type a regular expression in the **Regex** field, and optionally, select one or more check boxes to define the matching modes.
2. To add another regular expression, click **ADD REGEX**, and repeat the previous step.

Filter Type

Specifies whether the filter keeps or discards incoming messages that contain the regular expression.

You can choose either of the following values. The default value is Accept.

Accept

Specifies that any messages that contain the regular expression are kept in the data stream.

Deny

Specifies that any messages that contain the regular expression are discarded from the data stream.

Time Filter transform

The **Time Filter** transform filters messages in the data stream according to a specified schedule, which you can define.

This filter discards messages that are not received within a time interval (or time window) that is defined in the schedule.

Configuration values that you can update

For the **Time Filter** transform, you can update the following field values in the "**Configure Filter transform**" window:

Inspect

Specifies whether, and at what stage, data packets in the data stream are to be inspected. For example, during transform processing, the data packets can be inspected by printing them to the z/OS console at the input stage, the output stage, or both stages.

You can choose any of the following values. The default value is **None**. To prevent the sending of large volumes of data to the z/OS console and to the IBM Common Data Provider for z Systems Data Streamer job log, use the default value, unless you are instructed by IBM Software Support to change this value for troubleshooting purposes.

None

Specifies that data packets are not inspected.

Input

Specifies that data packets are printed to the z/OS console before they are processed by the transform.

Output

Specifies that data packets are printed to the z/OS console after they are processed by the transform.

Both

Specifies that data packets are printed to the z/OS console both before and after they are processed by the transform.

Schedule

To define a new schedule with one or more time intervals, complete the following steps:

1. For this field value, select **Create a new schedule**, and click **OK**.
2. In the **Edit name** field of the resulting **Schedules** window, type the name for the schedule that you want to contain this time interval.
3. To set the time interval for this schedule, either type the time information in the **From** and **to** fields, or use the slider to adjust the time.
4. To add another time interval for this schedule, click **ADD WINDOW**, and repeat the previous step.
5. To save the schedule, click **APPLY**.

For more information about how to define or update schedules in a policy, see [“SCHEDULES properties: Defining time intervals for filtering operational data”](#) on page 95.

Subscriber configuration

This reference lists the configuration values that you can update in the "**Configure subscriber**" window.

Name

The name of the subscriber.

Description

An optional description for the subscriber.

Protocol

The streaming protocol that the Data Streamer uses to send data to the subscriber.

You can choose any of the following values, which are organized under the applicable subscriber:

Logstash

IZOA on IOA-LA via Logstash

The protocol for sending data to IBM Z Operations Analytics on IBM Operations Analytics - Log Analysis via Logstash, without encryption.

IZOA on IOA-LA via Logstash SSL

The protocol for sending data to IBM Z Operations Analytics on IBM Operations Analytics - Log Analysis via Logstash, with encryption. If you want to have secure communications between the Data Streamer and Logstash, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

IZOA on Elasticsearch via Logstash

The protocol for sending data that is supported by IBM Z Operations Analytics to Elasticsearch via Logstash, without encryption.

IZOA on Elasticsearch via Logstash SSL

The protocol for sending data that is supported by IBM Z Operations Analytics to Elasticsearch via Logstash, with encryption. If you want to have secure communications between the Data Streamer and Logstash, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

CDP Elasticsearch via Logstash

The protocol for sending data to Elasticsearch via Logstash, without encryption.

CDP Elasticsearch via Logstash SSL

The protocol for sending data to Elasticsearch via Logstash, with encryption. If you want to have secure communications between the Data Streamer and Logstash, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

CDP Logstash

The protocol for sending data to a Logstash subscriber, without encryption.

CDP Logstash SSL

The protocol for sending data to a Logstash subscriber, with encryption. If you want to have secure communications between the Data Streamer and Logstash, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

Data Receiver

IZOA on Splunk via Data Receiver

The protocol for sending data that is supported by IBM Z Operations Analytics to Splunk via Data Receiver, without encryption.

IZOA on Splunk via Data Receiver SSL

The protocol for sending data that is supported by IBM Z Operations Analytics to Splunk via Data Receiver, with encryption. If you want to have secure communications between the Data Streamer and Data Receiver, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

CDP Splunk via Data Receiver

The protocol for sending data to Splunk via Data Receiver, without encryption.

CDP Splunk via Data Receiver SSL

The protocol for sending data to Splunk via Data Receiver, with encryption. If you want to have secure communications between the Data Streamer and Data Receiver, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

HTTP Event Collector (HEC) of Splunk

CDP Splunk via HEC via HTTP

The protocol for sending data to Splunk HTTP Event Collector via HTTP, without encryption.

CDP Splunk via HEC via HTTPS

The protocol for sending data to Splunk HTTP Event Collector via HTTPS, with encryption. If you want to have secure communications between the Data Streamer and Splunk, use this value. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

Important: A cdpkey file is generated in the Configuration Tool working directory to store the key for encrypting the HEC token in the policy file. The cdpkey file must be in the same directory as the policy files, which means if you copy the policy files to other directories, you must copy the cdpkey file together. If the cdpkey file is damaged or deleted, you must restart the Configuration Tool to generate a new one. After that, you must provide token values to the Splunk subscribers in the policies again, and save the changes.

You must not send the cdpkey file to anyone including IBM personnel.

Generic HTTP or HTTPS subscriber

CDP Generic HTTP

The protocol for a generic HTTP subscriber, which does not provide encryption.

CDP Generic HTTPS

The protocol for a generic HTTPS subscriber, which provides encryption. You must also complete the relevant configuration steps that are described in [“Securing communications between the Data Streamer and its subscribers”](#) on page 56.

Tip: For more information about preparing your target destinations to receive data from the IBM Common Data Provider for z Systems Data Streamer, see [“Preparing the Common Data Provider and the target destinations to stream and receive data”](#) on page 59.

Host

The host name or IP address of the subscriber.

Port

The port on which the subscriber listens for data from the Data Streamer.

URL Path

This field is available only if the subscriber is a generic HTTP or HTTPS subscriber. It specifies the path that is used to create the URL for the subscriber. For example, if the subscriber **Host** value is `logstash.myco.com`, the **Port** value is 8080, and the **URL Path** value is `/myapp/upload/data`, the following URL is created for the subscriber:

```
http://logstash.myco.com:8080/myapp/upload/data
```

Auto-Qualify

A specification of whether to prepend system names or sysplex names to data source names in the data streams that are sent to the subscriber. The data source name is the value of the **dataSourceName** field in the data stream configuration.

If you use the same policy file for multiple systems within one sysplex, the data source names must be unique across all systems in that sysplex. If you use the same policy file for multiple sysplexes, the data source names must be unique across all systems in all sysplexes. You can use this field to fully qualify these data source names.

You can choose any of the following values. The default value is None.

None

Indicates that the data source name from the **dataSourceName** field in the data stream configuration is used.

System

Specifies that the system name and the data source name are used in the following format:

```
systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple systems within one sysplex, you might want to use the System value.

Sysplex

Specifies that the sysplex name, system name, and data source name are used in the following format:

```
sysplexName-systemName-dataSourceName
```

systemName represents the name of the system on which the IBM Common Data Provider for z Systems runs. *sysplexName* represents the name of the sysplex in which the IBM Common Data Provider for z Systems runs.

If you use the same policy file for multiple sysplexes, you might want to use the Sysplex value.

For more information about the **dataSourceName** field in the data stream configuration, see the following topics:

- [“Data stream configuration for data gathered by Log Forwarder” on page 116](#)
- [“Data stream configuration for data gathered by System Data Engine” on page 143](#)

Number of threads

The number of threads that will send data to the subscriber. The default value is 12. The value must range from 1 to 20. Generally you don't need to change this value.

Token

Specifies the token value. For more information about how to create a token value, see [“Preparing to send data to Splunk via the HTTP Event Collector” on page 62](#).

Language reference for System Data Engine

You can use the IBM Common Data Provider for z Systems System Data Engine language to specify how you want the System Data Engine to collect and process data. This reference lists and describes the language elements.

In System Data Engine language statements, expressions are used to specify calculations for processing the data in data streams. Simple expressions include a single identifier, a constant, or both, and an operator, but you can also specify more complex calculations. An expression that specifies a value of true or false is called a *condition*.

Language overview

Before you can use IBM Common Data Provider for z Systems System Data Engine language statements to create custom definitions, you must understand the concept of constants, data types, expressions, conditions, and functions.

Constants

You can specify a value explicitly by writing a string constant, an integer constant, or a floating-point constant.

String constant

A string constant is a sequence of zero or more characters enclosed within apostrophes ('). The sequence can contain any characters. You must add one apostrophe (') before the sequence and another apostrophe (') after the sequence. See the following examples:

```
'A 2'  
'a:b'  
''
```

A string constant represents the character string within the enclosing apostrophes. Therefore, the first two constants in the example represent the strings A 2 and a : b. The first string contains a blank in the middle. The last example is a sequence of zero characters that represents an empty string.

A string constant might contain sequences of double-byte characters, each enclosed between shift-out (SO, x'0E' in EBCDIC) and shift-in (SI, x'0F' in EBCDIC) characters. The apostrophes are single-byte characters and are recognized outside a double-byte sequence.

The maximum length of a string that is represented by a string constant is 254 bytes, which includes any shift-out and shift-in characters that enclose sequences of double-byte characters.

Integer constant

An integer constant is a sequence of one or more digits. See the following examples:

```
-127  
0  
5  
32767  
720176  
0000000015
```

An integer constant represents a whole number in decimal notation. The number must be no greater than 2,147,483,647, and no smaller than -2,147,483,648. The maximum length of a constant is 32 characters.

Floating-point constant

A floating-point constant is a sequence of one or more digits followed by a decimal point with zero or more digits, and optionally followed by an E and a signed or unsigned number of at most 2 digits. See the following examples:

```
25.5  
1000.  
0.0  
37589.33333  
15E1  
2.5E5  
2.2E-1  
5.E+22
```

A floating-point constant represents a 64-bit floating-point number of S/390® architecture. The number is represented in decimal notation, with *Enn* meaning multiplied by 10 to power *nn*. For example, 2.5E5 means 2.5×10^5 , and 2.2E-1 means 2.2×10^{-1} . The specified value is rounded to the closest value that can be represented as a 64-bit floating-point number.

The number must not exceed 16^{63} - 16^{49} , which is approximately 7.2E75. The smallest value different from 0 is 16^{-65} , which is approximately 54.E-79. The maximum length of a constant is 32 characters.

Integer constants or floating-point constants can represent non-negative numbers only. To represent a negative number, add a minus operator (-) in front of the constant.

Comments

To explain your text, use comments to add explanations that are ignored by the System Data Engine.

Line comment

A line comment is any sequence of characters that start with a double minus sign (--) to the end of the current input line. See the following examples:

```
-- This is a line comment.  
-- Another line comment. Notice that it may contain unpaired ' and ".
```

The comment can contain sequences of double-byte characters that are enclosed between shift-out (SO, x'0E' in EBCDIC) and shift-in (SI, x'0F' in EBCDIC) characters. The line must end in a single-byte sequence to end the line comment. If the line ends in a double-byte sequence, the next line is interpreted as starting in the single-byte mode, which usually causes an error.

Block comment

A block comment is any sequence of characters that start with slash asterisk (/*) to the nearest asterisk slash (*). See the following example:

```
/* This is a block comment.  
Notice that it can extend over several lines.  
It can contain -- and unpaired ' or " */
```

The comment can contain sequences of double-byte characters that are enclosed between shift-out (SO, x'0E' in EBCDIC) and shift-in (SI, x'0F' in EBCDIC) characters. The asterisk and slash that end the comment are single-byte characters and is only recognized outside a double-byte sequence.

Statements

The input in the System Data Engine is a sequence of statements. The statements must be separated by semicolons (;). The semicolons are not considered a part of the statement and are not shown in syntax diagrams.

Data types

The main task of the System Data Engine is to process data. The smallest unit of data is called a value. There are different types of values that can be obtained from a field of a record, stated in your definition, or computed from other values.

The System Data Engine can handle the following types of values:

- Integer numbers
- Floating-point numbers
- Character strings
- Dates
- Times
- Timestamps

The integer numbers and floating-point numbers are called numbers, or numeric values. The dates, times, and timestamps are called date and time values.

Integer numbers

An integer is a number in the range -2,147,483,648 to 2,147,483,647. For more information, see [“Integer constant” on page 151](#).

Floating-point numbers

A floating-point number is a number that can be represented as a 64-bit floating-point number of S/390 architecture. For more information, see [“Floating-point constant” on page 151](#).

Dates

A date value represents a day according to the Gregorian calendar. This value consists of three parts for day, month, and year. The range of the year part is 1 - 9,999. The range of the month part is 1 - 12.

The range of the day part is 1 to x, where x depends on the month. All dates are calculated under the condition that the Gregorian calendar was in effect since year 0001.

Times

A time value represents a time of day under a 24-hour clock. This value consists of four parts for hour, minute, second, and microsecond. The range of the hour part is 0 - 24, the range of the minute and second part is 0 - 59, and the range of the microsecond part is 0 - 999,999. If the hour is 24, the other parts must be 0.

Timestamps

A timestamp value represents a day and a time of that day. This value consists of seven parts for year, month, day, hour, minute, second, and microsecond. The year, month, and day parts represent the day as specified under [“Dates” on page 152](#). The hour, minute, second, and microsecond parts represent the time as specified under [“Times” on page 153](#).

Date and time strings

Date and time strings are character strings of a specific format, and are used to write specific date and time values.

To write specific date and time values, you must write expressions explicitly. The following expressions are specific cases of function calls with date and time strings.

```
DATE('2000-06-27')
TIME('10.32.55.123456')
TIMESTAMP('2000-06-27-10.32.55.123456')
```

The character strings '2000-06-27', '10.32.55.123456', and '2000-06-27-10.32.55.123456' are date and time strings.

Date string

Is a character string that represents a date in the format yyyy-mm-dd where yyyy is the year, mm is the month, and dd is the day.

The DATE function converts a date string to a date. The expression like DATE(*date_string*) specifies the result of such conversion. For example, the expression DATE('2000-06-27') specifies the date June 27, 2000.

Time string

Is a character string that represents a time in the format hh.mm.ss.uuuuuu, where hh is the hour, mm is the minute, ss is the second, and uuuuuu is the microsecond.

The TIME function converts a time string to a time. The expression like TIME(*time_string*) specifies the result of such conversion. For example, the expression TIME('10.32.55.123456') specifies the time 10 hours 32 minutes 55.123456 seconds.

Timestamp string

Is a character string that represents a timestamp in the format yyyy-mm-dd-hh.mm.ss.uuuuuu where yyyy, mm, dd, hh, mm, ss, and uuuuuu are as above.

The TIMESTAMP function converts a timestamp string to a timestamp. The expression like TIMESTAMP(*timestamp_string*) specifies the result of such conversion. For example, the expression TIMESTAMP('2000-06-27-10.32.55.123456') specifies the timestamp 10 hours 32 minutes 55.123456 seconds on June 27, 2000.

In some cases you can use a date and time string instead of a date and time value, and the System Date Engine converts the string for you. For example, if CREATION_DATE specifies a date, you can use CREATION_DATE < '2000-06-25'. The System Data Engine converts the date string to a date value and compares the result with the date that is specified by CREATION_DATE.

Operators

You can specify values by using arithmetic operations on numbers, comparisons, and logical operations. These operations are specified by an infix operator (which is an operator between operands), or by a prefix operator (which is an operator in front of operands).

Arithmetic operations

You can apply the prefix operator plus (+) or minus (-) to any numeric value. See the following example:

```
-DOWN_TIME  
+40  
-23.456  
-1E8
```

The result is of the same type as the operand. The prefix plus (+) does not change its operand. The prefix minus (-) reverses the sign of its operand.

You can apply the infix operator plus (+), minus (-), multiply (*), and divide (/) between any pair of numeric values. See the following example:

```
A+B  
N_DATASETS-5  
COUNT*1E-6  
RESP_TIME/60
```

The result depends on the operand types:

- If both operands are integers, the result is an integer. The operation is performed using integer arithmetic. The division is performed so that the remainder has the same sign as the dividend.
- If both operands are floating-point numbers, the result is a floating-point number. The operation is performed using long floating-point operations of S/390.
- If one of the operands is an integer and the other a floating-point number, the integer is converted to a floating-point number. The operation is then performed on the result of the conversion, using floating-point arithmetic. The result is a floating-point number.

The result of dividing an integer by another integer is also an integer. For example, if RESP_TIME is an integer less than 60, the result of RESP_TIME/60 is 0. If you want the exact result, write RESP_TIME/60.0 instead, which makes the operand on the right the floating-point number 60.0. Then the operand on the left, which is RESP_TIME, is converted to a floating-point number, making the result a floating-point number.

For all operators, the result is null if any of the operands is null. If the result is an integer, the result must be within the range of integers. If the result is a floating-point number, the result must be within the range of floating-point numbers. The operand on the right of a divide operator must not be 0.

Comparisons

You can compare two values by using the infix operator equal (=), not equal (<>), greater than (>), less than (<), greater than or equal (>=), less than or equal (<=). The result is a value of true or false. If one of the values in the comparison is null, the result is unknown. See the following example:

```
A>10  
JOB_NAME<'ABC'  
DATE<>'1993.04-15'
```

Only the following types of comparisons are allowed:

- Numbers with numbers
- Character strings with character strings or date and time values
- Date and time values with character strings or date and time values of the same type

Numbers are compared by their algebraic values. If both numbers are floating-point numbers, they are compared by using long floating-point operation of S/390. Two floating-point numbers are considered equal only if their normalized forms have identical bit configurations.

If one of the numbers is an integer number and the other a floating-point number, the integer number is converted to a floating-point number. The comparison is then performed with the result of the conversion.

Character strings are compared byte by byte, left to right. If the strings are different in length, the comparison is made with a temporary copy of the shorter string that placed on the right with blanks so that it has the same length as the other string.

Two strings are equal if they are both empty or if all corresponding bytes are equal. Otherwise, their relation is determined by the comparison of the first unequal pair of bytes.

When a character string is compared with a date and time value, it must be a valid date and time string of the corresponding kind. The string is converted to a date and time value and the comparison is performed on the result.

All comparisons of date and time values are chronological. The value that represents the later point of time is considered to be greater.

Because the hour part may range from 0 to 24, certain pairs of different timestamps represent the same time. When such timestamps are compared, the one with a greater date part is considered greater. For example, the result of this comparison is true:

```
TIMESTAMP('1985-02-23-00.00.00.000000') >  
TIMESTAMP('1985-02-22-24.00.00.000000')
```

However, the INTERVAL calculates the interval between these timestamps as 10.

Logical operations

You can apply the prefix operator NOT to any value of true or false. The following table shows the result that is defined for operand p:

Table 19. Logical operation NOT	
p	NOT p
True	False
False	True
Unknown	Unknown

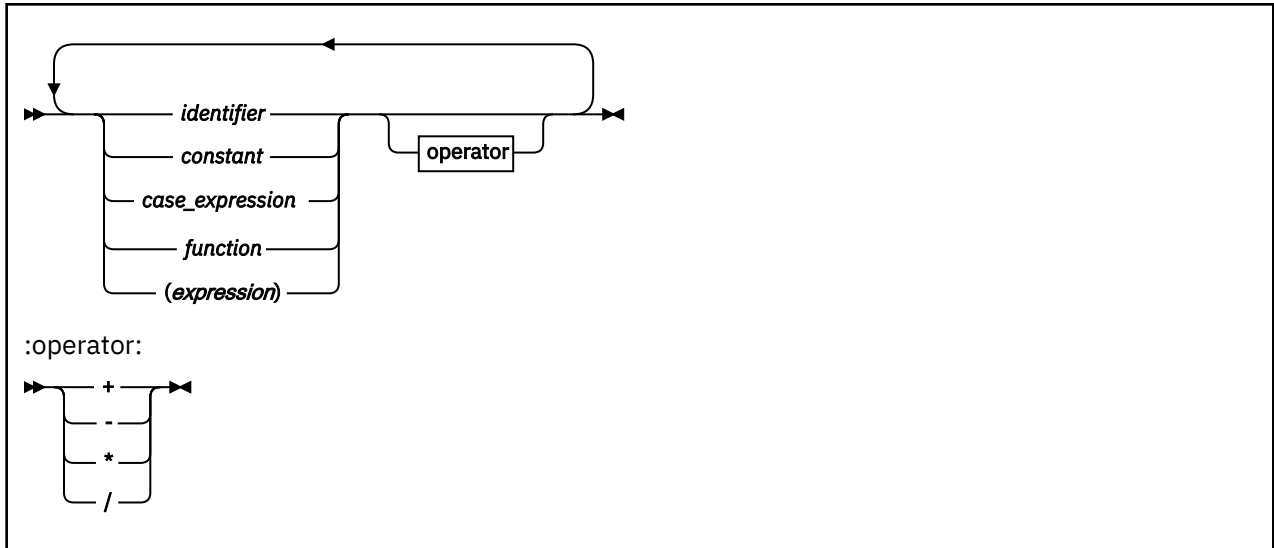
You can apply the infix operator AND and OR to any pair values of true or false. The following table shows the result that is defined for operand p and q:

Table 20. Logical operations AND and OR			
p	q	p AND q	p OR q
True	True	True	True
True	False	False	True
True	Unknown	Unknown	True
False	True	False	True
False	False	False	False
False	Unknown	False	Unknown
Unknown	True	Unknown	True
Unknown	False	False	Unknown
Unknown	Unknown	Unknown	Unknown

Expressions

In IBM Common Data Provider for z Systems System Data Engine language statements, expressions are used to specify calculations for processing the data.

The following syntax shows the general form of expression that you can use wherever the syntax specifies an expression. The diagram does not reflect all the rules that you must follow when you use operators.



identifier

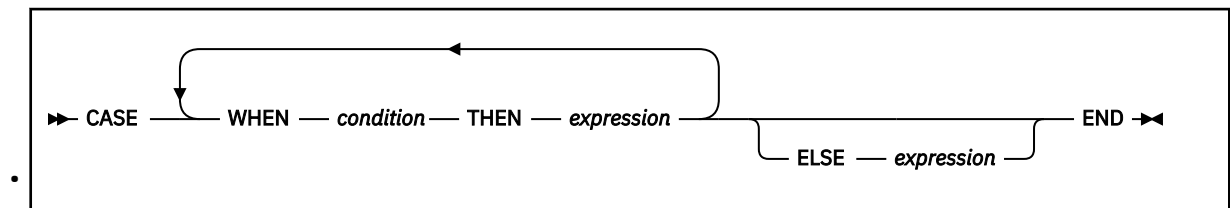
Specifies the name of a value, or the name of something that holds a value. In general, identifiers are used as names of logs, records, and fields within records. An identifier must not exceed 18 bytes.

constant

Specifies a value explicitly. You can specify a value by writing an integer constant, a floating-point constant, or a string constant. For more information, see [“Constants” on page 150](#).

case_expression

Is a case expression that specifies a value that is selected by testing one or more conditions. It has the following syntax.

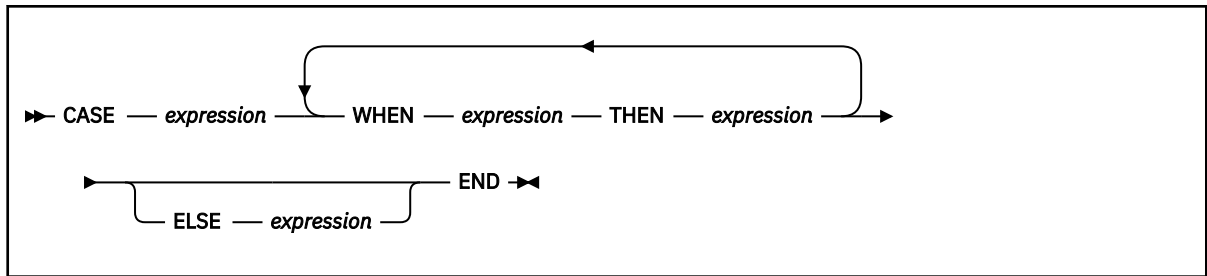


WHEN *condition* THEN *expression*

Defines one of the possible cases. The case is applicable if the value of *condition* is true. The result of the case expression is equal to the result of the expression in the applicable case. If several cases are applicable, the result is defined by the first one. If none of the cases is applicable, the result is defined by the ELSE clause. The expressions in all case expressions must specify values of the same type.

ELSE *expression*

Defines the result of the case expression if no case is applicable. If the ELSE clause is not specified, the result is null. The expression in the ELSE clause must specify a value of the same type as expressions in the case expressions.



CASE *expression*

The expression is evaluated and the result is compared with results of expressions in the WHEN clauses.

WHEN *expression* THEN *expression*

Defines one of the possible cases. The case is applicable if the result of WHEN *expression* is equal to the result of CASE *expression*. The result of the case expression is equal to the result of THEN *expression* in the applicable case. If several cases are applicable, the result is defined by the first one. If none of the cases is applicable, the result is defined by the ELSE clause.

All expressions in the WHEN clause must specify values of the same type as the expression in the CASE clause.

All expressions in the THEN clause must specify values of the same type.

ELSE *expression*

Defines the result of the case expression if no case is applicable. If the ELSE clause is not specified, the result is null. The expression in the ELSE clause must specify a value of the same type as expressions in the THEN clause.

function

Is a function call. It specifies a value as the result of a function. For more information, see [“Functions” on page 158](#).

(*expression*)

Specifies the value of expression. You can combine the expressions by using operators and parenthesis.

Operator

Are arithmetic operators. For more information about how to use each operator, see [“Arithmetic operations” on page 154](#).

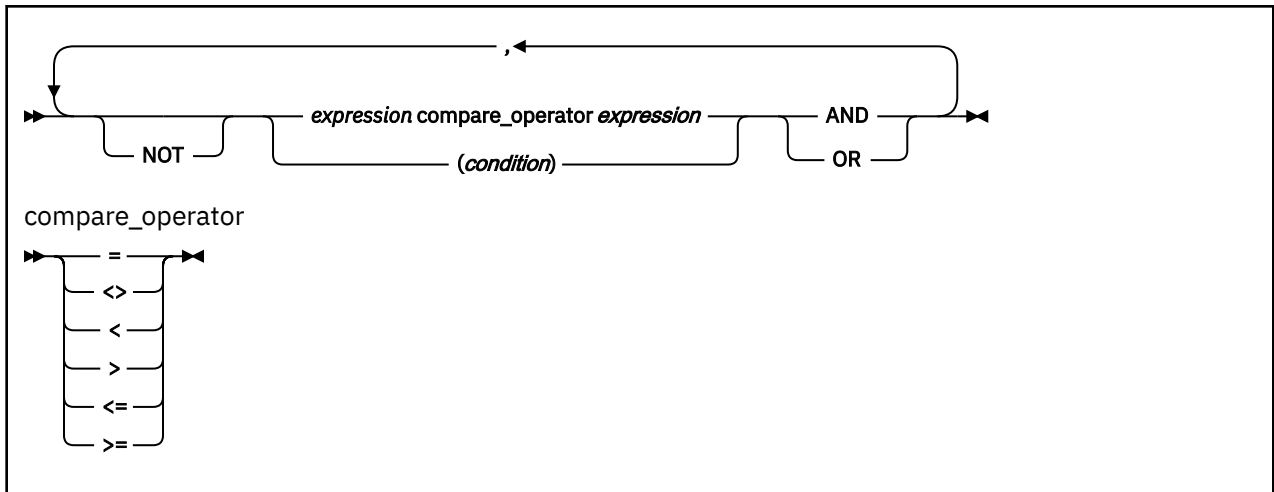
The result of an expression can be one of the following types and it must match the context.

- Integer numbers
- Floating-point numbers
- Character strings
- Dates
- Times
- Timestamps

Conditions

In IBM Common Data Provider for z Systems System Data Engine language statements, an expression that specifies a value of true or false is called a *condition*.

The following syntax shows the general form of expression that you can use wherever the syntax specifies a condition. The diagram does not reflect all the rules that you must follow when you use operators.



expression compare_operator expression

Is a comparison between expressions. The result is a value of true or false. If one of the compared values is null, the result is unknown.

compare operator

Includes equal (`=`), not equal (`<>`), greater than (`>`), less than (`<`), greater than or equal (`>=`), and less than or equal (`<=`). For more information, see [“Comparisons” on page 154](#).

AND and OR

Specifies the logical relationship between two expressions.

(condition)

You can combine the conditions by using operators and parenthesis. See the following example:

```
(SMF30XXX <> 3 AND SMF30YYY = 'A 2') OR SMF30ZZZ < 2.2E-1
```

Functions

A function call is a special form of expression. You can use a function call directly in the statements whenever the syntax specifies an expression. You can also use it as a part of more complex expressions.

CHAR

The CHAR function obtains a string representation of a date and time value. See the following syntax:

►► CHAR — (— *expression* —) ►◄

The argument *expression* must be a date, a time, or a timestamp. For more information about the date, time, and timestamp values, see [“Data types” on page 152](#).

Assume the following conditions:

- X_DATE has the value May 3, 2000.
- X_TIME has the value 5 hours, 17 minutes, and 34 seconds.
- X_TSTAMP has the value 5 hours, 17 minutes, and 34 seconds on May 3, 2000.

The CHAR function produces the following results:

```
CHAR(X_DATE) = '2000-05-03'
CHAR(X_TIME) = '05.17.34.000000'
CHAR(X_TSTAMP) = '2000-05-03-05.17.34.000000'
```

DATE

The DATE function obtains a date from a value.

►► DATE — (— *expression* —) ►◄

The argument *expression* must be a date, a timestamp, a number, or a date string.

The result of this function is a date.

- If the argument is a date, the result is that date.
- If the argument is a timestamp, the result is the date part of that timestamp.
- If the argument is a number, consider the integer part of that number as n. It must be in the range 1 - 3,652,059. The result of the function is the date of the day with sequential number n, counting from January 1, 0001 as day 1.
- If the argument is a date string, the result is the date that is represented by that string.

Assume the following conditions:

- X_DATE has the value April 22, 1993.
- X_TSTAMP has the value 15 hours, 2 minutes, and 1 second on March 6, 1993.
- X_STRING has the value '2000-03-06'.

The DATE function produces the following results:

```
DATE(X_DATE) = April 22, 1993
DATE(X_TSTAMP) = March 6, 1993
DATE(727159) = November 23, 1991
DATE('2000-06-15') = June 15, 2000
DATE(X_STRING) = March 6, 2000
```

DIGITS

The DIGITS function obtains a character string representation of a number.

►► DIGITS — (— *expression* —) ►◄

The argument *expression* must be an integer.

The result is the string of digits that represents the absolute value of the argument. Leading zeros are not included in the result.

```
DIGITS(754) = '754'
DIGITS(00054) = '54'
DIGITS(-54) = '54'
```

INTEGER

The INTEGER function obtains the integer part of a number.

►► INTEGER — (— *expression* —) ►◄

The argument must be a number.

If the argument is an integer, the result is that integer. If the argument is a floating-point number, the result is the integer part of that number.

```
INTEGER(45) = 45
INTEGER(-75.3) = -75
INTEGER(0.0005) = 0
```

INTERVAL

The INTERVAL function obtains the length of a time interval in seconds.

►► INTERVAL — (— *expression* — , — *expression* —) ►◄

Both arguments must be date and time values of the same type.

The result is a floating-point number. The result is the interval, in seconds, from the instant designated by the first argument to the instant designated by the second argument. If the first argument is later than the

second, the result is negative. The result has the maximum precision that is allowed by its floating-point representation. Therefore, results up to 2283 years have a precision of 1 microsecond. Assume the following conditions:

- TME1 has the value of 6 hours, 20 minutes, 29 seconds, and 25000 microseconds.
- TME2 has the value of 18 hours, 25 minutes, 20 seconds.
- DAY1 has the value of March 5, 1993.
- DAY2 has the value of March 8, 1993.
- TS1 has the value of 5 hours on March 5, 1993.
- TS2 has the value of 10 hours, 30 minutes on March 11, 1993.

The INTERVAL function produces the following results:

```
INTERVAL(TME1, TME2) = 43490.975
INTERVAL(TME2, TME1) = -43490.975
INTERVAL(DAY1, DAY2) = 259200.0
INTERVAL(TS1, TS2) = 538200.0
```

LENGTH

The LENGTH function obtains the length of a character string.

►► LENGTH — (— *expression* —) ►◄

The argument must be a character string.

The result of the LENGTH function is an integer. Assume that X_STRING has the value of 'LOG_NAME', the function produces the following results:

```
LENGTH(X_STRING) = 8
LENGTH('REC_LOG') = 7
LENGTH(' ') = 1
LENGTH('') = 0
```

SUBSTR

The SUBSTR function obtains a substring of a character string.

►► SUBSTR — (— *expression1* — , — *expression2* — , — *expression3* —) ►◄

The *expression1* is called string, and must be a character string. The *expression2* is called start, and must be an integer in the range 1 - 254. The *expression3* is called length, and must be an integer in the range 0 to 255-*expression2*.

The result is a character string. If length is specified, the result consists of length bytes of string, starting at the start position. The string is regarded as extended on the right with the necessary number of blanks so that the specified substring exists.

If length is not specified, the result consists of all bytes of string, starting at the position start and extending to the end of string. If start is greater than the length of string, the result is an empty string.

Both start and length are expressed in bytes. The SUBSTR function does not recognize double-byte characters, and the result is not necessarily a well-formed character string.

The function produces these results:

```
SUBSTR('SUB_REC',3,2) = 'B_'
SUBSTR('SUB_REC',3) = 'B_REC'
SUBSTR('SUB_REC',3,10) = 'B_REC'
```

TIMESTAMP

The TIMESTAMP function obtains a timestamp from a value or a pair of values.

➤ **TIMESTAMP** — (— *expression1* — , — *expression2* —) ➤

The result of the function depends on whether *expression1* or *expression2* are specified.

If only one argument is specified, *expression1* must be a timestamp or a timestamp string. The result is a timestamp:

- If *expression1* is a timestamp, the result is that timestamp.
- If *expression1* is a timestamp string, the result is the timestamp represented by that string.

If both arguments are specified, *expression1* must be a date or a date string, and *expression2* must be a time or a time string. The result is a timestamp. It consists of the date and time that is specified by the arguments. Assume the following conditions:

- X_TIME has the value 3 hours, 24 minutes, 20 seconds, and 2 microseconds.
- X_DATE has the value February 11, 1993.
- X_TSTAMP has the value 15 hours, 33 minutes, 25 seconds, and 75 microseconds on June 20, 1993.

The function produces the following results:

```
TIMESTAMP(X_TSTAMP) = 15 hours, 33 minutes, 25 seconds, and 75 microseconds on
June 20, 1993
TIMESTAMP('1993-04-17-19.01.25.000000') = 19 hours, 1 minute, 25 seconds on
April 17, 1993
TIMESTAMP(X_DATE, X_TIME) = 3 hours, 24 minutes, 20 seconds, and 2 microseconds
on February 11, 1993
```

VALUE

The VALUE function returns the first argument that is not null.

➤ **VALUE** — (— *expression* — , — *expression* — , — *expression* —) ➤

All arguments must have the same data type.

The result has the same data type as the arguments. It is equal to the first argument that is not null. If all arguments are null, the result is null.

Assume the following conditions:

- EXPA has the value of 25.
- EXPB has the value of 50.
- EXPC has a null value.

The function produces the following results:

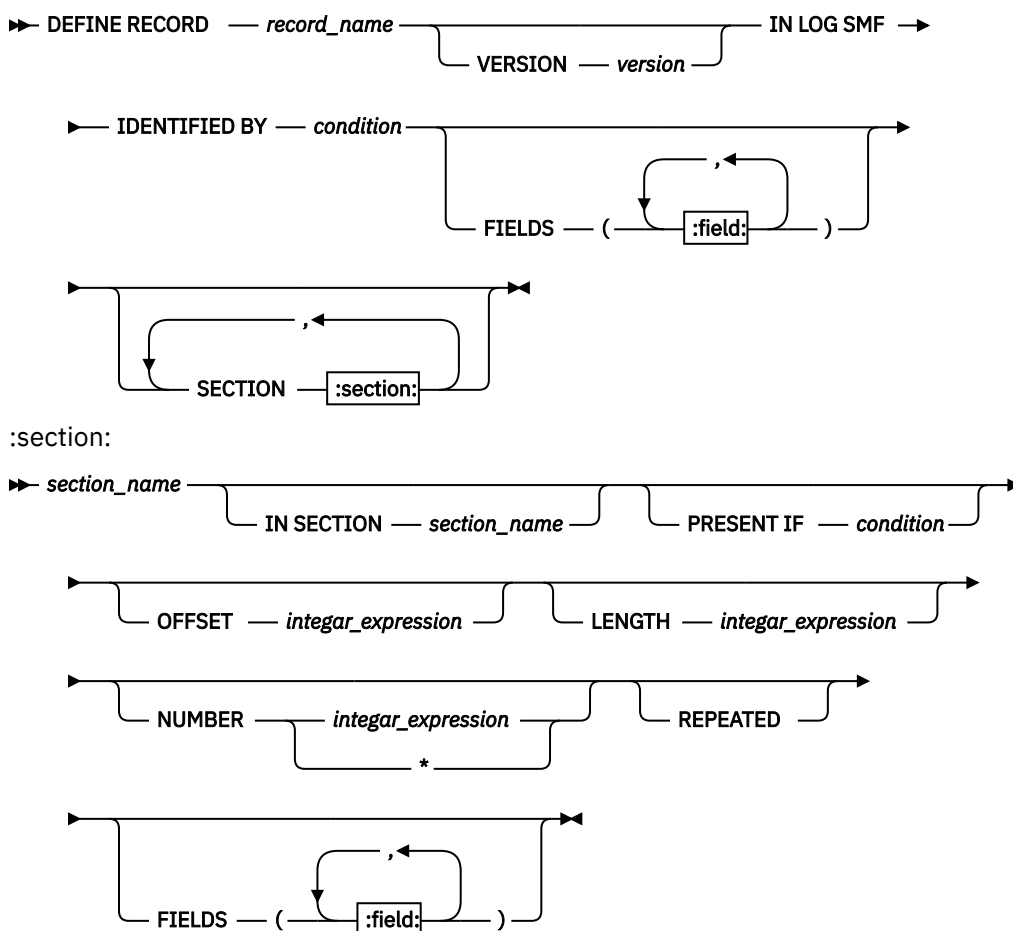
```
VALUE(EXPA, EXPB, EXPC) = 25
VALUE(EXPC, EXPB, EXPA) = 50
VALUE(EXPB, EXPA) = 50
```

DEFINE RECORD statement

The DEFINE RECORD statement defines a new record type, which is used to create a custom data stream.

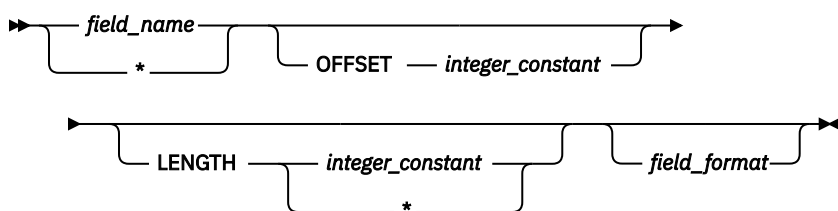
- [“Syntax” on page 162](#)
- [“Parameters” on page 162](#)
- [“SECTION clause” on page 163](#)
- [“FIELDS clause” on page 164](#)
- [“Examples” on page 169](#)

Syntax



DEFINE RECORD statement

:field:



Parameters

DEFINE RECORD *record_name*

Specifies the name of the record type that you are defining to the System Data Engine. The record name must be unique, and cannot both start and end with an asterisk (*). Verify that in the SHB0DEFS data set that is used as the CDP concatenation library, and in the data set that is used as the USER concatenation library, no existing definition has the same name.

VERSION *version*

Specifies the version for the definition. The maximum length for this value is 18 characters. You might want to specify this optional value for troubleshooting purposes.

IDENTIFIED BY

Specifies how records of this type are distinguished from each other. If a record meets the condition *condition*, it's identified as the type *record_name*. A specific record might meet the condition of

several record definitions. In this case, only one of these definitions (undefined which one) is used by the System Data Engine for processing this record.

condition

Specifies the condition according to which the records are identified. For more information, see [“Conditions” on page 157](#).

Any identifier that is used in the *condition* must be a field name that is defined directly in the record, not in the sections within the record.

SECTION clause and FIELDS clause

Define the sections and fields with options. For more information, see [“SECTION clause” on page 163](#) and [“FIELDS clause” on page 164](#).

SECTION clause

SECTION *section* defines one section. A record can have a maximum of 300 sections.

section_name

Specifies the name of the section. Section names must be unique within a record type.

IN SECTION *section_name*

Specifies that the section that is being defined is a subsection of the section named *section_name*. If you omit the IN SECTION clause, the section is a section of the record. The containing section *section_name* must be an existing section in this record definition.

PRESENT IF

Specifies that the section is optional. The section is absent if the *condition* isn't true. If the *condition* is true while the containing section or record is too short to contain the first byte of the section, the section is also absent.

condition

Specifies the condition for sections to present. For more information, see [“Conditions” on page 157](#).

Any identifier that is used in the *condition* must be a field name in one of the following areas:

- The section that is being defined
- The containing section
- The record
- The previously defined non-repeated subsections

OFFSET

Specifies the offset of the section within the containing section or record.

integer_expression

Specifies an integer value that is defined by an expression. The expression result must be an integer value such as an integer constant or a field that holds an integer value. The value must be no smaller than 0. For more information, see [“Expressions” on page 156](#).

Any identifier that is used in the expression must be a field name in one of the following areas:

- The containing section
- The record
- The previously defined non-repeated subsections

If you omit OFFSET, the section starts at the end of the most recently defined section with the same IN SECTION attribute. That section can't be a repeated section. If no section with the same IN SECTION attribute has been previously defined, an omitted OFFSET means offset 0.

LENGTH

Specifies the length of the section.

integer_expression

Specifies an integer value that is defined by an expression. The expression result must be an integer value such as an integer constant or a field that holds an integer value. The value must be no smaller than 0. For more information, see [“Expressions” on page 156](#).

Any identifier that is used in the expression must be a field name in one of the following areas:

- The section that is being defined
- The containing section
- The record
- The previously defined non-repeated subsections

If you omit LENGTH, the System Data Engine takes the minimum length needed to contain all named fields specified for this section.

If the containing section or record is too short to contain the whole section, the System Data Engine assumes that the section extends up to the end of the containing section or record. If the containing section or record is too short to contain the first byte of the section, the section is absent.

NUMBER

Specifies the number of occurrences of the section.

integer_expression

Specifies an integer value that is defined by an expression. The expression result must be an integer value such as an integer constant or a field that holds an integer value. The value must be no smaller than 0. For more information, see [“Expressions” on page 156](#).

Any identifier that is used in the expression must be a field name in one of the following areas:

- The containing section
- The record
- The previously defined non-repeated subsections

Specifies that the number of occurrences of the section is as many occurrences as the containing section or record can contain.

If you omit NUMBER clause, it has the same behavior as NUMBER 1.

REPEATED

Specifies that the section can occur more than once. If you omit REPEATED, the section can occur only once.

Tip: If the keyword REPEATED is not included, even the value of NUMBER is larger than 1, the section can occur only once.

FIELDS clause

Specifies fields of the record or section. A record, including the fields inside and outside its sections, can have a maximum of 2000 fields. Use this clause to specify one or multiple fields. Multiple *field* entries are separated by commas.

This general rule applies to all fields. The LENGTH and OFFSET (explicit or default) define a field as an area and its length, starting at a specific place in the record or section. If the record or section is too short to contain all bytes of a field, the field is absent and a reference to it produces null value.

However, if you specify LENGTH *, which means the field extends up to the end of the record or section, the previous rule doesn't apply. The field is absent if the record or section is too short to contain the first byte of the field.

field_name

Specifies the name of the field. Field names must be unique within a record type.

OFFSET *integer_constant*

Defines the offset, in bytes, of the field in the record or section.

If you omit OFFSET, the field starts at the end of the field defined just before it. The preceding field can't have LENGTH *. If you omit the OFFSET for the first field in the list, that field begins at offset 0.

LENGTH

Specifies the length of the field.

LENGTH *integer_constant*

Specifies the length of the field in bytes. The allowed lengths depend on the format of the field. See the Length in bytes column in table [Table 21 on page 165](#) for more information.

If you omit LENGTH, the System Data Engine uses the default length depending on the field format. See the Length in bytes column in table [Table 21 on page 165](#) for more information. If there is only one value in the column, it's the default.

LENGTH *

Specifies that the field extends up to the end of the containing record or section.

field_format

Specifies the format of the data in the field. The possible values of *field_format* are listed in [Table 21 on page 165](#), in the Field format column. The Data type column states the data type to which the System Data Engine automatically converts.

If you omit the *field_format*, the field format is HEX.

Table 21. Field formats for the <i>FIELDS</i> clause of the <i>DEFINE RECORD</i> statement			
Field format	Contents	Length in bytes	Data type
BINARY BINARY SIGNED BINARY UNSIGNED	Binary integer represented according to System/390® architecture. The default is SIGNED for lengths 2, 4, and 8, and UNSIGNED for lengths 1 and 3.	1, 2, 3, 4 (default), 8	Integer for SIGNED of length ≤ 4 and UNSIGNED of length ≤ 3; otherwise floating-point
EXTERNAL HEX	A string of bits representing an integer in hexadecimal characters.	2, 4, 8 (default)	String
DECIMAL(<i>p</i> , <i>s</i>) where $1 \leq p \leq 31$ and $0 \leq s \leq p$	Packed decimal number of System/390 architecture, with precision <i>p</i> and scale <i>s</i> . The precision is the total number of decimal digits. Odd <i>p</i> means a signed number; even <i>p</i> means an unsigned number. The scale is the number of digits after the decimal point.	Integer part of $(p + 1)/2$	Integer if <i>s</i> = 0 and $p \leq 9$; otherwise floating-point
ZONED(<i>p</i> , <i>s</i>) where $1 \leq p \leq 31$, and $0 \leq s \leq p$	Unsigned zoned decimal number of System/390 architecture, with precision <i>p</i> and scale <i>s</i> . The precision is the total number of decimal digits. The scale is the number of digits after the decimal point.	<i>p</i>	Integer if <i>s</i> = 0 and $p \leq 9$; otherwise floating-point
FLOAT	A floating-point number of System/390 architecture, short (4 bytes) or long (8 bytes).	4, 8 (default)	Floating-point

Table 21. Field formats for the *FIELDS* clause of the *DEFINE RECORD* statement (continued)

Field format	Contents	Length in bytes	Data type
EXTERNAL FLOAT	A string of characters expressing a floating-point number in the same format used for floating-point constants. Leading and trailing blanks are allowed.	1 to 32 while the default value is 8	Floating-point
CHAR	A string of characters. May include sequences of double-byte characters, enclosed between shift-out and shift-in characters.	1 to 254 while the default value is 1	String
CHAR(<i>n</i>) where $1 \leq n \leq 254$	A string of characters occupying <i>n</i> bytes. May include sequences of double-byte characters, enclosed between shift-out and shift-in characters.	<i>n</i>	String
CHAR(*)	A string of characters, extending up to the end of the containing structure. If the string is longer than 254 bytes, it's truncated. This format is only allowed with LENGTH *.	* (1 to 254)	String
VARCHAR	A string of characters including length information. The first two bytes contain the length <i>l</i> of the data as a binary integer; the remaining bytes contain the data itself. The length <i>l</i> may be 0, and can't exceed the length of the field minus 2. The data portion of the string may include sequences of double-byte characters, enclosed between shift-out and shift-in characters.	3 to 256 while the default value is 8	String
BIT	A string of bits. Converted to string of characters "0" and "1" and "1" representing individual bits.	1 to 31 while the default value is 1	String
BIT(<i>n</i>) where $8 \leq n \leq 248$, <i>n</i> multiple of 8	A string of <i>n</i> bits. Converted to string of characters "0" and "1" representing individual bits.	<i>n</i> /8	String
HEX	A string of bits. Converted to string of characters "0" through "F" representing the string in hexadecimal notation.	1 to 127 while the default value is 1	String

Table 21. Field formats for the *FIELDS* clause of the *DEFINE RECORD* statement (continued)

Field format	Contents	Length in bytes	Data type
DATE(0CYYDDDF)	Date in the format <i>0cyydddF</i> (packed), where <i>c</i> indicates the century (0=1900, 1=2000), <i>yy</i> is the year within the century, <i>ddd</i> is the day within the year, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Date
DATE(YYYYDDDF)	Date in the format <i>yyyydddF</i> (packed), where <i>yyyy</i> is the year, <i>ddd</i> is the day within the year, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Date
DATE(YYDDDF)	Date in the format <i>yydddF</i> (packed), where <i>yy</i> is the year, <i>ddd</i> is the day within the year, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	3	Date
DATE(CYYMMDDF)	Date in the format <i>cyyymmddF</i> (packed), where <i>c</i> indicates the century (0=1900, 1=2000), <i>yy</i> is the year within the century, <i>mm</i> is the month, <i>dd</i> is the day of month, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Date
DATE(YYMMDD)	Date as character string <i>yymmdd</i> , where <i>yy</i> is the year, <i>mm</i> is the month, and <i>dd</i> is the day. <i>yy</i> ≥50 means year 19 <i>yy</i> ; <i>yy</i> <50 means year 20 <i>yy</i> .	6	Date
DATE(MMDDYY)	Date as character string <i>mmddyy</i> , where <i>mm</i> is the month, <i>dd</i> is the day, and <i>yy</i> is the year.	6	Date
DATE(MMDDYYYY)	Date as character string <i>mmddyyyy</i> , where <i>mm</i> is the month, <i>dd</i> is the day, and <i>yyyy</i> is the year.	8	Date
TIME(1/100S)	A 32-bit binary integer representing time in hundredths of a second elapsed since hour 0.	4	Time

Table 21. Field formats for the *FIELDS* clause of the *DEFINE RECORD* statement (continued)

Field format	Contents	Length in bytes	Data type
TIME(HHMMSS T F)	Time in the format <i>hhmmssTF</i> (packed), where <i>hh</i> is hours, <i>mm</i> is minutes, <i>ss</i> is seconds, <i>t</i> is tenths of a second, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Time
TIME(OHHMMSS F)	Time in the format <i>OhhmmssF</i> (packed), where <i>hh</i> is hours, <i>mm</i> is minutes, <i>ss</i> is seconds, and <i>F</i> can have any value. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Time
TIME(HHMMSS T H)	Time in the format <i>hhmmssth</i> (packed), where <i>hh</i> is hours, <i>mm</i> is minutes, <i>ss</i> is seconds, and <i>th</i> is hundredths of a second.	4	Time
TIME(HHMMSSU6)	Time in the format <i>hhmmssuuuuuu</i> (packed), where <i>hh</i> is hours, <i>mm</i> is minutes, <i>ss</i> is seconds, and <i>uuuuuu</i> is microseconds.	6	Time
TIME(HHMMSS)	Time as character string <i>hhmmss</i> , where <i>hh</i> is hours, <i>mm</i> is minutes, and <i>ss</i> is seconds.	6	Time
INTV(MMSST T T T F)	Time duration in the format <i>mmsst$t$$t$$t$F</i> (packed), where <i>mm</i> is minutes of duration, <i>ss</i> is seconds, <i>ttt</i> is milliseconds, and <i>F</i> can have any value. The duration is converted to milliseconds and expressed as an integer. (<i>F</i> is ignored and isn't checked to be a valid decimal sign).	4	Integer
INTV(TOD)	Time duration in microseconds. If the difference of System/390 time-of-day(TOD) is smaller than 1970, it is converted to microseconds. Otherwise, the value is 0.	8	floating-point
IPADDR(IPV4)	Convert record data to IPv4 address	4	Dot separated decimal string
IPADDR(IPV6)	Convert record data to IPv6 address	16	Colon separated hex string

Table 21. Field formats for the *FIELDS* clause of the *DEFINE RECORD* statement (continued)

Field format	Contents	Length in bytes	Data type
TIMESTAMP(TOD)	Date and time in System/390 time-of-day (TOD) clock format: the number of microseconds since the start of year 1900, expressed as a binary number, with the highest bit position representing 2^{51} .	4, 8 (default)	Timestamp

Examples

The following example shows a *DEFINE RECORD* statement for a simple record.

```

DEFINE RECORD SMF_XXX_CUST
  VERSION 'CDP.110'
  IN LOG SMF
  IDENTIFIED BY SMFXXRTY = 130
  FIELDS
    (SMFXXLEN OFFSET 0   LENGTH 2 BINARY,
     SMFXXSEG OFFSET 2   LENGTH 2 BINARY,
     SMFXXFLG OFFSET 4   LENGTH 1 BIT,
     SMFXXRTY OFFSET 5   LENGTH 1 BINARY,
     SMFXXTME OFFSET 6   LENGTH 4 TIME(1/100S),
     SMFXXDTE OFFSET 10  LENGTH 4 DATE(0CYDDDF),
     SMFXXSID OFFSET 14  LENGTH 4 CHAR,
     *      OFFSET 18    LENGTH 4 CHAR,
     SMFXXSTP OFFSET 22  LENGTH 2 BINARY)
  SECTION SUBSYSTEM
    OFFSET 24
    LENGTH 12
    NUMBER 1
    FIELDS
      (SMFXXSOF OFFSET 0   LENGTH 4 BINARY,
       SMFXXSLN OFFSET 4   LENGTH 2 BINARY,
       SMFXXSON OFFSET 6   LENGTH 2 BINARY,
       SMFXXIOF OFFSET 8   LENGTH 4 BINARY)
  SECTION ACCOUNTING
    PRESENT IF SMFXXFLG > 0
    OFFSET SMFXXSOF
    LENGTH SMFXXSLN
    NUMBER SMFXXSON
    REPEATED
    FIELDS
      (SMFXXACL OFFSET 0   LENGTH 1 BINARY,
       SMFXXACT OFFSET 1   LENGTH 1 CHAR)

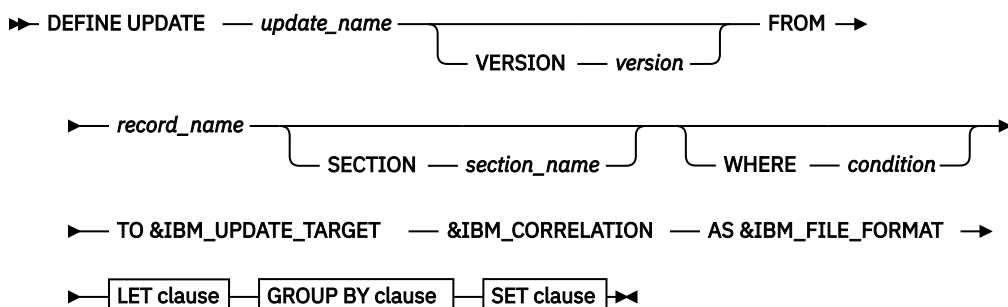
```

DEFINE UPDATE statement

The *DEFINE UPDATE* statement defines how to process data from the source record type.

- [“Syntax” on page 170](#)
- [“Parameters” on page 171](#)
- [“LET clause” on page 171](#)
- [“GROUP BY clause” on page 172](#)
- [“SET clause” on page 173](#)
- [“Examples” on page 174](#)

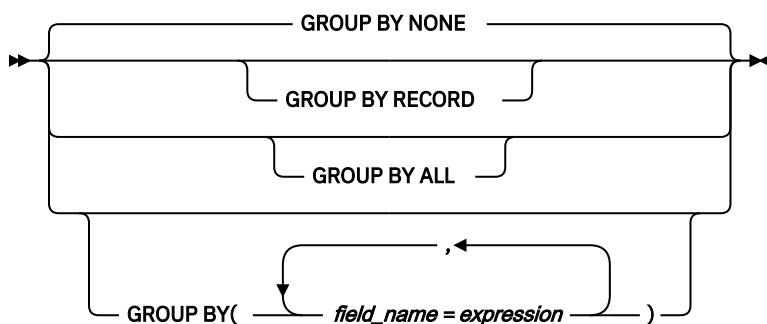
Syntax



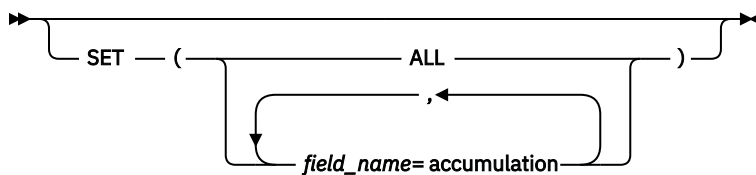
LET clause



GROUP BY clause

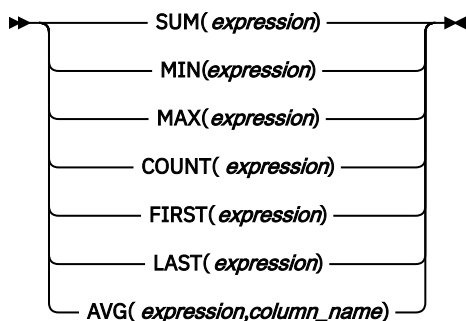


SET clause



DEFINE UPDATE statement

accumulation



Parameters

DEFINE UPDATE *update_name*

Specifies the name of the update that you are defining to the System Data Engine. The update name must be unique. Verify that in the SHBODEFS data set that is used as the CDP concatenation library, and in the data set that is used as the USER concatenation library, no existing definition has the same name.

Important: The DEFINE UPDATE statements must be included after the DEFINE RECORD statement.

VERSION *version*

Specifies the version for the update. The maximum length for this value is 18 characters. You might want to specify this optional value for troubleshooting purposes.

FROM *record_name*

Specifies the source of the data for the update. It must refer to a previously defined record.

SECTION *section_name*

Controls the processing of repeated sections. This clause specifies that the source of the update is a repeated section *section_name* of the record *record_name*. A section is repeated if the keyword REPEATED is included when it is defined. For more information about how to define a section, see [“SECTION clause” on page 163](#).

If you include this clause, the System Data Engine generates an internal record for each occurrence of the repeated section, and the source of the update is that internal record.

If the record *record_name* has repeated sections, and you omit the SECTION clause, the update can use only the data that is outside of the repeated sections, meaning that the repeated sections are not processed.

WHERE

Specifies that the update applies to only those source records or rows for which the condition that follows WHERE is true.

condition

Specifies the condition for the records or rows according to which the update applies. For more information, see [“Conditions” on page 157](#).

Any identifier that is used in the *condition* must be the name of a field in the source record.

TO &IBM_UPDATE_TARGET; &IBM_CORRELATION; AS &IBM_FILE_FORMAT

These parameters must be included as shown in the syntax.

LET, GROUP BY, and SET clauses

These clauses specify the processing to be performed by the update. These clauses includes advanced functions like GROUP BY (*field_name=expression*) and SET (*field_name=accumulation*) that are not commonly used.

The processing occurs in the order that the clauses are defined. You must specify at least one GROUP BY clause or SET clause.

The first clause in the definition uses the source records from *record_name* as input. Each subsequent clause uses the result of the preceding clause as input.

For more information about these clauses, see the following sections:

- [“LET clause” on page 171](#)
- [“GROUP BY clause” on page 172](#)
- [“SET clause” on page 173](#)

LET clause

Assigns names to expressions that are frequently used in subsequent clauses, which can simplify the DEFINE UPDATE definition and improve the efficiency of the update. For example, if you want to

calculate a value from a field, and use that value in several expressions, you can assign the result of the calculation to a name in the LET clause, and refer to that name wherever the result of the calculation is required in the definition.

identifier

Specifies the name that is assigned to the result of the expression. The name can be any identifier that is distinct from the names of fields in the source and names that are defined in the same LET clause.

expression

Specifies the expression to whose result the name is assigned. The expression can use the names that are defined in the same LET clause. For more information about how to use expressions, see [“Expressions” on page 156](#).

Any identifier that is used in this clause must be the name of a field in the source record, or a name that is introduced by this clause before.

GROUP BY clause

Organizes records in groups according to specified values. The input to this clause is the source data that is specified in the FROM clause.

The result of GROUP BY processing is groups of input records, such that all records within each group have the same grouping values. All grouping values must be non-null. A row that has a null grouping value is not included in any group.

If you omit the GROUP BY clause, all input records are processed as one group.

Important: You must specify at least one of the following elements:

- GROUP BY *group_by_option*
- SET *set_option*

GROUP BY RECORD

With this option, all the records and repeated sections are recombined into a single output record. Use GROUP BY RECORD to have each input record produce a single output record. For example, for a section SECTION_A that repeated three times, this option outputs only one record.

GROUP BY NONE

With this option, all the records and repeated sections are treated individually. If you specify GROUP BY NONE, no grouping is performed, and each input record is processed individually. For example, for a section SECTION_A that repeated three times, this option outputs three records.

Tip: If there are no repeated sections, GROUP BY RECORD and GROUP BY NONE have the same behavior.

GROUP BY ALL

With this option, all input records are processed as a single group.

GROUP BY (*field_name* = *expression*)

With this option, all input records are processed according to the following values:

field_name

Specifies the field based on which the records are grouped. The field value cannot be a decimal or a long string.

expression

Specifies one grouping value. For more information about how to use expressions, see [“Expressions” on page 156](#).

Any identifier that is used in an expression in any of the clauses must be the name of a field in the source record, or a name that is introduced in one of the preceding clauses.

You can have multiple (*field_name* = *expression*) entries, separated by commas.



CAUTION: Using the GROUP BY clause and LET clause might cause storage buffer overrun.

SET clause

Summarizes the groups of records that result from the GROUP BY clause. The SET clause produces one record in the target output for each group. In that record, the grouping values are stored in the fields that are specified in the GROUP BY clause. The values of other fields are derived from all the records in the group, as specified in the SET clause.

Important: You must specify at least one of the following elements:

- GROUP BY *group_by_option*
- SET *set_option*

SET (ALL)

Specifies that an update object is to be created that contains all fields in the source record object. If a section named TRIPLETS is present in the source record object, no fields from that section are included in the generated update object.

If SECTION *section_name* is not specified, fields in repeated sections are also omitted from the generated update object. If SECTION *section_name* is specified, all fields inside and outside of the repeated sections are included in the generated update object.

If GROUP BY RECORD is specified with SET (ALL), all fields of the generated update object are *field_name*=FIRST(*field_name*).

Important: If SET (ALL) is specified, the LET clause is not valid.

SET (*field_name*=*accumulation*)

You can have multiple (*field_name*=*accumulation*) entries, separated by commas.

field_name

Specifies a field of the target output.

accumulation

Specifies how to derive the value to be stored in the field. It can be one of the following expressions:

SUM(*expression*)

Evaluates the expression *expression* for each record in the group. The value of SUM is the sum of all non-null values that are obtained. If the value of *expression* is null for all records in the group, the value of SUM is null. The expression must specify a numerical value.

MIN(*expression*)

Evaluates the expression *expression* for each record in the group. The value of MIN is the least of all non-null values that are obtained. If the value of *expression* is null for all records in the group, the value of MIN is null.

MAX(*expression*)

Evaluates the expression *expression* for each record in the group. The value of MAX is the greatest of all non-null values that are obtained. If the value of *expression* is null for all records in the group, the value of MAX is null.

COUNT(*expression*)

Evaluates the expression *expression* for each record in the group. The value of COUNT is an integer that is the total number of non-null values that are obtained.

FIRST(*expression*)

Evaluates the expression *expression* for each record in the group, in the order in which the records are processed. The value of FIRST is the first non-null value of *expression*. If the value of *expression* is null for all records in the group, the value of FIRST is null.

LAST(*expression*)

Evaluates the expression *expression* for each record in the group, in the order in which the records are processed. The value of LAST is the last non-null value of *expression*. If the value of *expression* is null for all records in the group, the value of LAST is null.

AVG(*expression*,*field_name*)

Evaluates the expression *expression* for each record in the group. The value of AVG is the average, or weighted average, of the values that are obtained, depending on the *field_name* value, which must name a field whose value is specified in the same SET clause. The value of *field_name* must be specified by using either COUNT or SUM. If *field_name* is specified by using COUNT, its value must be equal to the number of non-null values of the expression. The result of AVG is the average of all non-null values of the expression in the group. If the value of expression is null for all records in the group, the value of AVG is null. If *field_name* is specified by using SUM, the result of AVG is the weighted average of all non-null values of expression in the group. The argument of SUM obtained for the same record is used as the weight. If the value of expression is null for all records in the group, or the sum of all weights is 0, the value of AVG is null. The expression must specify a numeric value. The result of AVG is a floating-point field.

Any identifier that is used in an expression in any of the clauses must be the name of a field in the source record, or a name that is introduced in one of the preceding clauses. For more information about how to use expressions, see [“Expressions” on page 156](#).

Examples

The following example of a DEFINE UPDATE statement specifies how the System Data Engine is to extract data from SMF record type 130 and type 140.

Example 1

```
DEFINE UPDATE SMF_130
  VERSION 'CDP.V110'
  FROM SMF_130
  WHERE (SMF130PNAME='SYSTEMA')
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT
  SET(ALL);
```

Example 2

```
DEFINE UPDATE SMF_140
  VERSION CDP.V110'
  FROM SMF_140
  WHERE (SMF140PNAME='SYSTEMA')
  TO &IBM_UPDATE_TARGET
  &IBM_CORRELATION
  AS &IBM_FILE_FORMAT
  LET (C_S_TIME = TIME(SMF140STME),
       C_Q_TIME = TIME(SMF140QTME))
  GROUP BY NONE
  SET (C_SYSTEM_ID = FIRST(SMF140SYSID),
       C_A_SYSTEM_ID = FIRST(SMF140ASYID),
       C_LAST_SEC = SUM(INTERVAL(C_S_TIME,C_Q_TIME)));
```

DEFINE TEMPLATE statement

The DEFINE TEMPLATE statement refines the update definition to include only specified fields and to change the order of these fields in the output file.

- [“Syntax” on page 175](#)
- [“Parameters” on page 175](#)
- [“Example” on page 175](#)

Syntax

DEFINE TEMPLATE statement

►► DEFINE TEMPLATE — *template_name* — FOR — *update_name* — ORDER — (→

 ↙ , ↘
 ↙ *field_name* ↘) — AS &IBM_FILE_FORMAT ►◄

Parameters

DEFINE TEMPLATE *template_name*

Specifies the name of the template. The name of the template definition must be the same as the update definition that it is associated with. The custom template definition replaces the default template definition that is created by the update definition.

Important: The DEFINE TEMPLATE statements must be included after the DEFINE UPDATE statement in the input stream because the field names are verified based on the fields in the update definition.

FOR *update_name*

Specifies the name of the update definition that this template augments.

ORDER (*field_name*,...)

Specifies the order of fields for this template. One *field_name* entry represents a field that is required for this template.

field_name

Specifies the name of the field. The field names must be unique in a template, and they must be names that are already defined in the update definition that this template is augmenting.

Important: Do not specify repeated field names.

AS &IBM_FILE_FORMAT

This parameter must be included as shown in the syntax.

Example

The following example shows a simple DEFINE TEMPLATE statement.

```
DEFINE TEMPLATE SMF_030_CUST FOR SMF_030_CUST
ORDER
  (SMF30JBN,
   SMF30PGM,
   SMF30STM,
   SMF30UIF)
AS &IBM_FILE_FORMAT;
```

Chapter 5. Operating Common Data Provider for z Systems

To operate IBM Common Data Provider for z Systems, you must know how to run (for example, start, stop, or update) key components, such as the data gatherer components (Log Forwarder and System Data Engine), the Data Streamer, and the Data Receiver.

Before you begin

Before you start IBM Common Data Provider for z Systems, verify that the z/OS environment is set up correctly for IBM Common Data Provider for z Systems to do the following tasks:

- Access the TCP/IP resolver configuration file.
- Resolve host names.

Search order for the TCP/IP resolver configuration file

For more information, see [“Verifying the search order for the TCP/IP resolver configuration file” on page 91.](#)

Host name resolution

To operate, IBM Common Data Provider for z Systems must determine the fully qualified domain name (FQDN) of the system on which it is running. Therefore, activate the networking and name resolution services that are configured in the system for use by IBM Common Data Provider for z Systems before you start IBM Common Data Provider for z Systems.

About this task

The following lists indicate the best practice for the order in which to start or stop the components:

Order in which to start the components

1. Start the data receiving components, such as the Data Receiver or Logstash.
2. Start the Data Streamer.
3. Start the data gatherer components, such as the Log Forwarder and System Data Engine.

Order in which to stop the components

1. Stop the data gatherer components, such as the Log Forwarder and System Data Engine.
2. Stop the Data Streamer.
3. Stop the data receiving components.

Running the Data Receiver

To start the IBM Common Data Provider for z Systems Data Receiver, you run a Java command with multiple parameters. A best practice is to create a shell or batch file for starting the Data Receiver.

Before you begin

To run the Data Receiver, Java Runtime Environment (JRE) 8 must be installed.

When it runs, the Data Receiver uses the configuration values of the Data Receiver properties in the `cdpdr.properties` file in the Data Receiver working directory (`CDPDR_HOME` directory). For more information about these configuration values, see [“Updating the Data Receiver properties” on page 66.](#)

About this task

When you run the Data Receiver, you can override the values in the `cdpdr.properties` file by using the following command-line parameters:

-p port

Overrides the port on which the Data Receiver listens for data from the Data Streamer.

Example

```
-p 8888
```

-c cycle

Overrides the number of output files that can simultaneously exist in the Data Receiver output directory (`CDPDR_PATH` directory).

Example

```
-c 5
```

-s ssl

Overrides the specification of whether to use the Transport Layer Security (TLS) protocol for Data Receiver communication with the Data Streamer.

Example

```
-s y
```

-t trace

Overrides the specification of whether to activate tracing for the Data Receiver.

Example

```
-t y
```

Procedure

To start the Data Receiver, run the following command:

```
java -jar -Dfile.encoding=UTF-8 DataReceiver.jar
```

The following example shows how you can use command-line parameters to override the values of the configuration properties in the `cdpdr.properties` file:

```
java -jar -Dfile.encoding=UTF-8 DataReceiver.jar -p 6767 -c 10
```

Running the Data Streamer

To start the IBM Common Data Provider for z Systems Data Streamer, you use the Data Streamer started task. When a policy is updated, you must stop and restart the Data Streamer to make the updated definitions take effect.

Before you begin

Create the Data Streamer started task, as described in [“Configuring the Data Streamer”](#) on page 68.

About this task

You use z/OS console commands to control the operation of the Data Streamer and to view information about the current policy.

Troubleshooting tip: After the Data Streamer is started, it should not stop until you stop it. If it does stop without your stopping it explicitly, review the Data Streamer job log output for possible errors.

Procedure

To run the Data Streamer, issue the following console commands, where *procname* represents the name of the started task (such as HBODSPRO).

Action	z/OS console command
Start the Data Streamer	<code>START <i>procname</i></code>
Stop the Data Streamer	<code>STOP <i>procname</i></code>
View information about the current policy	<code>MODIFY <i>procname</i>,APPL=DISPLAY,POLICY</code> The following message is sample output from the command: HB06076I The current policy is /usr/lpp/IBM/cdpz/v1r1m0/UI/LIB/Sample.policy.

Running the Log Forwarder

To start the IBM Common Data Provider for z Systems Log Forwarder, you use the Log Forwarder started task. If you are collecting NetView for z/OS message data, the NetView message provider must also be active. The NetView message provider is started as a started task by using the REXX module GLANETV.

Before you begin

Create the Log Forwarder started task, as described in [“Creating the Log Forwarder started task”](#) on page 72.

If you configure a **NetView Netlog** data stream for gathering NetView for z/OS message data, also configure the NetView message provider to monitor and forward NetView for z/OS messages to the Log Forwarder, as described in [“Configuring the z/OS NetView message provider for collecting NetView messages”](#) on page 78. You can start the REXX module GLANETV from the command line of an existing NetView user ID, or create a new NetView user ID to support the running of this REXX module. Always start the NetView message provider after the Log Forwarder is started for the first time.

If you configure a **z/OS SYSLOG** data stream for gathering z/OS SYSLOG data from a user exit, you must install either the GLASYSO or GLAMDBG user exit, as described in [“Installing the user exit for collecting z/OS SYSLOG data”](#) on page 75. The GLASYSO and GLAMDBG user exits create system resources that might need to be managed while they are in operation. The `manageUserExit` utility is a shell script that can be used to manage the system resources. For more information about this utility, see [“manageUserExit utility for managing the installed user exit”](#) on page 76.

About this task

You use z/OS console commands to control the operation of the Log Forwarder, including to start, stop, or view status or configuration information for Log Forwarder data streams.

For more information about Log Forwarder data streams, including the correlation between the sources from which the Log Forwarder gathers data and the data streams that can be defined for those sources, see [“Data stream configuration for data gathered by Log Forwarder”](#) on page 116.

Troubleshooting tip: After the Log Forwarder is started, it should not stop until you stop it. If it does stop without your stopping it explicitly, review the Log Forwarder job log output for possible errors.

Procedure

1. To run the Log Forwarder, issue the following console commands, where *procname* represents the name of the started task (such as GLAPROC).

Action	z/OS console command
Start the Log Forwarder	<p>Issue one of the following console commands:</p> <p>Warm start</p> <pre>START procname</pre> <p>A warm start resumes data collection where it previously stopped.</p> <p>Cold start</p> <pre>START procname,OPT=-C</pre> <p>A cold start starts data collection anew. Any operational data that was written while the Log Forwarder was stopped is not collected.</p> <p>Tip: If the Log Forwarder is shut down for more than a few minutes, you might want to use cold start mode to avoid having to wait for the Log Forwarder to collect accumulated data.</p>
Stop the Log Forwarder	<pre>STOP procname</pre>
View the status of all known Log Forwarder data streams	<pre>MODIFY procname,APPL=DISPLAY,GATHERER,LIST</pre>
Start, stop, or view the status or configuration information for an individual data stream	Table 22 on page 180 lists the commands, which vary depending on the source of the data stream.

Table 22. z/OS console commands for starting, stopping, or viewing status or configuration information for individual Log Forwarder data streams

Source of data stream	z/OS console commands for controlling a data stream from this source
Job log	<p>Start the data stream</p> <pre>MODIFY procname,APPL=START,GATHERER,JOBNAME=jobname,DDNAME=ddname</pre> <p>Stop the data stream</p> <pre>MODIFY procname,APPL=STOP,GATHERER,JOBNAME=jobname,DDNAME=ddname</pre> <p>View the status of the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,JOBNAME=jobname,DDNAME=ddname</pre> <p>View the configuration information for the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,JOBNAME=jobname,DDNAME=ddname</pre> <p>Usage note: If you used wildcard characters in the job name when you defined the data stream, the values of the JOBNAME and DDNAME parameters in these commands must reference the specific instance of the job log data stream. For example, you must specify JOB0011 or JOB0021 rather than JOB*1.</p>

Table 22. z/OS console commands for starting, stopping, or viewing status or configuration information for individual Log Forwarder data streams (continued)

Source of data stream	z/OS console commands for controlling a data stream from this source
z/OS UNIX log file	<p>Start the data stream</p> <pre>MODIFY procname,APPL=START,GATHERER,UNIXFILEPATH='UNIXfilepath'</pre> <p>Stop the data stream</p> <pre>MODIFY procname,APPL=STOP,GATHERER,UNIXFILEPATH='UNIXfilepath'</pre> <p>View the status of the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,UNIXFILEPATH='UNIXfilepath'</pre> <p>View the configuration information for the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,UNIXFILEPATH='UNIXfilepath'</pre> <p>Usage note: To prevent an error message, the UNIX file path must be enclosed in quotation marks.</p>
Entry-sequenced VSAM cluster	<p>Start the data stream</p> <pre>MODIFY procname,APPL=START,GATHERER,DATASET=dataset</pre> <p>Stop the data stream</p> <pre>MODIFY procname,APPL=STOP,GATHERER,DATASET=dataset</pre> <p>View the status of the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,DATASET=dataset</pre> <p>View the configuration information for the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,DATASET=dataset</pre> <p>Usage note: <i>dataset</i> represents the name of the dataset.</p>

Table 22. z/OS console commands for starting, stopping, or viewing status or configuration information for individual Log Forwarder data streams (continued)

Source of data stream	z/OS console commands for controlling a data stream from this source
z/OS SYSLOG	<p>Start the data stream</p> <ul style="list-style-type: none"> From the user exit: <pre>MODIFY procname,APPL=START,GATHERER,SYSLOG</pre> From OPERLOG: <pre>MODIFY procname,APPL=START,GATHERER,OPERLOG</pre> <p>Stop the data stream</p> <ul style="list-style-type: none"> From the user exit: <pre>MODIFY procname,APPL=STOP,GATHERER,SYSLOG</pre> From OPERLOG: <pre>MODIFY procname,APPL=STOP,GATHERER,OPERLOG</pre> <p>View the status of the data stream</p> <ul style="list-style-type: none"> From the user exit: <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,SYSLOG</pre> From OPERLOG: <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,OPERLOG</pre> <p>View the configuration information for the data stream</p> <ul style="list-style-type: none"> From the user exit: <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,SYSLOG</pre> From OPERLOG: <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,OPERLOG</pre>
IBM Tivoli NetView for z/OS messages	<p>Start the data stream</p> <pre>MODIFY procname,APPL=START,GATHERER,DOMAIN=domain</pre> <p>Stop the data stream</p> <pre>MODIFY procname,APPL=STOP,GATHERER,DOMAIN=domain</pre> <p>View the status of the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,DOMAIN=domain</pre> <p>View the configuration information for the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,DOMAIN=domain</pre> <p>Usage note: <i>domain</i> represents the name of the NetView domain.</p>

Table 22. z/OS console commands for starting, stopping, or viewing status or configuration information for individual Log Forwarder data streams (continued)

Source of data stream	z/OS console commands for controlling a data stream from this source
IBM WebSphere Application Server for z/OS HPEL log	<p>Start the data stream</p> <pre>MODIFY procname,APPL=START,GATHERER,HPELDIRECTORY='hpeldirectory'</pre> <p>Stop the data stream</p> <pre>MODIFY procname,APPL=STOP,GATHERER,HPELDIRECTORY='hpeldirectory'</pre> <p>View the status of the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,STATUS,HPELDIRECTORY='hpeldirectory'</pre> <p>View the configuration information for the data stream</p> <pre>MODIFY procname,APPL=DISPLAY,GATHERER,CONFIG,HPELDIRECTORY='hpeldirectory'</pre> <p>Usage note: <i>hpeldirectory</i> represents the High Performance Extensible Logging (HPEL) log directory. To prevent an error message, the directory must be enclosed in quotation marks.</p>

2. To run the NetView message provider, complete the following actions as appropriate.

Action	Instructions
Start the NetView message provider	<p>Specify either C (cold start) or W (warm start) as the value for the <i>COMMON.GLANETV.START</i> variable in the CNMSTYLE member, as shown in the following example:</p> <pre>COMMON.GLANETV.START = W</pre>
Stop the NetView message provider	Change the value of the <i>GLANETV.STOP</i> variable in the CGED panel to YES.

Running the System Data Engine

To start the IBM Common Data Provider for z Systems System Data Engine to have it stream SMF data to the Data Streamer, you use the System Data Engine started task.

Before you begin

Create the System Data Engine started task, as described in [“Creating the System Data Engine started task for streaming SMF data”](#) on page 80.

About this task

You use z/OS console commands to query the status of the System Data Engine and control its operation.

Troubleshooting tip: After the System Data Engine is started, it should not stop until you stop it. If it does stop without your stopping it explicitly, review the System Data Engine job log output for possible errors.

Procedure

To run the System Data Engine, issue the following console commands, where *procname* represents the name of the started task (such as HBOSMF).

Action	z/OS console command
Start the System Data Engine	START <i>procname</i>
Stop the System Data Engine	STOP <i>procname</i>
View System Data Engine status	MODIFY <i>procname</i> ,DISPLAY STATUS The status of the System Data Engine is written to the System Data Engine HBOOK file.

Chapter 6. Sending user application data to the Data Streamer

The IBM Common Data Provider for z Systems Open Streaming API provides an efficient way to gather operational data from your own applications by enabling your applications to be data gatherers. You can use the API to send your application data to the Data Streamer and stream it to analytics platforms.

Before you begin

Create your application stream definition, as described in [“Creating an application data stream definition” on page 54](#).

About this task

For sending your application data to the Data Streamer, IBM Common Data Provider for z Systems provides the Data Transfer Protocol, and Java and REXX APIs that implement the Data Transfer Protocol. When the Data Streamer receives a data packet, it processes and sends the data to subscribers, based on the policy that you define in the Configuration Tool.

The Data Streamer has an open TCP/IP port on which it accepts connections. It accepts connections only from data gatherers that are running on the same system, and using the same TCP/IP stack.

Tip: For more information about the Data Streamer port, see [“Data Streamer port definition” on page 7](#).

Data Transfer Protocol

The Data Transfer Protocol is used to transfer data among IBM Common Data Provider for z Systems components. It uses a binary, self-describing format that is delivered over TCP/IP. Data that is sent by using the Data Transfer Protocol can be split or unsplit.

Header information for data that is sent by using the Data Transfer Protocol

The transmission of data is preceded by the following information, in the following order:

1. A header that has a length of 96 bytes. Headers are listed and described in [Table 23 on page 185](#).
2. One of the following payload structures, which is described in the header:
 - [“Unsplit payload” on page 186](#)
 - [“Split payload” on page 187](#)

Table 23. Headers for data that is sent by using the Data Transfer Protocol			
Number of bytes	Type	Description	Value
4	Binary	Endianness	0x12345678

Table 23. Headers for data that is sent by using the Data Transfer Protocol (continued)

Number of bytes	Type	Description	Value
8	Text	Header encoding	The name of an encoding to use for all other text in the header, and for metadata in the payload. The encoding must be supported by Java. The name should be padded with blanks to 8 characters and encoded in UTF-8.
8	Text	Eye catcher	HBOCDP (encoded in the header encoding)
8	Text	Sender identifier	A unique identifier for the sender (encoded in header encoding)
4	Binary	Version	0x00000001
8	Reserved		
4	Binary	Payload type	<ul style="list-style-type: none"> • For unsplit data, 0x00000001 • For split data, 0x00000002
4	Binary	Payload length	The number of bytes in the payload. The maximum value is 2000000000.
48	Reserved		

Unsplit payload

To transmit unsplit data, use the payload format that is shown in [Table 24 on page 186](#).

Table 24. Unsplit payload format

Number of bytes	Type	Description
4	Binary	Number of metadata values
16 times the number of metadata values	Binary	Metadata keyword and value lengths and offsets. For each metadata value, the following information is included: <ul style="list-style-type: none"> • 4 bytes, which is the offset from the beginning of the payload to the metadata keyword • 4 bytes, which is the length of the metadata keyword • 4 bytes, which is the offset from the beginning of the payload to the metadata value • 4 bytes, which is the length of the metadata value
4	Binary	Offset from the beginning of the payload to the data
4	Binary	Length of data

Table 24. Unsplit payload format (continued)		
Number of bytes	Type	Description
Variable	Text	Metadata keywords and values. Tip: The lengths and offsets of these keywords and values are previously described in this payload. For more information about the metadata keywords and values, see Table 26 on page 188 .
Variable	Text	The data that is being transmitted

Split payload

To transmit split data, use the payload format that is shown in [Table 25 on page 187](#).

Table 25. Split payload format		
Number of bytes	Type	Description
4	Binary	Number of metadata values
16 times the number of metadata values	Binary	Metadata keyword and value lengths and offsets. For each metadata value, the following information is included: <ul style="list-style-type: none"> • 4 bytes, which are offset from the beginning of the payload to the metadata keyword • 4 bytes, which is the length of the metadata keyword • 4 bytes, which are offset from the beginning of the payload to the metadata value • 4 bytes, which is the length of the metadata value
4	Binary	Number of records in the data
8 times the number of records in the data	Binary	Record offsets and lengths. For each record, the following information is included: <ul style="list-style-type: none"> • 4 bytes, which are offset from the beginning of the payload to the record • 4 bytes, which is the length of the record
4	Binary	Offset from the beginning of the payload to the data
4	Binary	Length of data
Variable	Text	Metadata keywords and values. Tip: The lengths and offsets of these keywords and values are previously described in this payload. For more information about the metadata keywords and values, see Table 26 on page 188 .
Variable	Text	The data that is being transmitted. Tip: Based on the lengths and offsets, you can have data in this field that is not included in any record.

Metadata keywords and values

Table 26 on page 188 lists and describes the expected metadata keywords and values.

Table 26. Metadata keywords and values		
Keyword	Value	Indication of whether the keyword is required or optional
encoding	The character encoding of the data, which is typically one of the following values: <ul style="list-style-type: none">• IBM037• IBM1047• UTF-8	Required
sourcetype	The source type of the data. This value must be the same as the name of the data source type that is specified in the <code>parms</code> object in the <code>.streams.json</code> file.	Required
sourcename	The source name of the data. This value must be the same as the data source name that is specified in the <code>parms</code> object in the <code>.streams.json</code> file.	Required
timezone	If the time stamps in the data do not include a time zone, this value specifies a time zone to the target destination. Specify this value if the time zone is different from the system time zone. This value must be in the following format, where <i>plus_or_minus</i> represents the + or - sign, <i>HH</i> represents two digits for the hour, and <i>MM</i> represents two digits for the minute: <code>plus_or_minusHHMM</code>	Optional

Sending data by using the Java API

The IBM Common Data Provider for z Systems Java API is a set of Java classes that IBM Common Data Provider for z Systems uses to exchange data internally. You can use these classes to write Java applications that send data to the Data Streamer.

About this task

To send data, the API must have the port number on which the Data Streamer listens for data.

Tip: For more information about the Data Streamer port, see [“Data Streamer port definition” on page 7](#).

Procedure

To use the Java API to send data to the Data Streamer, complete the following steps:

1. Extract the `/DS/LIB/CDPzLibraries.tar` file, and add the `CdpCommon.jar` and `CdpProtocol.jar` files to the Java build path. Java API documentation is included in the TAR file.

2. As shown in the following example, define a Java class for the sender, where *port* is the port number on which the Data Streamer listens for data:

```
CDPSender sender = new CDPSender("localhost",port);
```

3. As shown in the following example, define a variable for identifying the origin of the data in traces and dumps.

The value of *senderName*, which must have a maximum length of 8 characters, is included in the header to identify the origin of the data.

```
String senderName = "SAMPSNDR";
```

4. As shown in the following example, define a Java class for containing the metadata for the data.

This table must contain the metadata keywords and values that are described in [“Metadata keywords and values”](#) on page 188.

```
HashMap<String, String> metadata = new HashMap<String, String>(4);
metadata.put(Dictionary.encoding.name(), "IBM1047");
metadata.put(Dictionary.sourcename.name(), "mySourceName");
metadata.put(Dictionary.sourcetype.name(), "mySourceType");
metadata.put(Dictionary.timezone.name(), "+0000");
```

5. To send the data to the Data Streamer, complete the following steps that apply, depending on whether you are sending split or unsplit data:

Option	Description
Split data	<p>a. Define a Java class for containing the records to be sent, and for adding records as they are collected, as shown in the following example:</p> <pre>List<String> records = new ArrayList<String>(); records.add(someRecord);</pre> <p>b. Send the data to the Data Streamer, as shown in the following example:</p> <pre>sender.sendType2(senderName, metadata, records);</pre>
Unsplit data	<p>a. Send the data to the Data Streamer, as shown in the following example:</p> <pre>String data = someData; sender.sendType1(senderName, metadata, data);</pre>

Important: The following Java exceptions are thrown by the `sendType2` and `sendType1` methods and must be caught:

IOException

Thrown if an I/O error occurs when connecting to or sending data to the Data Streamer.

IllegalArgumentException

Thrown when the metadata does not contain a value for encoding, or when the length of the sender name is greater than 8 characters.

UnsupportedEncodingException

Thrown when the encoding that is provided in the metadata is not supported by Java.

Sending data by using the REXX API

The IBM Common Data Provider for z Systems REXX API is a set of REstructured eXtended eXecutor (REXX) language functions that can be used to send data to the Data Streamer.

About this task

The sample REXX program HBORS001 in the *hlq.SHBOSAMP* library illustrates how to use the REXX API as described in the following procedure.

To send data, the API must have the port number on which the Data Streamer listens for data.

Tip: For more information about the Data Streamer port, see [“Data Streamer port definition” on page 7](#).

Procedure

To use the REXX API to send data to the Data Streamer, complete the following steps:

1. In your REXX program, include the REXX procedures from the HBORDAPI sample program, which is in the *hlq.SHBOSAMP* library.
2. As shown in the following example, define your metadata in a stem variable that is named *"Meta."*.
This table must contain the metadata keywords and values that are described in [“Metadata keywords and values” on page 188](#), with one value for each entry in *keyword=value* format.

```
Meta.0 = 4
Meta.1 = 'encoding=IBM1047'
Meta.2 = 'sourcetype=mySourceType'
Meta.3 = 'sourcename=mySourceName'
Meta.4 = 'timezone=+0000'
```

3. As shown in the following example, define your data in a stem variable that is named *"Data."*:

```
Data.0 = 3
Data.1 = 'Record 1'
Data.2 = 'Record 2'
Data.3 = 'Record 3'
```

4. To send data to the Data Streamer, complete the following steps that apply, depending on whether you are sending data in a single transmission or multiple transmissions:

Option	Description
Single transmission	<p>To connect to the Data Streamer, send the data, and disconnect from the Data Streamer, call HBO_Send_Data, as shown in the following example:</p> <pre>Call HBO_Send_Data port, type, sender</pre>
Multiple transmission	<p>If you have a long running program, you can open a connection to the Data Streamer before you call HBO_Send_Data so that the connection remains open, and you do not have to reconnect to send more data.</p> <p>a. Call HBO_Open, as shown in the following example:</p> <pre>Call HBO_Open port</pre> <p>b. Call HBO_Send_Data, as shown in the following example, which sends the data without connecting to, or disconnecting from, the Data Streamer:</p> <pre>Call HBO_Send_Data port, type, sender</pre>

Option	Description
	<p>Tip: In this call, the value for <i>port</i> is ignored because the connection to the Data Streamer is already open.</p> <p>c. When the program completes the sending of data, call HBO_Close to disconnect from the Data Streamer.</p> <p>Tip: If you make these calls from multiple, different REXX subroutines, ensure that any procedure statements expose the following variables:</p> <ul style="list-style-type: none"> • <i>HBO_Socket</i> • <i>hbo.</i> • <i>ecpref</i> • <i>ecname</i>

The following information describes the variables that are used in the calls:

port

The port number on which the local Data Streamer listens for data.

type

A value of 1 indicates unsplit data, and a value of 2 indicates split data.

sender

An eye catcher, with a maximum length of 8 characters, for identifying the origin of the data in traces and dumps.

Chapter 7. Loading data to IBM Db2 Analytics Accelerator for target destination IBM Tivoli Decision Support for z/OS

If your target destination is IBM Tivoli Decision Support for z/OS, you must load the z/OS operational data in batch mode from IBM Common Data Provider for z Systems to IBM Db2 Analytics Accelerator for z/OS for use by IBM Tivoli Decision Support for z/OS.

About this task

IBM Db2 Analytics Accelerator for z/OS is a high-performance component that is tightly integrated with Db2 for z/OS. It delivers high-speed processing for complex Db2 queries to support business-critical reporting and analytics workloads.

IBM Common Data Provider for z Systems can send System Management Facilities (SMF) data directly to IBM Db2 Analytics Accelerator for z/OS for storage, analytics, and reporting. The data is stored in IBM Db2 Analytics Accelerator for z/OS by using a database schema from IBM Tivoli Decision Support for z/OS analytics components.

The IBM Common Data Provider for z Systems System Data Engine converts SMF data into data sets that contain the IBM Tivoli Decision Support for z/OS analytics component tables in Db2 UNLOAD format. The IBM Db2 Analytics Accelerator Loader for z/OS is then used to load the data sets directly into IBM Db2 Analytics Accelerator for z/OS.

By sending data directly to IBM Db2 Analytics Accelerator for z/OS, you gain the following advantages:

- The need to store data in Db2 for z/OS is eliminated.
- More detailed timestamp level records can be stored.
- More CPU work is eliminated from the z/OS system.
- Reporting functions benefit from the high query speeds of IBM Db2 Analytics Accelerator for z/OS.

Configuring IBM Tivoli Decision Support for z/OS for loading the data

You must configure IBM Tivoli Decision Support for z/OS in preparation for loading the z/OS operational data in batch mode from IBM Common Data Provider for z Systems to IBM Db2 Analytics Accelerator for z/OS.

Before you begin

Apply the following updates for the following prerequisite software:

IBM Tivoli Decision Support for z/OS Version 1.8.2

APAR PI70968

IBM Db2 Analytics Accelerator for z/OS Version 5.1

One of the following two sets of PTFs are required (either PTF-2 level or PTF-3 level), as indicated:

- PTF-2 level with the following PTFs applied:
 - UI30285
 - UI30337
 - UI30740
 - UI31021
 - UI31148

- UI31287
- UI31302
- UI31507
- UI31571
- UI31739
- UI32368
- UI32588
- UI32707
- UI32810
- UI35006
- UI35007
- UI35008
- UI35009
- UI35010
- UI35011
- UI35012
- UI37271
- UI37783
- UI37784
- UI37785
- UI37786
- UI37793
- UI37794
- UI37795
- UI37796
- UI38702
- PTF-3 level with the following PTFs applied:
 - UI33493
 - UI33603
 - UI33797
 - UI35501
 - UI36461
 - UI37053
 - UI37534
 - UI39653
 - UI39921
 - UI40892
 - UI41378
 - UI42327
 - UI42328
 - UI42329

IBM Db2 Analytics Accelerator Loader for z/OS Version 2.1

The following PTFs are required:

- UI18415
- UI20963
- UI21883
- UI22759
- UI23712
- UI26834
- UI27815
- UI33956
- UI35108
- UI36231
- UI36343
- UI38008
- UI38201
- UI38202
- UI38810
- UI38811
- UI38939
- UI38943
- UI38973
- UI39437
- UI39451
- UI39454

IBM Common Data Provider for z Systems Version 1.1.0

The following PTFs are required:

- UA91431
- UA91450
- UA91451
- UA91452

About this task

IBM Tivoli Decision Support for z/OS includes an analytics component for each set of tables that are supported in IBM Db2 Analytics Accelerator for z/OS. [“IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System Data Engine” on page 201](#) lists these analytics components with their subcomponents and the names of the corresponding base components in IBM Tivoli Decision Support for z/OS.

Procedure

To configure IBM Tivoli Decision Support for z/OS for loading the data, complete the following steps:

1. Bind the Db2 plan that is used by IBM Tivoli Decision Support for z/OS by specifying one of the following BIND options:
 - QUERYACCELERATION(ELIGIBLE)
 - QUERYACCELERATION(ENABLE)

For example, if you use the default plan name DRLPLAN, the following BIND PACKAGE is used to set the query acceleration register as eligible:

```
//SYSTSIN DD *
DSN SYSTEM(DSN)
  BIND PACKAGE(DRLPLAN) OWNER(authid) MEMBER(DRLPSQLX) -
    ACTION(REPLACE) ISOLATION(CS) ENCODING(EBCDIC) -
    QUERYACCELERATION(ELIGIBLE)
  BIND PLAN(DRLPLAN) OWNER(authid) PKLIST(*.DRLPLAN.*) -
    ACTION(REPLACE) RETAIN

  RUN PROGRAM(DSNTIAD) PLAN(DSNTIAxx) -
    LIB('xxx.RUNLIB.LOAD')
END
```

The SDRLCNTL (DRLJDBIN) job includes sample instructions for binding the plan with QUERYACCELERATION specified.

2. Modify DRLFPROF, which is the IBM Tivoli Decision Support for z/OS data set that contains user-modified parameters, to reflect the settings to apply when installing new analytics components. The following parameters in DRLFPROF provide support for the IBM Db2 Analytics Accelerator for z/OS:

def_useaot = "YES" | "NO"

"YES"

Means that the table is created as an Accelerator Only table.

"NO"

Means that the table is created in Db2 and can be used either as a Db2 table or as an IDAA_ONLY table. The default value is "NO".

def_accelerator = "xxxxxxx"

"xxxxxxx"

The name of the accelerator where the table resides.

def_timeint = "H" | "S" | "T"

"H"

The time stamp for tables is rounded to an hourly interval (similar to tables with a suffix of _H in other components).

"S"

The time stamp for tables is rounded to a seconds interval (similar to tables with a time field rather than a time stamp in other components).

"T"

The time stamp for tables is the actual time stamp in the SMF record (similar to tables with suffix _T). The default value is "T".

3. If you are using IBM Tivoli Decision Support for z/OS to collect and populate the component tables in Db2 for z/OS, or if you are using IBM Tivoli Decision Support for z/OS reporting, customize each new lookup table in the IBM Tivoli Decision Support for z/OS analytics components to reflect the contents of any existing lookup tables in IBM Tivoli Decision Support for z/OS. For example, insert the same rows that are currently in the DB2_APPLICATION table into the A_DB2_APPLICATION table.

Table 27 on page 197 lists the lookup table members to customize.

Tip: If you are collecting data only into IBM Db2 Analytics Accelerator for z/OS rather than having the data reside in Db2 for z/OS, the lookup tables are configured in IBM Common Data Provider for z Systems, as described in [“Running the System Data Engine to write data in Db2 UNLOAD format” on page 198](#).

<i>Table 27. IBM Tivoli Decision Support for z/OS lookup table members to customize</i>		
Member	Base component table	Analytics component table
DRLTA2AP	DB2_APPLICATION	A_DB2_APPLICATION
DRLTA2AC	DB2_ACCUMAC	A_DB2_ACCUMAC
DRLTALUG	USER_GROUP	A_USER_GROUP
DRLTALKP	KPM_THRESHOLDS	A_KPM_THRESHOLDS_L
DRLTALW2	MVS_WORKLOAD2_TYPE	A_WORKLOAD2_L
DRLTALDA	MVSPM_DEVICE_ADDR	A_DEVICE_ADDR_L
DRLTALUT	MVSPM_UNIT_TYPE	A_UNIT_TYPE_L
DRLTALMI	MVS_MIPS_T	A_MIPS_L
DRLTALSP	MVS_SYSPLEX	A_SYSPLEX_L
DRLTALWL	MVS_WORKLOAD_TYPE	A_WORKLOAD_L
DRLTALW2	MVS_WORKLOAD2_TYPE	A_WORKLOAD2_L
DRLTALTR	MVSPM_TIME_RES	A_TIME_RES_L

4. Install the IBM Tivoli Decision Support for z/OS analytics components that you want to use into IBM Tivoli Decision Support for z/OS.

For information about how to install components into IBM Tivoli Decision Support for z/OS, see the IBM Tivoli Decision Support for z/OS administration documentation in the [IBM Knowledge Center](#).

5. After the IBM Tivoli Decision Support for z/OS analytics components and their associated tables are created in IBM Tivoli Decision Support for z/OS, add them to IBM Db2 Analytics Accelerator for z/OS by using the Data Studio Eclipse application or by using stored procedures.

[Table 28 on page 197](#) lists sample jobs for adding tables to IBM Db2 Analytics Accelerator for z/OS.

<i>Table 28. Sample jobs for adding tables to IBM Db2 Analytics Accelerator for z/OS</i>	
Analytics component	SDRLCNTL member
Analytics - z/OS Performance	DRLJAPMA
Analytics – Db2	DRLJA2DA
Analytics - KPM CICS	DRLJAKCA
Analytics - KPM Db2	DRLJAKDA
Analytics - KPM z/OS	DRLJAKZA

6. To move the contents of the lookup tables into the IBM Db2 Analytics Accelerator for z/OS, modify and submit the SDRLCNTL members that are listed in [Table 29 on page 197](#).

<i>Table 29. Sample jobs for moving lookup table contents to IBM Db2 Analytics Accelerator for z/OS</i>	
Analytics component	SDRLCNTL member
Analytics - z/OS Performance	DRLJAPMK
Analytics - KPM Db2	DRLJAKDK
Analytics - KPM z/OS	DRLJAKZK

Running the System Data Engine to write data in Db2 UNLOAD format

The IBM Common Data Provider for z Systems System Data Engine converts System Management Facilities (SMF) data into data sets that contain the IBM Tivoli Decision Support for z/OS analytics component tables in Db2 UNLOAD format. The IBM Db2 Analytics Accelerator Loader for z/OS is then used to load the data sets directly into IBM Db2 Analytics Accelerator for z/OS.

Procedure

To run the System Data Engine to write data in Db2 UNLOAD format, complete the following steps:

1. Copy and customize the IBM Common Data Provider for z Systems lookup definition members in [Table 30 on page 198](#) to reflect the contents of the corresponding IBM Tivoli Decision Support for z/OS lookup tables.

For example, insert the same rows that are currently in the DB2_APPLICATION table into the A_DB2_APPLICATION table.

These lookup tables are used by the System Data Engine when generating the Db2 UNLOAD format for each table. The System Data Engine lookup tables that are defined in these members have the same names as the IBM Tivoli Decision Support for z/OS analytics component lookup tables.

Table 30. IBM Common Data Provider for z Systems lookup table members

HBOvrm.SHBODEFS member	Analytics component lookup table	Base component lookup table
HBOTA2AP	A_DB2_APPLICATION	DB2_APPLICATION
HBOTA2AC	A_DB2_ACCUMAC	DB2_ACCUMAC
HBOTALUG	A_USER_GROUP	USER_GROUP
HBOTALKP	A_KPM_THRESHOLDS_L	KPM_THRESHOLDS
HBOTALWL	A_WORKLOAD2_L	MVS_WORKLOAD2_TYPE
HBOTALMI	A_MIPS_L	MVS_MIPS_T
HBOTALSP	A_SYSPLEX_L	MVS_SYSPLEX
HBOTALWL	A_WORKLOAD_L	MVS_WORKLOAD_TYPE
HBOTALW2	A_WORKLOAD2_L	MVS_WORKLOAD2_TYPE
HBOTALDA	A_DEVICE_ADDR_L	MVSPM_DEVICE_ADDR
HBOTALUT	A_UNIT_TYPE_L	MVSPM_UNIT_TYPE
HBOTALTR	A_TIME_RES_L	MVSPM_TIME_RES

2. Run the System Data Engine to generate Db2 UNLOAD format for the tables that are created for the IBM Db2 Analytics Accelerator by the IBM Tivoli Decision Support for z/OS analytics components.

The HBOvrm.SHBOCNTL members that are listed in [Table 31 on page 198](#) include sample JCL jobs for generating Db2 UNLOAD format for each of the analytics component tables.

Table 31. Sample jobs for generating Db2 UNLOAD format

HBOvrm.SHBOCNTL member	Analytics component
HBOAPMUN	Analytics - z/OS Performance
HBOA2DUN	Analytics – Db2

Table 31. Sample jobs for generating Db2 UNLOAD format (continued)	
HBOvrm.SHBOCNTL member	Analytics component
HBOAKCUN	Analytics - KPM CICS
HBOAKDUN	Analytics - KPM Db2
HBOAKZUN	Analytics - KPM Z/OS

Each sample includes two steps. The first step deletes output files that are created by a prior run, and the second step allocates output files and generates Db2 UNLOAD format from a data set that contains SMF records. The second COLLECT step uses the following DD names:

- HBOIN provides control statement input to the System Data Engine. It references the following members of HBOvrm.SHB0DEFS:
 - HBOLLSMF contains control statements defining the SMF log as input.
 - HBORS* members contain control statements for extracting data from SMF records.
 - HBOT* members contain control statements to define the lookup tables that are used by the System Data Engine.
 - HBOUA* members contain control statements to store the SMF data in Db2 UNLOAD format.
 - The in-stream COLLECT statement initiates System Data Engine processing.
- HBOLOG provides the input to the System Data Engine, which must be a data set that contains SMF records.
- Various UA* DD names refer to the output files to be written in Db2 UNLOAD format. The convention is that the DD name of the file matches the name of the HBOvrm.SHB0DEFS member, without the HBO prefix. Each file that is produced by a definition member is in a sequence (such as 1, 2, 3, or 4). For example, DD name UA2D11 refers to table A_DB2_SYS_PARM_I in Db2 UNLOAD format, which is the first file that is output by definition member HBOUA2D1.

Loading data to IBM Db2 Analytics Accelerator

The IBM Db2 Analytics Accelerator Loader for z/OS is used to load the data that is output from the IBM Common Data Provider for z Systems System Data Engine directly into IBM Db2 Analytics Accelerator for z/OS.

Procedure

Run the IBM Db2 Analytics Accelerator Loader for z/OS by using the Db2 LOAD utility with the following updates:

- A DD statement that enables the loader to intercept the Db2 LOAD utility:

```
//HLODUMMY DD DUMMY
```

- A statement that directs the loader to load data into the IBM Db2 Analytics Accelerator. This statement indicates the name of the accelerator and indicates that the target is an IDAA_ONLY table, as shown in the following example:

```
//SYSIN DD *
LOAD DATA RESUME YES LOG NO INDDN input_data_set_ddname
      IDAA_ONLY ON accelerator-name
      INTO TABLE DRLxx.KPMZ_WORKLOAD_T FORMAT INTERNAL;
```

The DRLvrm.SDRLCNTL members that are listed in [Table 32 on page 200](#) include sample JCL jobs for loading Db2 UNLOAD format data for each of the analytics component tables to IBM Db2 Analytics Accelerator.

Table 32. Sample jobs for loading data into IBM Db2 Analytics Accelerator

DRLVIM.SDRLCNTL member	Analytics component
DRLJAPMD	Analytics - z/OS Performance
DRLJA2DD	Analytics – Db2
DRLJAKCD	Analytics - KPM CICS
DRLJAKDD	Analytics - KPM Db2
DRLJAKZD	Analytics - KPM Z/OS

After the load is complete from the first time that you load an IDAA_ONLY table, the table must be enabled for acceleration in IBM Db2 Analytics Accelerator for z/OS. Tables can be enabled for acceleration by using the Data Studio Eclipse application, or by using stored procedures.

The DRLVIM.SDRLCNTL members that are listed in [Table 33 on page 200](#) include sample JCL jobs for using stored procedures to enable tables for acceleration.

Table 33. Sample jobs for enabling tables for acceleration in IBM Db2 Analytics Accelerator

DRLVIM.SDRLCNTL member	Analytics component
DRLJAPME	Analytics - z/OS Performance
DRLJA2DE	Analytics – Db2
DRLJAKCE	Analytics - KPM CICS
DRLJAKDE	Analytics - KPM Db2
DRLJAKZE	Analytics - KPM Z/OS

Removing tables from IBM Db2 Analytics Accelerator

If you want to uninstall a component in IBM Tivoli Decision Support for z/OS that has tables that were added to IBM Db2 Analytics Accelerator for z/OS (by using job DRLJAKXA), you must first remove the tables from IBM Db2 Analytics Accelerator for z/OS.

Procedure

To remove tables from IBM Db2 Analytics Accelerator for z/OS, customize and submit one or more of the jobs in [Table 34 on page 200](#).

Table 34. Sample jobs that are provided by IBM Tivoli Decision Support for z/OS for removing tables from IBM Db2 Analytics Accelerator for z/OS

DRLVIM.SDRLCNTL member	Analytics component
DRLJAPMR	Analytics - z/OS Performance
DRLJA2DR	Analytics – Db2
DRLJAKCR	Analytics - KPM CICS
DRLJAKDR	Analytics - KPM Db2
DRLJAKZR	Analytics - KPM Z/OS

IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System Data Engine

This reference lists the analytics components of IBM Tivoli Decision Support for z/OS that can be loaded by the IBM Common Data Provider for z Systems System Data Engine and used for storing data directly in the IBM Db2 Analytics Accelerator for z/OS.

Table 35 on page 201 lists the analytics components with their subcomponents and the names of the corresponding base components in IBM Tivoli Decision Support for z/OS.

Table 35. IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System Data Engine		
Analytics component	Subcomponents	Corresponding base component in IBM Tivoli Decision Support for z/OS
Analytics - z/OS Performance	<ul style="list-style-type: none">• Coupling Facility (CF)• Cross System Coupling Facility (XCF)• Open MVS (OMVS)• System• Workload• I/O• Global Storage• Virtual Storage• Device• Cryptography• Application	MVSPM
Analytics - Db2	<ul style="list-style-type: none">• Initialization• Address Space• Buffer Pool• Accnt and RespTime• Package• Data Sharing• DDF• Storage	Db2
Analytics - KPM CICS	<ul style="list-style-type: none">• Monitoring	CICS Key Performance Metrics
Analytics - KPM Db2	<ul style="list-style-type: none">• Db2 Accounting Level• Db2 Package• Db2 System Level	Db2 Key Performance Metrics

Table 35. IBM Tivoli Decision Support for z/OS analytics components that can be loaded by the System Data Engine (continued)

Analytics component	Subcomponents	Corresponding base component in IBM Tivoli Decision Support for z/OS
Analytics - KPM z/OS	<ul style="list-style-type: none"> • Address Space • LPAR • Storage • Workload • Capture Ratio Workload/LPAR • Channel • Coupling Facility • Hardware Capacity • Problem Determination 	z/OS Key Performance Metrics

Analytics component tables

For each IBM Tivoli Decision Support for z/OS analytics component that can be loaded by the IBM Common Data Provider for z Systems System Data Engine, this reference lists the associated tables, with the corresponding IBM Tivoli Decision Support for z/OS base component tables.

The tables are listed for the following analytics components:

- [“Analytics - z/OS Performance component” on page 202](#)
- [“Analytics - Db2 component” on page 204](#)
- [“Analytics - KPM CICS component” on page 205](#)
- [“Analytics - KPM Db2 component” on page 205](#)
- [“Analytics - KPM z/OS component” on page 206](#)

Analytics - z/OS Performance component

*Table 36. Tables for **Analytics - z/OS Performance** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables*

Table	Corresponding base component table
A_PM_CF_I	MVSPM_CF_H
A_PM_CF_LINK_I	MVSPM_CF_LINK_H
A_PM_CF_PROC_I	MVSPM_CF_PROC_H
A_PM_CF_REQ_I	MVSPM_CF_REQUEST_H
A_PM_CF_CF_I	MVSPM_CF_TO_CF_H
A_PM_XCF_MEMBER_I	MVSPM_XCF_MEMBER_H
A_PM_XCF_PATH_I	MVSPM_XCF_PATH_H
A_PM_XCF_SYS_I	MVSPM_XCF_SYS_H
A_PM_OMVS_BUF_I	MVSPM_OMVS_BUF_H
A_PM_OMVS_FILE_I	MVSPM_OMVS_FILE_H

Table 36. Tables for **Analytics - z/OS Performance** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables (continued)

Table	Corresponding base component table
A_PM_OMVS_GHFS_I	MVSPM_OMVS_GHFS_H
A_PM_OMVS_HFS_I	MVSPM_OMVS_HFS_H
A_PM_OMVS_KERN_I	MVSPM_OMVS_KERN_H
A_PM_OMVS_MOUNT_I	MVSPM_OMVS_MOUNT_H
A_PM_SYS_CLUST_I	MVSPM_CLUSTER_H
A_PM_SYS_CPU_I	MVSPM_CPU_H
A_PM_SYS_CPUMT_I	MVSPM_CPUMT_H
A_PM_SYS_ENQ_I	MVSPM_ENQUEUE_H
A_PM_SYS_LPAR_I	MVSPM_LPAR_H
A_PM_SYS_SYS_I	MVSPM_SYSTEM_H
A_PM_SYS_PROD_I	MVSPM_PROD_T
A_PM_SYS_PRDINT_I	MVSPM_PROD_INT_T
A_PM_SYS_MSU_I	MVSPM_LPAR_MSU_T
A_PM_WL_GOAL_I	MVSPM_GOAL_ACT_H
A_PM_WL_SERVED_I	MVSPM_WLM_SERVED_H
A_PM_WL_STATE_I	MVSPM_WLM_STATE_H
A_PM_WL_WKLD_I	MVSPM_WORKLOAD_H
A_PM_WL_WKLD2_I	MVSPM_WORKLOAD2_H
A_PM_IO_DATASET_I	MVSPM_DATASET_H
A_PM_IO_VOLUME_I	MVSPM_VOLUME_H
A_PM_IO_LCU_I	MVSPM_LCU_IO_H
A_PM_GS_BMF_I	MVSPM_BMF_H
A_PM_GS_CACHE_I	MVSPM_CACHE_H
A_PM_GS_PAGDS_I	MVSPM_PAGE_DS_H
A_PM_GS_PAGING_I	MVSPM_PAGING_H
A_PM_GS_STORAGE_I	MVSPM_STORAGE_H
A_PM_GS_STORCLS_I	MVSPM_STORCLASS_H
A_PM_GS_SWAP_I	MVSPM_SWAP_H
A_PM_GS_CACHESS_I	MVSPM_CACHE_ESS_H
A_PM_VS_VLF_I	MVSPM_VLF_H
A_PM_VS_CSASQA_I	MVSPM_VS_CSASQA_H
A_PM_VS_PRIVATE_I	MVSPM_VS_PRIVATE_H
A_PM_VS_SUBPOOL_I	MVSPM_VS_SUBPOOL_H
A_PM_DEV_CHAN_I	MVSPM_CHANNEL_H

Table 36. Tables for **Analytics - z/OS Performance** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables (continued)

Table	Corresponding base component table
A_PM_DEV_HSCHAN_I	MVSPM_HS_CHAN_H
A_PM_DEV_AP_I	MVSPM_DEVICE_AP_H
A_PM_DEV_DEVICE_I	MVSPM_DEVICE_H
A_PM_DEV_FICON_I	MVSPM_FICON_H
A_PM_DEV_RAID_I	MVSPM_RAID_RANK_H
A_PM_DEV_ESSLNK_I	MVSPM_ESSLINK_H
A_PM_DEV_ESSEXT_I	MVSPM_ESS_EXTENT_H
A_PM_DEV_ESSRNK_I	MVSPM_ESS_RANK_H
A_PM_DEV_PCIE_I	MVSPM_PCIE_H
A_PM_Cryp_PCI_I	MVSPM_CRYPTOPCI_H
A_PM_Cryp_CCF_I	MVSPM_CRYPTOPCF_H
A_PM_APP_APPL_I	MVSPM_APPL_H

Analytics - Db2 component

Table 37. Tables for **Analytics - Db2** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables

Table	Corresponding base component table
A_DB2_SYS_PARM_I	DB2_SYS_PARAMETER
A_DB2_DB_I	DB2_DATABASE_T
A_DB2_DB_BIND_I	DB2_DATABASE_T
A_DB2_DB_QIST_I	DB2_DATABASE_T
A_DB2_DB_SYS_I	DB2_SYSTEM_T
A_DB2_BP_I	DB2_BUFFER_POOL_T
A_DB2_USERTRAN_I	DB2_USER_TRAN_H
A_DB2_UT_BP_I	DB2_USER_TRAN_H
A_DB2_UT_SACC_I	DB2_USER_TRAN_H
A_DB2_UT_IDAA_I	DB2_USER_TRAN_H
A_DB2_IDAA_STAT_I	DB2_IDAA_STAT_H
A_DB2_IDAA_ACC_I	DB2_IDAA_ACC_H
A_DB2_IDAA_ST_A_I	DB2_IDAA_STAT_A_H
A_DB2_IDAA_ST_S_I	DB2_IDAA_STAT_S_H
A_DB2_PACK_I	DB2_PACKAGE_H

Table 37. Tables for **Analytics - Db2** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables (continued)

Table	Corresponding base component table
A_DB2_SHR_BP_I	DB2_BP_SHARING_T
A_DB2_SHR_BPAT_I	DB2_BPATTR_SHR_T
A_DB2_SHR_LOCK_I	DB2_LOCK_SHARING_T
A_DB2_SHR_INIT_I	DB2_SHARING_INIT
A_DB2_SHR_TRAN_I	DB2_US_TRAN_SHAR_H
A_DB2_DDF_I	DB2_USER_DIST_H
A_DB2_SYSTEM_I	DB2_SYSTEM_DIST_T
A_DB2_STORAGE_I	DB2_STORAGE_T
A_DB2_TRAN_IV	DB2_TRANSACTION_D
A_DB2_DATABASE_IV	DB2_DATABASE_T

Analytics - KPM CICS component

Table 38. Tables for **Analytics - KPM CICS** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables

Table	Corresponding base component table
A_KC_MON_TRAN_I	KPMC_MON_TRAN_H

Analytics - KPM Db2 component

Table 39. Tables for **Analytics - KPM Db2** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables

Table	Corresponding base component table
A_KD_UT_I	KPM_DB2_USERTRAN_H
A_KD_UT_BP_I	KPM_DB2_USERTRAN_H
A_KD_EU_I	KPM_DB2_ENDUSER_H
A_KD_EU_BP_I	KPM_DB2_ENDUSER_H
A_KD_PACKAGE_I	KPM_DB2_PACKAGE_H
A_KD_SYS_IO_I	KPM_DB2_SYSTEM_T
A_KD_SYS_TCBSRB_I	KPM_DB2_SYSTEM_T
A_KD_SYS_LATCH_I	KPM_DB2_LATCH_T
A_KD_SYS_BP_I	KPM_DB2_BP_T
A_KD_SYS_BP_SHR_I	KPM_DB2_BP_SHR_T

Table 39. Tables for **Analytics - KPM Db2** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables (continued)

Table	Corresponding base component table
A_KD_SYS_ST_DBM_I	KPM_DB2_STORAGE_T
A_KD_SYS_ST_DST_I	KPM_DB2_STORAGE_T
A_KD_SYS_ST_COM_I	KPM_DB2_STORAGE_T
A_DB_SYS_DB_WF_I	KPM_DB2_DATABASE_T
A_DB_SYS_DB_EDM_I	KPM_DB2_DATABASE_T
A_DB_SYS_DB_SET_I	KPM_DB2_DATABASE_T
A_DB_SYS_DB_LOCK_I	KPM_DB2_LOCK_T

Analytics - KPM z/OS component

Table 40. Tables for **Analytics - KPM z/OS** component of IBM Tivoli Decision Support for z/OS, with corresponding base component tables

Table	Corresponding base component table
A_KPM_EXCEPTION_I	KPM_EXCEPTION_T
A_KZ_JOB_INT_I	KPMZ_JOB_INT_T
A_KZ_JOB_STEP_I	KPMZ_JOB_STEP_T
A_KZ_LPAR_I	KPMZ_LPAR_T
A_KZ_STORAGE_I	KPMZ_STORAGE_T
A_KZ_WORKLOAD_I	KPMZ_WORKLOAD_T
A_KZ_CHANNEL_I	KPMZ_CHANNEL_T
A_KZ_CF_I	KPMZ_CF_T
A_KZ_CF_STRUC_I	KPMZ_CF_STRUCTR_T
A_KZ_CPUMF_I	KPMZ_CPUMF_T
A_KZ_CPUMF1_I	KPMZ_CPUMF1_T
A_KZ_CPUMF_PT_I	KPMZ_CPUMF_PT_T
A_KZ_CPUMF1_PT_I	KPMZ_CPUMF1_PT_T
A_KZ_SRM_WKLD_I	KPMZ_SRM_WKLD_T

Analytics component views that are based on multiple tables

In some cases, multiple tables from an IBM Tivoli Decision Support for z/OS analytics component are combined into a single view. In these cases, the resulting view matches a table from an IBM Tivoli Decision Support for z/OS base component. This reference lists these analytics component views that are based on multiple tables.

Table 41. IBM Tivoli Decision Support for z/OS analytics component views that are based on multiple tables

Analytics component	View name	Analytics component tables that are used in view	Base component table on which view is based
Analytics - Db2	A_DB2_USERTRAN_IV	<ul style="list-style-type: none"> • A_DB2_USERTRAN_I • A_DB2_UT_BP_I • A_DB2_UT_SACC_I • A_DB2_UT_IDAA_ 	DB2_USER_TRAN_H
Analytics - Db2	A_DB2_DATABASE_IV	<ul style="list-style-type: none"> • A_DB2_DB_I • A_DB2_DB_BIND_I • A_DB2_DB_QIST_I 	DB2_DATABASE_T
Analytics - KPM Db2	A_KD_USERTRAN_IV	<ul style="list-style-type: none"> • A_KD_UT_I • A_KD_UT_BP_I 	KPM_DB2_USERTRAN_H
Analytics - KPM Db2	A_KD_ENDUSER_IV	<ul style="list-style-type: none"> • A_KD_EU_I • A_KD_EU_BP_I 	KPM_DB2_ENDUSER_H
Analytics - KPM Db2	A_KD_SYSTEM_IV	<ul style="list-style-type: none"> • A_KD_SYS_IO_I • A_KD_SYS_TCBSRB_I 	KPM_DB2_SYSTEM_T
Analytics - KPM Db2	A_KD_STORAGE_IV	<ul style="list-style-type: none"> • A_KD_SYS_ST_DBM_I • A_KD_SYS_ST_DST_I • A_KD_SYS_ST_COM_I 	KPM_DB2_STORAGE_T
Analytics - KPM Db2	A_KD_DATABASE_IV	<ul style="list-style-type: none"> • A_DB_SYS_DB_WF_I • A_DB_SYS_DB_EDM_I • A_DB_SYS_DB_SET_I 	KPM_DB2_DATABASE_T

Chapter 8. Troubleshooting Common Data Provider for z Systems

This reference lists known problems that you might experience in using the IBM Common Data Provider for z Systems and describes known solutions. It also includes information about Log Forwarder logging and tracing.

Log Forwarder log files

Troubleshooting information is available in log files that are generated by the Log Forwarder.

Log Forwarder log files

Log Forwarder logging information (and tracing information, if tracing is enabled) is sent to the STDERR data set on the GLAPROC job.

Significant Log Forwarder messages, such as the following messages, are also written to the console:

- Startup and shutdown messages are written as information messages.
- Certain errors are written as action messages. These messages are cleared when the error condition is resolved or when the Log Forwarder is stopped.

The level of message information that is provided on the console and in the STDERR data set is the same. However, if stack trace data is available, it is included in only the STDERR data set.

Log Forwarder: enabling tracing

For certain problems, IBM Software Support might request that you enable tracing for the Log Forwarder.

About this task

For the Log Forwarder, you can enable tracing in either of the following ways:

- Enable static tracing by using the logging configuration file
- Enable dynamic tracing by using the MVS **MODIFY** command

Enabling static tracing for the Log Forwarder

For the Log Forwarder, you can enable static tracing by using the logging configuration file.

About this task

The trace settings in the logging configuration file are applied each time that the Log Forwarder is started.

If you do not want to restart the Log Forwarder to enable tracing, use the MVS **MODIFY** command to enable dynamic tracing.

Procedure

To enable static tracing, complete the following steps:

1. Copy the `logging.properties` file from the `samples` directory to a read/write directory.
2. Edit the `logging.properties` file as instructed by IBM Software Support.
3. Update the environment configuration file to set the value of the `ZLF_LOG` environment variable to the directory where you copied the `logging.properties` file in step “1” on [page 209](#).
4. Restart the Log Forwarder.

What to do next

When the trace settings are no longer needed, return the logging configuration file to its original contents.

Enabling dynamic tracing for the Log Forwarder

For the Log Forwarder, you can enable dynamic tracing by using the MVS **MODIFY** command.

About this task

Trace settings that are changed by using the MVS **MODIFY** command are not persisted and therefore have no effect when the Log Forwarder is restarted.

To configure trace settings that persist each time that the Log Forwarder is started, use the logging configuration file to enable static tracing.

Procedure

To enable dynamic tracing, IBM Software Support might require you to issue one or more of the following commands:

Option	Description
Set trace	<p>To set the trace level for a specific component of the Log Forwarder, issue the following system command:</p> <pre>F GLAPROC,APPL=SET,TRACE,<i>logger</i>,<i>level</i></pre> <p>The values for <i>logger</i> and <i>level</i> are provided by IBM Software Support. Typically, <i>level</i> is one of the following three values:</p> <p>EVENT The default tracing level. This level provides limited tracing that shows detailed error and warning responses from other applications.</p> <p>DEBUG This level provides moderate tracing that shows values of significant variables at key points in the code path.</p> <p>TRACE This level provides extensive tracing that shows detailed paths through the code, including method entry and exit.</p>
Display trace	<p>a. To display the trace levels for all components of the Log Forwarder, issue the following system command:</p> <pre>F GLAPROC,APPL=DISPLAY,TRACE</pre> <p>The loggers with level values that are explicitly set are the only ones that are displayed. Typically, only the root logger has a level value set, and that value is typically EVENT (the default level). By default, all other loggers inherit their level from the root logger.</p> <p>b. To display the trace level for a specific component of the Log Forwarder, issue the following system command:</p> <pre>F GLAPROC,APPL=DISPLAY,TRACE,<i>logger</i></pre> <p>The value for <i>logger</i> is provided by IBM Software Support.</p>
Clear trace	<p>a. To clear the trace levels for all components of the Log Forwarder, issue the following system command:</p> <pre>F GLAPROC,APPL=CLEAR,TRACE</pre>

Option	Description
	<p>This command sets the value of the root logger level to EVENT. It also clears the values for all other loggers so that they inherit their level from the root logger. After the enabled trace settings are no longer needed, use this command to return to the default tracing state.</p> <p>b. To clear the trace level for a specific component of the Log Forwarder, issue the following system command:</p> <pre>F GLAPROC,APPL=CLEAR,TRACE,logger</pre> <p>The value for <i>logger</i> is provided by IBM Software Support. After a logger level is cleared, the logger inherits its level from another component.</p>

System Data Engine log files

Troubleshooting information is available in log files that are generated by the System Data Engine.

System Data Engine log files

System Data Engine logging information is in the following data sets:

- The HB00UT data set contains the general messages that are generated by the System Data Engine. There might be error messages that refer to the HB0DUMP data set for more information.
- The HB0DUMP data set contains diagnostic messages for the errors that the System Data Engine encounters. It also contains the tracing information if the HB0DEBUG data set is not in the JCL for the System Data Engine started task.
- The HB0DEBUG data set contains the tracing information if this data set exists in the JCL for the System Data Engine started task before you turn on the trace function.

System Data Engine: enabling tracing

For certain problems, IBM Software Support might request that you enable tracing for the System Data Engine.

About this task

For the System Data Engine, you can enable tracing in either of the following ways:

- Enable tracing at startup by modifying the HB0IN DD.
- Enable tracing after startup by using the MVS **MODIFY** command.

Enabling tracing for the System Data Engine at startup

For certain problems, IBM Software Support might request that you enable tracing for the System Data Engine at startup by modifying the HB0IN DD.

Procedure

1. To enable tracing, IBM Software Support might require you to add the trace command `DEBUG LEVEL` to the beginning of the HB0IN DD.

```
//HB0IN DD *
        DEBUG LEVEL
        ...
```

LEVEL

Is the level of tracing to be enabled. The value of level will be provided by IBM Software Support when you must enable tracing. Tracing should not be enabled otherwise.

2. Start the System Data Engine for the trace command to take effect.
3. Verify that the appropriate messages are written in the HBODEBUG data set.

Results

The tracing output is in the HBODEBUG data set if it is in the JCL for the System Data Engine started task, or in the HBODUMP data set if the HBODEBUG data set is not in the JCL for the System Data Engine started task. See the following example for specifying the HBODEBUG DD.

```
//HBODEBUG DD SYSOUT=*
```

What to do next

To disable tracing, use one of the following methods:

- Remove the trace command `DEBUG LEVEL` and recycle the started task.
- Run the following command:

```
F STCNAME, DEBUG CLEAR
```

Ensure that you remove the trace command when the tracing function is not needed.

Enabling tracing for the System Data Engine after startup

For certain problems, IBM Software Support might request that you enable tracing for the System Data Engine after startup by using the MVS **MODIFY** command.

Procedure

1. To enable tracing, IBM Software Support might require you to run a command that is similar to one of the following lines.

```
F JOBNAME, DEBUG LEVEL  
F JOBNAME, DEBUG,LEVEL
```

JOBNAME

Is the name of the job for which you want to enable tracing, for example DEV\$SDE.

LEVEL

Is the level of tracing to be enabled.

The exact syntax of the command will be provided by IBM Software Support when you must enable tracing. Tracing should not be enabled otherwise.

2. Verify that the appropriate messages are written in the HBODEBUG data set after you run each command.

Results

The tracing output is in the HBODEBUG data set if it is in the JCL for the System Data Engine started task, or in the HBODUMP data set if the HBODEBUG data set is not in the JCL for the System Data Engine started task. See the following example for specifying the HBODEBUG DD.

```
//HBODEBUG DD SYSOUT=*
```

What to do next

To disable tracing, run the following command:

```
F STCNAME, DEBUG CLEAR
```

Log Forwarder user ID has insufficient authority

The Log Forwarder does not operate correctly due to insufficient authority.

Symptom

The following symptoms are possible:

- Messages in the procedure STDERR data set indicate that the `startup.sh` script cannot be found.
- System Authorization Facility (SAF) messages indicate that the user has insufficient authority to complete an operation. For example, the ICH408I message is issued by the Resource Access Control Facility (RACF) for authority issues.

Solution

Verify that the user ID that is associated with the Log Forwarder has the appropriate authority to access the Log Forwarder files and directories.

BPX messages precede GLA messages in the z/OS SYSLOG

In the z/OS SYSLOG, messages with the prefix BPX precede the Log Forwarder messages, which are messages with the prefix GLA.

Symptom

The first messages that you see when you start the Log Forwarder are the following messages:

```
S GLAPROC
BPXM023I (GLALGF) GLAA001I The z/OS Log Forwarder started successfully
BPXM023I (GLALGF) GLAA002I The z/OS Log Forwarder initialization is complete
```

Cause

The user ID that is associated with the Log Forwarder start procedure has insufficient authority.

If the appropriate authority is granted to the Log Forwarder start procedure, the BPX messages do not precede the GLA messages.

Because the BPX messages precede the GLA messages, the z/OS SYSLOG Insight® Pack does not recognize the GLA messages.

Solution

If you are using the Resource Access Control Facility (RACF) as your System Authorization Facility (SAF) product, for example, either use the GLARACF sample that is provided in the SGLASAMP data set, or complete the following steps to resolve this problem:

1. In RACF, add the BPX.CONSOLE resource to the class FACILITY by using the General Resource Profiles option in the RACF Services Option Menu.
2. In the BPX.CONSOLE profile that was created (or updated) in the preceding step, add the user ID that the Log Forwarder start procedure is associated with, and assign READ access to the user ID.
3. Issue the following command to activate your changes:

```
SETROPTS RACLIST(FACILITY) REFRESH
```

After the changes are made in RACF, the z/OS SYSLOG includes the following messages when the Log Forwarder starts:

```
S GLAPROC
GLAA001I The z/OS Log Forwarder started successfully
GLAA002I The z/OS Log Forwarder initialization is complete
```

Log Forwarder message states that PPI issued return code 24

A Log Forwarder message states that the NetView for z/OS program-to-program interface (PPI) issued return code 24.

Cause

The Log Forwarder PPI receiver failed with return code 24, which can indicate that the NetView for z/OS PPI is not active.

Solution

To start the PPI, start the NetView for z/OS subsystem.

NetView message provider GLANETV issues message GLAL004E with return code 15

The NetView message provider, which is the REXX module GLANETV, issues message GLAL004E with return code 15.

Cause

The GLANETV module received return code 15, which indicates that the Log Forwarder program-to-program interface (PPI) receiver buffer for the NetView for z/OS PPI is not yet defined. The GLANETV module stops because the PPI receiver buffer is not defined.

Solution

Start the Log Forwarder before you start the NetView message provider.

NetView message provider GLANETV issues message GLAL006E

The NetView message provider, which is the REXX module GLANETV, was started, but it logged message GLAL006E and did not gather data.

Cause

Possibly, the NetView message provider tried to gather messages by using the CZR pipeline stage. Based on settings in the **DEFAULTS CZBRWAIT** command and on the amount of data in the NetView for z/OS program, the CZR pipeline stage might time out and therefore, not return any data, which causes the GLANETV module to issue message GLAL006E.

Solution

Use one of the following solutions:

- Restart the NetView message provider (the GLANETV module) in cold start mode by specifying C for the *COMMON.GLANETV.START* variable in the CNMSTYLE member, as shown in the following example:

```
COMMON.GLANETV.START = C
```


This action forces a cold start, which causes the provider to try to gather the most recent data.

- Specify a high value for the **CZBRWAIT** command by using the NetView **DEFAULTS** or **OVERRIDE** command, and restart the GLANETV module.

If neither solution resolves the problem, contact IBM Software Support.

Data Streamer does not start

Message HBO6001I, which indicates that the Data Streamer started successfully, is not present in the IBM Common Data Provider for z Systems log files, which means that the Data Streamer did not start.

Cause

The cause might be one of the following problems:

- The Data Streamer is not connected to its specified port.
- An error occurred in reading or loading the policy file.

Solution

To determine the cause, review the log files for error messages, such as the following messages, for example:

- HBO6004E
- HBO6005E
- HBO6006E

Verify that the Data Streamer port is not closed or in use. To run the Data Streamer on a different port, change the port number. For more information about this port, see [“Data Streamer port definition” on page 7](#).

Verify that the policy file is valid and correctly formatted. You can use the provided sample policy as a reference. For more information about the sample policy and policy-related files, see the following information:

- [“Setting up a working directory for the Configuration Tool” on page 18](#)
- [“Output from the Configuration Tool” on page 20](#)

System Data Engine does not start

When the System Data Engine is started, it abends with system completion code 047.

Cause

The cause might be one of the following problems:

- The SHBOLoad library is not authorized with the authorized program facility (APF). For the System Data Engine to gather SMF data, the SHBOLoad library must be authorized with APF.
- In the System Data Engine started task, an SMF in-memory resource name is specified as the source from which SMF data is to be gathered (the value for IBM_RESOURCE), but the name is incorrectly coded. Therefore, the System Data Engine assumes that the name is an SMF log stream name.

Solution

To resolve the problem, complete one or more of the following steps, as appropriate:

- Verify that the SHBOLoad library is authorized with APF. For more information, see [“Authorizing the System Data Engine with APF” on page 80](#).

- In the System Data Engine started task, verify the value for IBM_RESOURCE. For more information, see [“Creating the System Data Engine started task for streaming SMF data”](#) on page 80.

Data Streamer is not receiving data from System Data Engine

The System Data Engine started task started, but the Data Streamer is not receiving data from the System Data Engine.

Solution

Complete the following steps to determine and resolve the problem:

1. Verify that the z/OS system and relevant z/OS subsystems are configured to produce the required records, and correct the configuration as appropriate.
2. Verify that the required records are being sent to the correct SMF log stream or in-memory resource. If no records of the required type were written during the problem time period, investigate why.
3. Review HBOOUT file HBO0319I messages to confirm that the required records are being recognized by the System Data Engine. If these messages indicate that the required records are not being recognized by the System Data Engine, use the SMF dump utility to examine the SMF data that is produced during the problem time period to determine if any records of the required type were written.
4. Review HBOOUT file HBO0326I messages to confirm what streams are being produced by the System Data Engine.
5. Confirm that the Data Streamer is correctly configured to send the required records to the subscriber.
6. Check the HBOOUT file for error messages. In the System Data Engine started task, if the port value that is set for IBM_UPDATE_TARGET is incorrect, syntax errors can occur, which results in a failure to define the System Data Engine objects that are required to process and send data. The port number must be the one on which the Data Streamer listens for data from the data gathers. For more information about the port number, see [“Configuring the Data Streamer”](#) on page 68.

Stop the System Data Engine, correct the value for IBM_UPDATE_TARGET in the System Data Engine started task, and restart the System Data Engine.

For more information, see [“Creating the System Data Engine started task for streaming SMF data”](#) on page 80.

Subscriber is not receiving data

The Data Streamer successfully started and is operational, but a subscriber is not receiving data.

Cause

The cause might be one of the following problems:

- The policy file is incorrect.
- The policy file is referencing an incorrect tag to the file path for a data stream. If the trace is activated, check the logs for message HBO6021E, which indicates that an incorrect tag is present in the policy file.
- The subscriber is subscribed to the wrong streams.
- The parameters that are used to connect to the subscriber are incorrect.
- The Data Streamer cannot connect to, or is trying to reconnect to, the subscriber.
- The data packets are being discarded.
- The data gatherer (Log Forwarder or System Data Engine) is not connected to the Data Streamer.

Solution

To resolve the problem, complete one or more of the following steps, as appropriate:

- Verify that the policy file is correct. Also, verify that the subscriber is subscribed to the correct data streams, and that all parameters that are used to connect to the subscriber are correctly defined. You can use the provided sample policy as a reference.

If the policy file contains data streams that have no subscribers, the logs include message HBO6020E.

If you update the policy file, rerun the Data Streamer with the correct policy file.

- Check the logs to determine whether a connection is established between the Data Streamer and the subscriber. Message HBO6012E indicates that this connection is not established.

Verify that the subscriber host and port is correct and available for connection.

- Check the logs to determine whether a connection is established between the Data Streamer and the data gatherer (Log Forwarder or System Data Engine). Message HBO6003I indicates that this connection is established.

If the connection is established, also verify that the data gatherers are providing output to the Data Streamer.

For more information about troubleshooting the connection between the Data Streamer and the System Data Engine, see [“Data Streamer is not receiving data from System Data Engine” on page 216](#).

syslogd message problems: inconsistencies in time stamps, or missing or misplaced messages

In the forwarded syslog daemon (syslogd) messages, you notice either that the time stamps are inconsistent, or that some messages are missing or misplaced.

Symptom

When syslogd messages are sorted by time in the output, some messages are not shown in the expected order based on the time stamps.

Cause

On the z/OS system, the TZ variable for individual applications might not be correctly set to log data to syslogd.

Solution

In the syslogd file from which you want to collect data, check the time stamps of the messages that are missing or misplaced in the output. If the time stamps are inconsistent in the syslogd file, the TZ variable for individual applications is probably not correctly set to log data to syslogd.

To configure syslogd to accurately record time stamps, see the information about [Starting and stopping syslogd](#) in the [z/OS product documentation](#) in the IBM Knowledge Center.

User ID of parameter AUTHORIZED_USER is not found

When you run the `defracf.cmd` script, the existing user ID that is specified for the parameter **AUTHORIZED_USER** is not found.

Symptom

The existing user ID that is specified for the parameter **AUTHORIZED_USER** is not found when you run the `defracf.cmd` script.

Cause

This issue happens on systems where zSecure command verifier modifies the standard output of RACF's LISTUSER command. In this instance the `defracf.cmd` script is not able to find an existing ID as configured.

Solution

1. Open the `/var/cdp-uiconfig/cdpui.properties` file.
2. Find the parameter **AUTHORIZED_USER** and delete the user ID to the right of the equal sign.
3. Rerun the `defracf.cmd` script and specify G0 at the last prompt to run the RACF commands.
4. Manually run the following command:

```
CONNECT user_id GROUP(group_id)
```

Change *user_id* to the user ID that the `defracf.cmd` script couldn't find. Change *group_id* to the RACF group that is assigned to the **AUTHORIZED_GROUP**. This command allows the user specified on the connect command to access and use the configuration tool.

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)[®] are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

