This edition applies to Version 1 Release 13 of z/OS (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC33-7998-07.

© Copyright IBM Corporation 2005, 2011.
US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Contents

Figures ........................................ vii

Tables ............................................. ix

About this publication ........................ x i
Who should use this document .............. xi
How to send your comments to IBM .......... xiii
If you have a technical problem .............. xiii

Summary of changes ............................ xv
What's new in z/OS V1R13 ..................... xv
Web Services for Management ................. xv
CIM indication delivery retry ................ xv
Improve the availability of the CIM server by
better isolation from faulty providers ....... xv
Indication support for FICON ports .......... xv
Enhanced cimcli command .................... xv
New and changed z/OS-specific messages .... xvi

Part 1. Introduction and concepts .............. 1
Chapter 1. Introduction ....................... 3

Chapter 2. CIM indication concept ............ 7
Indication delivery retry ..................... 8
How indications work .......................... 8

Chapter 3. z/OS CIM security concept ......... 11

Part 2. Installation and setup ................. 15
Chapter 4. Installation ....................... 17
Migration from z/OS 1.11 or z/OS 1.12 to z/OS 1.13 .. 18
Fallback from z/OS 1.13 to z/OS 1.11 or z/OS 1.12 .. 18

Chapter 5. Quick guide: CIM server
setup and verification .......................... 19
Step 1: Setting up the security for the CIM server .... 19
Quick security setup for RACF ................ 19
Security setup for a production environment .... 20
Step 2: Customizing the file systems and directories ... 20
Step 3: Using default TCP/IP ports 5988 and 5989 .... 21
Step 4: Starting the CIM server .............. 21
Step 5: Customizing the UNIX System Services shell .... 21
Step 6: Running the installation verification program .. 21
(IVP) ............................................. 22

Chapter 6. CIM server security setup ........... 23
Defining a RACF class and profile for the CIM server .... 24
Defining a CIM server user ID .................. 25
Configuring the CIM server's resource authorization
model ........................................... 25
Enabling the must-stay-clean feature .......... 26
Setting up program control ................... 27
Granting clients and administrators access to the
CIM server ................................... 28
Switching identity (surrogate) ................ 29
Configuring the CIM server HTTPS connection
using AT-TLS .................................. 29
Example: Configuring AT-TLS for secure
communication ................................. 30
Defining the CFZAPPL profile for the APPL class .... 35
Defining an encryption key for PassTicket validation .... 35
Setting up multilevel security (MLS) support .... 35
Considering Automatic Restart Manager security .... 36

Chapter 7. CIM provider setup and
security .......................................... 37
Setting up the CIM server for RMF monitoring .... 37
Setting up the CIM server for network providers .... 38
Setting up the CIM server for Cluster, .......... 38
CoupleDataset, and JES2-JES3Jobs providers .... 38
PARMLIB updates .............................. 39
RACF setup .................................. 39
Sysplex couple dataset formatting ............. 39
Setting up the CIM server for WLM management .... 40
Setting up the CIM server for storage management .. 41
Running providers in a designated user context .... 42
Utilizing the provider based authorization model .... 43

Chapter 8. Customization ...................... 45
Configuring the ports for the CIM server ......... 45
Customizing CFZRCUST ........................ 46
Prerequisites .................................. 46
Option 1: Placing /var/wbem in a separate file
system ....................................... 46
Option 2: Using an existing file system for
/var/wbem .................................... 48
System specific directories .................... 48
Customizing the CIM server startup .......... 49
Customizing the started task procedure CFZCIM .... 49
Customizing the UNIX System Services shell .... 50
Setting the CIM server environment variables .... 51
Selecting a WLM service class for z/OS CIM
priority ....................................... 52

Chapter 9. CIM server configuration ......... 53

Chapter 10. Setup verification .................. 57

Part 3. Administration and
operation ......................................... 59

© Copyright IBM Corp. 2005, 2011
## Figures

1. Sample network environment managed with CIM management applications .... 3
2. Exemplary components of the CIM server in a z/OS environment ........... 5
3. CIM indication flow and processing. ........... 8
4. Security components ........... 12
5. Host Discovered Resources Instance Diagram 118
6. HBA instance diagram ........... 119
7. CIM Base classes extended by z/OS-specific classes (1) ........... 126
8. OS management BaseBoard Class ........... 138
9. OS management Processor classes ........... 142
10. CIM Base classes extended by z/OS-specific classes (2) ........... 147
11. OS management File System classes ........... 150
12. OS management Network classes ........... 155
13. WLM classes ........... 261
14. WLM indications ........... 262
15. CMPI provider interfaces ........... 271
16. OpenPegasus CVS Repository ........... 273
17. CIM classes from the provider registration schema ........... 278
18. z/OS CIM indication hierarchy ........... 288
19. Indication subscription class diagram ........... 289
20. Indication Handler ........... 291
## Tables

1. Default SMP/E installation directories for z/OS CIM .......................... 17
2. Access types required for CIM operations .............................................. 28
3. Sample sysplex couple dataset formatting JCL ........................................ 40
4. Installation directories for z/OS CIM ......................................................... 48
5. CIM server configuration properties ......................................................... 53
6. Log and syslog levels ........................................................................... 68
7. UCB control block information ................................................................. 149
8. Jobs providers’ reason codes ................................................................. 339
9. UML syntax ......................................................................................... 345
About this publication

This document describes the implementation of the Common Information Model (CIM) and Web Based Enterprise Management (WBEM) standards for z/OS. It explains how to set up and use the CIM server and CIM resource instrumentation provided together with the z/OS operating system. CIM is a standard data model for describing and accessing systems management data in heterogeneous environments. It allows system administrators and vendors to write applications that monitor and manage system resources in a network with different operating systems and hardware.

The focus of this document is on the z/OS-specific implementation of CIM. For more detailed information about the CIM and WBEM standards please review the information provided by the Distributed Management Task Force (DMTF), which is found in the internet on the DMTF website.

This document describes how to set up security using Resource Access Control Facility (RACF®) as security product. However, you can use any other suitable security product for this purpose.

Explicit link addresses are listed in “Related links” on page 343.

Who should use this document

This document is intended for all users of the z/OS Common Information Model (CIM). It covers all z/OS specific aspects of CIM including installation, configuration and setup, application development, and problem diagnosis.
How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:
1. Send an e-mail to s390id@de.ibm.com
3. Mail the comments to the following address:
   IBM Deutschland Research & Development GmbH
   Department 3248
   Schönaicher Str. 220
   D-71032 Böblingen
   Federal Republic of Germany
4. Fax the comments to us as follows:
   From Germany: 07031-16-3456
   From all other countries: +(49)+7031-16-3456

Include the following information:
• Your name and address
• Your e-mail address
• Your telephone or fax number
• The publication title and order number:
  Common Information Model User's Guide
  SC33-7998–08
• The topic and page number related to your comment
• The text of your comment.

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

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:
• Contact your IBM service representative
• Call IBM technical support
• Visit the IBM zSeries support website at http://www.ibm.com/systems/z/support/
Summary of changes

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations in the present release are indicated by a vertical line to the left of the change.

The following sections list the changes that have been made to CIM with various operating system releases.

What's new in z/OS V1R13

**Web Services for Management**

The CIM server now implements the WS-Transfer and WS-Enumeration operations of the Web Services for Management (WS-Management) standards. The behavior of the CIM server processing WS-Management requests is guided by these standards:

- DMTF DSP0226: Web Services for Management (WS-Management) Specification
- DMTF DSP0227: WS-Management CIM Binding Specification
- DMTF DSP0230: WS-CIM Mapping Specification

Not all mandatory requirements in these specifications are met in this version of the CIM server.

For this, the new trace component `WsmServer` has been introduced.

**CIM indication delivery retry**

With z/OS 1.13, the implementation of the DMTF Indications Profile DSP1054 1.1 introduces sequence identifiers to deliver indications more reliably.

**Improve the availability of the CIM server by better isolation from faulty providers**

The new configuration property `maxFailedProviderModuleRestarts` determines the number of times a failed provider module with indications enabled is restarted automatically before it is disabled.

**Indication support for FICON ports**

IBMzOS_PortController now supports indication subscriptions to the creation and deletion of port controllers. IBMzOS_SBInitiatorTargetLogicalUnitPath now supports the indication subscriptions to the creation, modification, and deletion of channel path connections between disk devices and z/OS channel ports.

**Enhanced cimcli command**

The cimcli command has been modified: The new function `setProperty` allows to set a single property on a named instance.
New and changed z/OS-specific messages

The following z/OS-specific messages have been added or changed:

| CEZ03000E | CEZ03001E | CEZ03002W | CEZ03003W | CEZ03004I | CEZ03005I |
| CEZ03006E | CEZ03007E | CEZ03008W | CEZ03009W | CEZ05010E | CEZ05011E |
| CEZ03012E | CFZ00409E | CFZ07801E | CFZ08001W | CFZ10033E | CFZ10206W |
| CFZ10215W | CFZ10405W | CFZ12530E | CFZ12535W |          |           |
Part 1. Introduction and concepts
Chapter 1. Introduction

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors (including IBM®) called the Distributed Management Task Force (DMTF) as part of the Web Based Enterprise Management (WBEM) initiative. WBEM includes a set of standards and technologies that provide management solutions for a distributed network environment. Interoperability is a major focus of WBEM, and using WBEM technologies can help you develop a single set of management applications for a diverse set of resources and systems.

Figure 1 shows a sample environment in which management applications can run that use the DMTF CIM standard data model.

CIM is a major component of the WBEM initiative, providing a model for describing and accessing data across an enterprise. CIM consists of both a specification and a schema. The specification defines the details for integration with other management models, while the schema provides the actual model descriptions.

CIM supports the concept of indications as described in Chapter 2, “CIM indication concept,” on page 7.

With support for the CIM server on systems running z/OS, users have the ability to access z/OS resources through an extendible industry standard model. This document contains information about how to use the CIM server for z/OS for this...
CIM for z/OS includes:

**CIM server**

The open source implementation of the CIM server manages the communication between clients and providers. The CIM server also provides several management functions, including security, and a set of commands that provide configuration and management functions to administrators.

The CIM server implementation on z/OS is based on the OpenPegasus CIM server from The Open Group. See the [OpenPegasus website](https://www.openpegasus.org) for more information.

**CIM operations over HTTP**

The "CIM over HTTP" protocol is an implementation of the standardized formats for communication between clients and the CIM server. The CIM server supports the CIM Operation and Operations over HTTP (DSP0200). The CIM server for z/OS supports most of the CIM operations defined in the CIM Operations over HTTP specification by the DMTF.

For more information about these standards, see the [WBEM website](https://www.wbem.org).

**Web Services for Management**

With z/OS 1.13, the CIM server for z/OS supports the WS-Transfer and WS-Enumeration operations defined in the WS-CIM Mapping specification. Web Services for Management (DSP0226) is a general SOAP-based protocol for managing systems. The WS-CIM Mapping specification (DSP0230) describes how to use the Web Services for Management (WS-Management) protocol to communicate with resources modeled with CIM and exposed through the XML schema mapping described by the WS-Management CIM Binding Specification (DSP0227).

**DMTF CIM Schema**

A CIM Schema defines an information model for representing systems management functions. For z/OS 1.13, CIM Schema version 2.25 is supported by the CIM server.

**Instrumentation for server resources**

Instrumentation for server resources on the system are called **providers**. The providers, which are based on a subset of the standardized CIM classes, gather data on a system. CIM clients can work with this data by accessing the providers through the CIM server. For more information about what is supported in z/OS, refer to Chapter 14, “z/OS Management Instrumentation for CIM,” on page 121.

**CIM client for Java**

z/OS CIM includes the CIM client for Java library from the SBLIM project. With z/OS 1.13, version 2.1 of the CIM client for Java is included. The CIM client for Java is a programming API that enables z/OS applications written in Java for local and remote access of CIM instrumentation through the CIM over HTTP access protocol. It consists of a Java library and associated online Java documentation.

**Note:** Version 1 of the CIM client for Java (SBLIM CIM client) will be removed in a future release of z/OS.
Figure 2 illustrates how the CIM server works in the z/OS environment: A CIM client application requests the CIM server to return information about z/OS resources, in this case about basic operating system (OS) data as well as monitoring metrics, in this example RMF™ metrics. The CIM server invokes the according CIM providers which retrieve the requested data associated to z/OS system resources. The z/OS RMF monitoring provider invokes the RMF Distributed Data Server (DDS) which in turn collects RMF Monitor III performance data. The CIM server consolidates the data from the providers and returns them back to the calling client through the CIM over HTTP protocol.

Figure 2 shows two types of CIM providers: RMF monitoring providers that use the RMF DDS to access the z/OS system, and z/OS operating system management providers that access the z/OS system data directly.
Important Note:
Each IBM eServer™ operating system is supporting a specific open source implementation of a CIM server. The "eServer Common Information Model" document contains overall information about how to use CIM for systems management on IBM eServers. Users of CIM for z/OS need to know this information. The present z/OS Common Information Model User's Guide contains the z/OS-specific supplements and deviations from the common eServer CIM and from OpenPegasus.
Chapter 2. CIM indication concept

Copyright attribution:
The introduction to CIM indications provided in this section is based on the information in the CIM Event Model White Paper, DSP0107, Document Version 2.1 June 10, 2003, provided by the Distributed Management Task Force (DMTF).

In CIM terminology, an indication is the representation of the occurrence of an event. For example, an event can be the unexpected termination of a program, or the modification of a property value of a CIM instance. There is not necessarily a one-to-one correspondence between events and indications. In particular, multiple indications can be generated for the same underlying event if multiple CIM client applications had subscribed for the event. An event can also occur without causing a related indication to be raised, for example if no subscription was made for the event.

z/OS supports additional indications for the CIM infrastructure. As an example, the Storage Management CIM providers can generate indications for the state change of channel paths, this way enabling CIM clients to support event-based monitoring to avoid polling the CIM server. A CIM client can subscribe for conditions, for example when a channel path goes offline. While the subscription is active, an according CIM indication provider monitors the resource(s) and notifies the CIM client whenever the condition becomes true.

The CIM indication support comprises the following steps:
- Defining an indication filter condition: This describes the event that you might want to be notified about, that is, when to send an indication
- Defining an indication listener: This describes how and where to send an indication
- Activating the subscription by associating a filter with a listener
- Consuming the indication once it is raised: The indication is sent to the indication listener, which decides how to react to the event

The CIM Event Model defines the CIM classes used for indication support. It defines the CIM indication class hierarchy that is used to model various types of events, and the CIM subscription mechanism.

Further readings:
- Specification for CIM Operations over HTTP describes how the CIM server transmits CIM indications to the CIM listener.
- DMTF Indications Profile DSP1054 1.1 describes the behavior of CIM indication delivery.
- Chapter 17, “CIM indications,” on page 287 describes CIM indication classes and the CIM subscription mechanism.
**Indication delivery retry**

To improve the reliability of indication delivery, DMTF Indications Profile DSP1054 1.1 introduces sequence identifiers. Sequence identifiers flag the order of deliveries. This makes indication delivery more reliable, because the CIM server can retry unsuccessful deliveries, and a CIM listener can detect lost and duplicate deliveries and reorder indications arriving out of order.

Indication delivery is based on a publish/subscribe event paradigm, where a CIM server delivers indications to subscribed WBEM listeners.

If the attempt of a WBEM server to deliver an indication to a WBEM listener fails, the service retries the delivery. For this, the number of delivery retry attempts and the minimum delivery retry interval are specified (with the DeliveryRetryAttempts and DeliveryRetryInterval properties of the appropriate CIM_IndicationService instance associated with the CIM_IndicationFilter or CIM_FilterCollection instance). Each sequence identifier has a lifetime, which is the number of delivery retry attempts multiplied by the minimum delivery retry interval multiplied by 10.

The indication is not delivered to the listener, if the number of retry attempts or the lifetime of the sequence identifier is exceeded.

For more information, see DMTF Indications Profile DSP1054 1.1.

---

**How indications work**

Indications are generated and processed as shown in Figure 3 and described in the subsequent list:

1. **Indication providers are registered:**
   - An indication provider is a CIM provider that recognizes when a particular type of event occurs on the managed system. The indication provider turns that event into a type of CIM_Indication and passes it to the CIM server.
   - An indication provider is registered with the CIM server just as any other provider is registered using PG_ProviderCapabilities as described in "Registering a provider with the CIM server” on page 277.

2. The CIM client creates the three CIM instances mentioned above:
For this, the CIM client uses the `createInstance` CIM operation. The instances must be created in the root/PF_InterOp namespace of the CIM server.

a. **To request the notification of a specific event, a CIM client defines an indication filter condition:**
   The CIM client issues CIM operation requests to the CIM server to create an instance of the CIM_IndicationFilter class.
   The CIM_IndicationFilter instance defines the event with a query string in a query language like CIM Query Language (CQL) or WBEM query language (WQL).
   For details on CQL, see the [CIM Query Language Specification](#).

b. **To specify how to handle and where to send an indication, the CIM client defines an indication listener:**
   The CIM client issues CIM operation requests to the CIM server to create an instance of the CIM_ListenerDestination class.
   A CIM_ListenerDestination is an abstract superclass that specifies how to handle and where to send the indication. It may define a destination and protocol for delivering indications, or a process to be invoked. z/OS supports the subclass CIM_ListenerDestinationCIMXML as a vehicle to describe the destination URL for indications, which can receive indications in CIMXML format.

c. **The CIM client activates the subscription:**
   The CIM client issues CIM operation requests to the CIM server to create an instance of the CIM_IndicationSubscription class.
   A CIM_IndicationSubscription is an association between a CIM_IndicationFilter and a CIM_ListenerDestination (see Figure 19 on page 289).

3. **When an event occurs on the managed system, it is detected by the CIM indication provider:**
   The CIM indication provider turns that event into a specific indication. At this stage, the indication is a local representation of an instance of a subclass of class CIM_Indication. The indication provider delivers that indication to the CIM server for further processing and delivery.
   Typically the indication is an instance of a subclass of class CIM_ProcessIndication or class CIM_InstIndication.

4. **The CIM server delivers the indications to the CIM listeners:**
   a. **The CIM server filters the indications:**
      The indications delivered by the indication provider are filtered according to the filter conditions of the active subscriptions.
   b. **The CIM server generates a CIM export message to transmit the CIM_Indication instance to the CIM listener URL according to the matching filter conditions in the format and protocol specified in the CIM_ListenerDestination instance.

5. **The CIM listener receives the CIM_Indication instance:**
   The CIM listener or CIM server coordinates the distribution of the indication to one or more registered indication consumers and sends CIM export responses.

6. **The CIM_Indication is delivered to one or more indication consumers.**
Chapter 3. z/OS CIM security concept

Although the CIM server on z/OS is based on the open source implementation, the security design has been considerably extended and adapted to meet the z/OS security strengths.

The CIM server security consists of two major areas: Protection of resources on the managed system through authentication and authorization, and protection of communicated information through network security.

The AT-TLS feature of z/OS is used to encrypt data using SSL for data security on the network. It is recommended to utilize this support.

To protect resources on the managed system from unauthorized access, first of all users have to be authenticated to ensure the CIM server is really communicating with an identified entity (user). Users can be authenticated by either a user identity (ID) and a password, a user identity and a PassTicket, or a user certificate. In all cases after successful authentication the user who wants to access the system is well known and now authorization checks are performed against that specific user identity.

The CIM server performs three types of authorization checks:

1. For each user, the CIM server checks the authority to access CIM. To get general access to CIM, a user needs at least READ access to profile CIMSERV in System Authorization Facility (SAF) class WBEM.

2. The access to the provider is checked. Access to a provider can be explicitly restricted by defining a provider-specific profile in SAF class WBEM and registering the provider with that security profile. This access restriction is optional and depends on whether a provider was registered with a security profile or not.

3. The last checks of authorization are performed based on the z/OS system resources a user tries to access, what effectively means that users can only access the resources for which they were entitled before.
Figure 4 shows the CIM server runtime environment security:

**Network security**

AT-TLS provides network security. It is recommended to utilize this feature.

**Authentication**

Authentication is always enabled for the CIM server. The CIM server checks whether the requestor is entitled to use the CIM server. A requestor authenticates with a user ID and a password, with a user ID and a PassTicket, or with a user certificate.
Authorization

**CIM authorization (RACF class WBEM)**

The CIM server controls whether the user ID is authorized to access the CIM server using the RACF class WBEM. The profile CIMSERV restricts access to the CIM server.

**Provider based authorization**

Optionally, a provider can be registered with a specific security profile. In this case, the user ID has to be authorized before it can invoke the provider. A provider-specific profile in RACF class WBEM restricts the access to the provider.

These checks are strongly recommended for providers which use a designated user ID.

**z/OS® resource authorization**

The z/OS system resource access authorization is verified against the requesting user ID.

For authorization purposes to specific z/OS system resources, the CIM server processes requests either under the user ID which has generated the request or under a designated user ID which was registered for the provider. To do this, the CIM server uses thread-level security, which is provided by the UNIX System Services.

For that reason certain providers require additional authorization to extra security profiles.

Additionally, the CIM server is enabled for the *Enhanced Security model*. Under the Enhanced Security model, the CIM server does not load any dynamic load library that is not program controlled, in particular it does not load any such provider dynamic load library.
Part 2. Installation and setup
Chapter 4. Installation

This chapter describes how to install the CIM server, how to migrate the CIM server to the current release, and how to fall back to a previous CIM server version.

Use SMP/E to install z/OS CIM for the first time or to migrate z/OS CIM as a replacement of a previous z/OS CIM version.

For details on installing a product using SMP/E, see [z/OS Program Directory](#). After successful installation, the components of z/OS CIM are located in the following hierarchical file system directory.

Table 1. Default SMP/E installation directories for z/OS CIM

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/lpp/wbem</td>
<td>Base hierarchical file system directory</td>
</tr>
<tr>
<td>/usr/lpp/wbem/bin</td>
<td>CIM server executables</td>
</tr>
<tr>
<td>/usr/lpp/wbem/lib</td>
<td>CIM server libraries</td>
</tr>
<tr>
<td>/usr/lpp/wbem/install</td>
<td>Sample profile</td>
</tr>
<tr>
<td>/usr/lpp/wbem/provider</td>
<td>CIM provider libraries provided with z/OS</td>
</tr>
<tr>
<td>/usr/lpp/wbem/schemas</td>
<td>IBM z/OS instrumentation MOF files</td>
</tr>
<tr>
<td>/usr/lpp/wbem/msg</td>
<td>CIM message files for NLS</td>
</tr>
<tr>
<td>/usr/lpp/wbem/schemas</td>
<td>DMTF CIM schema files (MOF)</td>
</tr>
<tr>
<td>/usr/lpp/wbem/repository</td>
<td>CIM schema master repository</td>
</tr>
<tr>
<td>/usr/lpp/wbem/jclient</td>
<td>CIM client for Java</td>
</tr>
<tr>
<td>/usr/lpp/wbem/IBM</td>
<td>SMP/E target library path</td>
</tr>
</tbody>
</table>

The modules CFZENF09, CFZENF27, and CFZENF33 are located in system image SYS1.LPALIB. These modules are needed by the Common Event Adapter (CEA) for the life cycle indications defined for the storage management instrumentation.

If you migrate to z/OS 1.13, the new master repository is located in /usr/lpp/wbem/repository. Previous versions of the repository are backed up as /var/wbem/repository/repository_old_<timestamp>, where <timestamp> is the current time.

If you migrate the CIM server to a new z/OS release, it is recommended that you replace the environment variable file cimserver.env located in /etc/wbem with the new sample installed in directory:

```
/usr/lpp/wbem
```

If you do not intend to replace the environment variable file cimserver.env with the new sample, make sure that the following directories are included in the LIBPATH defined in cimserver.env:

```
/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib
```
Migration from z/OS 1.11 or z/OS 1.12 to z/OS 1.13

You can install z/OS 1.13 CIM as described above as a replacement of a previous z/OS CIM version without affecting any external programs interfacing with the CIM server, such as management applications.

During startup, the z/OS 1.13 CIM server will automatically correct eventually missing file tags in its repository. In addition, it will detect if an existing repository is up to date.

If back-level, the CIM server will automatically upgrade the repository in /var/wbem following these steps:
1. The CIM server backs up the current repository into
   repository_old_<timestamp>
   where <timestamp> is the current time.
2. The CIM server copies the master repository from /usr/lpp/wbem/repository to /var/wbem/repository.
3. The CIM server migrates the previous repository content to the current repository.

If the CIM server does not find a repository in /var/wbem at startup, it automatically creates a default repository from the master repository that is shipped under /usr/lpp/wbem. To recover a damaged repository, you can create a new repository by removing the damaged repository from /var/wbem. Then the CIM server creates a new copy at the next startup. Please note that all your custom changes for the repository, for example additional provider registrations, will be lost in this case and have to be done again.

The CIM server also checks for syntactical errors in the cimserver.env file located in directory /etc/wbem/. Errors recognized by the CIM server are automatically corrected. The corrected version of cimserver.env replaces the old one, but the startup fails. Restart the CIM server.

Fallback from z/OS 1.13 to z/OS 1.11 or z/OS 1.12

The CIM server does not automatically support fallbacks to a previous version. To do so, you must recover the necessary files from repository backups.

To fall back to a previous z/OS CIM server version,
   ___ 1. Stop the CIM server
   ___ 2. Delete /var/wbem/repository
   ___ 3. Delete /var/wbem/repository_status
   ___ 4. Copy /var/wbem/repository_old_<timestamp> to /usr/lpp/wbem/repository,
         where <timestamp> is the time when you migrated from the former to the later release
         If this file is no longer available and you do not have your own backup, you can find the originally delivered version in the master repository located in /var/wbem/repository, but all your changes, such as special provider registrations, will be lost.
   ___ 5. Restart the CIM server
Chapter 5. Quick guide: CIM server setup and verification

This chapter describes the necessary setup steps of the CIM server on a z/OS system. It can be used for a quick setup - to configure CIM without the need to understand the specifics of the CIM server's features and fine-grained authorization model - or as a guide through the setup steps from security setup to customization and finally the setup verification.

To set up the CIM server for the first time, perform the following steps which are described in more detail in the chapters below:

Step 1. Set up the security for the CIM server (once per security domain/sysplex)
   - For a quick setup, use job CFZSEC from the installation SAMPLIB
Step 2. Customize the file systems and directories used by the CIM server (once per z/OS system for which you want to configure CIM)
   - Use job CFZRCUST from the installation SAMPLIB
Step 3. Use the default TCP/IP ports 5988 and 5989
Step 4. Start the CIM server (once per z/OS system)
   - Copy the CFZCIM started task procedure from the installation PROCLIB
   - START CFZCIM
Step 5. Customize the UNIX System Services shell
   - Add the content of /usr/lpp/wbem/install/profile.add to /etc/profile or to the user specific profiles residing in the user home path.
Step 6. Run the installation verification program (IVP) (once per CIM server)
   - Use job CFZIVP from the installation SAMPLIB

Step 1: Setting up the security for the CIM server

The security setup for the CIM server is done once per security domain and works for all systems that share this security domain, for example all systems that use the same shared RACF database.

Quick security setup for RACF

If you are using RACF as your security product, the quickest way to set up CIM server security is using the job CFZSEC provided in the installation SAMPLIB.

With little customization, this sample provides a working security setup for CIM, which allows you to start the CIM server and users or applications to connect to the CIM server.

Please note that the CFZSEC job is meant for a quick setup only. It is not recommended to use it as the final configuration without having reviewed the details of the CIM security setup described in Chapter 6, “CIM server security setup,” on page 23.
1. Review the CFZSEC job and customize the following steps:

   **Required updates:**
   
   a. If profile BPX.SERVER in the FACILITY class is active on your system, you should change the UID for CFZSRV to a value other than 0 in step CRUSR. In this case, the default for the UID is 9500. If the profile is not already active on your system, it is recommended to define the CIM server user with a UID of 0 in the initial setup for simplicity reasons.

      **Note:** Do not assign a password to the CFZSRV user ID.

   b. If you are using the z/OS Resource Measurement Facility™ (RMF) optional element, replace #rkeymask with a 16-digit (0-9, A-F) keymask value to set up the connectivity between CIM and RMF via PassTickets. Otherwise, you may remove the step ENRMF from the job.

      **Note:** The keymask value is a secret passkey. In a secure environment it is recommended to perform step ENRMF separately to avoid storing the passkey in the job log in readable format.

   **Optional changes:**
   
   Check that the GIDs (9501-9503) used in step CRUSR are not already in use on your system, otherwise change them.

   For details on each step of the CFZSEC job see "Appendix B. Step-by-step explanation of the CFZSEC job" on page 327

2. Submit CFZSEC

   Please note that, because this job provides a solution for each configuration, necessarily the job steps which do not apply to your system will fail. This does not affect the job's functionality.

3. Authorize users to CIM by connecting them to group CFZUSRGP

   Be sure to have at least one user authorized for CIM in order to run the Installation Verification Procedure as described in "Step 6: Running the installation verification program (IVP)" on page 22.

### Security setup for a production environment

To set up the security for a production environment, see

- Chapter 6, “CIM server security setup,” on page 23
- Chapter 7, “CIM provider setup and security,” on page 37

### Step 2: Customizing the file systems and directories

On each z/OS system where you want to start the CIM server, you need to set up the directories in the UNIX file system, where the CIM server stores its configuration and runtime data:

1. If you have installed z/OS CIM for the very first time, customize the CFZRCUST sample job from the SAMPLIB as described in "Customizing CFZRCUST" on page 46.

2. Submit the CFZRCUST sample job from the SAMPLIB

   CFZRCUST sets up the directories /etc/wbem and /var/wbem for the CIM server.
3. Change the owner of the /etc/wbem and /var/wbem directories to the CIM server user (default CFZSRV). For this, enter the following commands on the UNIX System Services command prompt from a user with superuser privileges:
   chown -R CFZSRV:CFZSRVGP /etc/wbem
   chown -R CFZSRV:CFZSRVGP /var/wbem

4. If you are setting up the CIM server for a production environment, please refer to additional customization steps as described in Chapter 8, “Customization,” on page 45.

**Step 3: Using default TCP/IP ports 5988 and 5989**

For a successful startup, the CIM server must be able to listen to the configured HTTP or HTTPS ports. Ensure that the CIM server can use the default TCP/IP port 5988 for HTTP or 5989 for HTTPS. Check if another server is listening on one of these ports, your security product is protecting these ports, or the port is blocked by the TCP/IP configuration.

“Configuring the ports for the CIM server” on page 45 describes how you can check and, if necessary, set up the port configuration.

**Step 4: Starting the CIM server**

To start the CIM server,

1. Copy the CFZCIM started task procedure from your installation PROCLIB to a data set that is part of your PROCLIB concatenation
2. Start the CIM server from the z/OS system console via the START CFZCIM command

A successful start of the CIM server is indicated (among others) by the following console messages:
   CFZ10025I: The CIM server is listening on HTTP port 5988.
   CFZ10028I: The CIM server is listening on the local connection socket.
   CFZ10030I: Started CIM Server version 2.10.0.
   CFZ12533I: The CIM server failed to register with ARM using element name CFZ_SRV_SY1: return code 0x0C, reason code 0x0160.

For a different way to start the CIM server, see “Customizing the CIM server startup” on page 49 and “Running the CIM server from the UNIX System Services command prompt” on page 62.

**Step 5: Customizing the UNIX System Services shell**

To be able to run CIM server commands, the UNIX System Services shell has to be tailored. The file /usr/lpp/wbem/install/profile.add contains the required environment variables to run CIM server commands.

To prepare the UNIX System Services shell to run CIM server commands,

   add the content of /usr/lpp/wbem/install/profile.add to /etc/profile or to the user specific profiles residing in the user home path.

For a detailed description, see “Customizing the UNIX System Services shell” on page 50.
Step 6: Running the installation verification program (IVP)

To verify that your CIM installation and customization was completed successfully, you can

- Submit the job CFZIVP contained in your installation SAMPLIB

  This job needs to run under a user that was previously authorized for CIM
  as described at the end of chapter “Step 1: Setting up the security for the
  CIM server” on page 19.

A successful CIM setup is indicated by a MAXCC=0 for the CFZIVP job along with
a success message at the end of the job output like this:

cimivp - All tests completed successfully

For a detailed description of the installation verification program, see Chapter 10,
“Setup verification,” on page 57.
Chapter 6. CIM server security setup

The z/OS implementation of the CIM server requires each requestor to have a real z/OS user ID. Only users who have been successfully authenticated with the z/OS security product and who have been granted access to the CIM server, will be able to execute requests against the CIM server. This chapter describes the details on how to set up these features.

Setting up security for the CIM server includes the following steps:

1. Define a RACF class and profile for the CIM server
   (see “Defining a RACF class and profile for the CIM server” on page 24).

2. Define a user ID for the CIM server and grant it access to the CIM server's RACF profile
   (see “Defining a CIM server user ID” on page 25).

3. Configure the CIM server's resource authorization model
   (see “Configuring the CIM server's resource authorization model” on page 25).

4. Grant client users and administrators access to the CIM server
   (see “Granting clients and administrators access to the CIM server” on page 28).

5. Allow the CIM server to surrogate for a client ID
   (see “Switching identity (surrogate)” on page 29).

6. Optionally configure secure connections (HTTPS) for the CIM server
   (see “Configuring the CIM server HTTPS connection using AT-TLS” on page 29).

7. If the APPL class for your security product is active, optionally define the CFZAPPL profile
   (see “Defining the CFZAPPL profile for the APPL class” on page 35).

8. For PassTicket usage define an encryption key for the application ID CFZAPPL
   (see “Defining an encryption key for PassTicket validation” on page 35).

9. If multilevel security (MLS) is active on your system and the CIM server UID≠0, grant the CIM server user ID READ access to security resource BPX.POE in the FACILITY class
   (see “Setting up multilevel security (MLS) support” on page 35).

10. If the CIM server is configured to use the Automatic Restart Manager (ARM) in a sysplex, you must ensure that the XCF address space has the proper authorization to perform a restart
    (see “Considering Automatic Restart Manager security” on page 36).

11. If you intend to run providers out-of-process, grant the CIM server user ID READ access to the profile BPX.JOBNAME defined in the FACILITY class
    (see “Running providers in separate address spaces” on page 62).
Defining a RACF class and profile for the CIM server

Access to the CIM server is controlled through RACF class WBEM. Define a new class in RACF through the dynamic CDT feature of the z/OS Security Server as follows:

__ 1. To be able to build the dynamic class WBEM, activate the class descriptor table (CDT) using the following RACF command:

```
Example:
SETROPTS CLASSACT(CDT) RACLIST(CDT)
```

__ 2. By adding a profile to the IBM class named CDT, you can create a new class definition. This profile then represents a dynamic class. The segment CDTINFO is used to define the class attributes. You can define the dynamic class WBEM with the following RACF commands:

```
Example:
RDEFINE CDT WBEM UACC(NONE) CDTINFO(
  CASE(UPPER)
  FIRST(ALPHA)
  OTHER(ALPHA, NUMERIC)
  MAXLENGTH(246)
  MAXLENX(246)
  KEYQUALIFIERS(0)
  PROFILEALLOWED(YES)
  POSIT(200)
  DEFAULTRC(8)
  DEFAULTUACC(NONE)
  RACLIST(REQUIRED)
)
SETROPTS RACLIST(CDT) REFRESH
```

The default values shown above (except POSIT(200)) are expected by the CIM server; do not use different values as this can yield unpredictable results.

You can ignore the warning message which is issued when adding class WBEM.

For a more detailed description of how to create a new class within RACF dynamic CDT, see [z/OS Security Server RACF Security Administrator’s Guide](#).

__ 3. To activate the new class, issue:

```
Example:
SETROPTS CLASSACT(WBEM) RACLIST(WBEM)
```

__ 4. After creating and activating the WBEM class, create the CIMSERV profile within this class. Profile CIMSERV is used to grant users access to the CIM server.

The following example illustrates the RACF commands that are required to define a profile named CIMSERV in this class:
Defining a CIM server user ID

To define a CIM server user ID:

1. Either select an existing user ID or create a new CIM server user ID. We recommend to create a CIM server user ID named CFZSRV with UID 9500 and a CIM server group ID named CFZSRVGP with GID 9501.
   
   Depending on the security model under which the CIM server runs, the user ID may need to be privileged (UID=0).
   
   For more information to decide on the privileges for the CIM server user ID, see "Configuring the CIM server's resource authorization model."

2. Allow the CIM server's user ID CONTROL access to profile CIMSERV in class WBEM.
   
   The following example shows the required RACF commands to achieve this, where the user ID CFZSRV was chosen for the CIM server:

   Example:
   ```
   PERMIT CIMSERV CL(WBEM) ACCESS(CONTROL) ID(CFZSRV)
   SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
   ```

3. If you run the CIM server as started task, it is recommended to define the CIM server user ID as protected user ID. Protected user IDs are protected from being used to log on to the system, and from being revoked through incorrect password attempts.
   
   You can define a protected user ID or change an existing user ID into a protected user ID by assigning the NOPASSWORD, NOPHRASE, and NOOIDCARD attributes through the ADDUSER or ALTUSER command.

   Example:
   ```
   ALTUSER CFZSRV NOPASSWORD NOOIDCARD NOPHRASE
   ```

For more details about protected user IDs see z/OS Security Server RACF Security Administrator's Guide.

For more information on how to associate the CIM server user ID with the started task, see "Customizing the started task procedure CFZCIM" on page 49.

Configuring the CIM server's resource authorization model

The CIM server can be run with two different authorization models, depending on whether the profile BPX.SERVER is defined in the FACILITY class or not. In any case, the CIM server follows a resource-based authorization model, which means that user requests are processed in separate threads, for which the security context is switched to the user ID of the requestor or to a designated user ID. So when a CIM provider performs a user request in such a thread, it accesses any z/OS system resource under the requestor’s or a designated user ID and thus,
authorization checks occur against this user ID. These checks are performed in addition to the general access check for the CIM server through the CIMSERV profile in class WBEM.

To let the resource based authorization security work properly, set up the CIM server user ID as follows:

__ 1. If the Enhanced Security model is disabled:

   When the Enhanced Security model is disabled, no profile BPX.SERVER is active in the FACILITY class.
   - Set up the user ID running the CIM server as a privileged user (UID=0).

If the Enhanced Security model is enabled:

   When the Enhanced Security model is enabled, profile BPX.SERVER exists in the FACILITY class, and the FACILITY class is active.

   Note: The definition of BPX.SERVER is not specific for the CIM server, but has system wide implications for all programs running on the z/OS system. Refer to Setting up the BPX.* FACILITY class profiles in z/OS UNIX System Services Planning for additional information.

   __ a. Set up the user ID running the CIM server with UPDATE access to BPX.SERVER.
   __ b. If the CIM server user ID is not privileged (UID ≠ 0), ensure that the directories /etc/wbem and /var/wbem are owned by this user ID.

The following example shows how to change ownership:

   Example:
   ```bash
   chown -R <Server UserID>:<Server GroupID> /etc/wbem
   chown -R <Server UserID>:<Server GroupID> /var/wbem
   ```

   If any of these requirements are not met, the CIM server will not start, but issue an according error message in the logs.

   __ 2. Consider to enable the must-stay-clean feature (see "Enabling the must-stay-clean feature").

   __ 3. If the Enhanced Security model or the must-stay-clean feature is enabled, make sure that the CIM server runs in a clean program controlled environment (see “Setting up program control” on page 27).

**Enabling the must-stay-clean feature**

To add additional system integrity to the CIM server, z/OS provides the optional must-stay-clean feature. To benefit from the feature, you must enable it explicitly.

**Must-stay-clean provides additional system integrity:**

- Provider libraries are loaded dynamically during runtime by the CIM server. The must-stay-clean feature prevents uncontrolled libraries to be loaded on behalf of a dynamic provider.
- Providers using the out-of-process support can be managed in separate address spaces rather than loading and calling provider libraries directly within the CIM
server process. This converts the CIM server process into a daemon process that
starts off several server processes (provider agent processes). Providers are then
run in threads by the provider agents.

Must-stay-clean secures the trust base between both address spaces.

To enable the must-stay-clean feature,

1. define the BPX.DAEMON FACILITY class in your security product

Defining BPX.DAEMON enforces program control. The following sample shows the according RACF commands:

```
Example:

SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST (FACILITY)
RDEFINE FACILITY BPX.DAEMON UACC(NONE)
SETROPTS RACLIST(FACILITY) REFRESH
```

Note: The definition of BPX.DAEMON is not specific for the CIM server, but
has system wide implications for all programs running on the z/OS
system. Refer to Setting up the BPX.* FACILITY class profiles and
Setting up security procedures for daemons in z/OS UNIX System
Services Planning for additional information.

Setting up program control

Program control means that all programs running in the address space have been
loaded from a library that is controlled by a security product. A library identified
to RACF program control is an example. Refer to z/OS UNIX System Services
Planning for additional information about program control.

If the CIM server runs with authority to BPX.SERVER or with the must-stay-clean
feature, the server must run in a clean program controlled environment.

To enable program control:

1. Ensure that all libraries are flagged as program controlled.

   By default, all libraries shipped with the CIM server are flagged as program
   controlled. If additional provider libraries are installed, it may be required to
   set the program control flag manually using the extattr +p <libname>
   command.

2. In addition to the UNIX System Services files, mark several MVS™ libraries
   as program controlled. The following sample shows the according MVS
   commands:

```
Example:

RALT PROGRAM * ADDMEM('SYS1.SCEERUN'/'******'/NOPADCHK) +
   UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCEERUN2'/'******'/NOPADCHK) +
   UACC(READ)
RDEFINE PROGRAM BLSUXTID
RALT PROGRAM BLSUXTID ADDMEM('SYS1.MIGLIB'/'******'/NOPADCHK) +
   UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH
```
If you are using z/OS Resource Measurement Facility (RMF), then the library SYS1.SERBLINK should also be program controlled.

3. Ensure that the CIM server runtime environment runs in its own address space:
   - either start the CIM server using the provided started task procedure
   - or set the environment variable _BPX_SHAREAS=NO in your z/OS UNIX System Services shell before starting the CIM server with the cimserver command.

Granting clients and administrators access to the CIM server

The CIM server authenticates users with the z/OS Security Server to determine which users can log into it. Authentication is performed for every new connection (local or remote) before a user is granted access to the CIM server.

For the CIM server for z/OS, users log on over HTTP or HTTPS using basic authentication or certificate authentication. When logging on, users are authenticated using their z/OS user ID and password as defined, for example, in RACF.

To access the CIM server, a user must be at least linked to a group with READ access to RACF profile CIMSERV. In order to use any of the administrative command-line tools of the CIM server, as described in Chapter 12, “CIM server command-line utilities and console commands,” on page 75, a group instead requires CONTROL access to the CIMSERV profile.

For detailed information about the required access rights, see the following table.

<table>
<thead>
<tr>
<th>Table 2. Access types required for CIM operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM operation type</td>
</tr>
<tr>
<td>--------------------</td>
</tr>
<tr>
<td>Basic read</td>
</tr>
<tr>
<td>Basic write</td>
</tr>
<tr>
<td>&quot;Method&quot;</td>
</tr>
<tr>
<td>Schema Manipulation</td>
</tr>
<tr>
<td>Instance Manipulation</td>
</tr>
<tr>
<td>Indication Subscription</td>
</tr>
<tr>
<td>Association Traversal</td>
</tr>
<tr>
<td>Query</td>
</tr>
<tr>
<td>Qualifier Declaration</td>
</tr>
</tbody>
</table>

The following example shows how to define UPDATE access for a client group called CFZUSRGP:
Example:

```
PERMIT CIMSERV CL(WBEM) ACCESS(UPDATE) ID(CFZUSRGP)
SETROPTS RACLST(WBEM) REFRESH
```

In addition, the CIM server user ID must be defined as a surrogate of the client user ID (see “Switching identity (surrogate)”).

To enable a user to use the command line tools, set up the UNIX System Services environment as described in “Customizing the UNIX System Services shell” on page 50.

Switching identity (surrogate)

The CIM server uses services which can be run in client or server security context. For this, the CIM server must be able to switch its user ID to the client user ID. To allow the CIM server for this, define BPX.SRV profiles for the SURROGAT class within your System Authorization Facility (SAF).

The recommended way to do this is:

__ Specify a general profile to allow the CIM server user ID to switch to any other z/OS user ID with a UNIX System Services segment defined.

The following sample shows the required RACF commands to create the generic profile, where the CIM server user ID is CFZSRV:

Example:

```
SETROPTS CLASSACT(SURROGAT) RACLIST(SURROGAT) GENERIC(SURROGAT)
RDEFINE SURROGAT BPX.SRV.** UACC(NONE)
PERMIT BPX.SRV.** CLASS(SURROGAT) ACCESS(READ) ID(CFZSRV)
SETROPTS GENERIC(SURROGAT) RACLIST(SURROGAT) REFRESH
```

Configuring the CIM server HTTPS connection using AT-TLS

The CIM server runtime environment can profit from the Application Transparent Transport Layer Security (AT-TLS) functionality. The communication between the CIM client and the CIM server can be secured by encryption (SSL). Additionally the CIM client can be authenticated by a certificate and mapped to a local z/OS user ID.

The following task describes how to configure the CIM server HTTPS connection using AT-TLS.

__ 1. Prerequisites

__ Ensure that the basic setup for the Policy Agent is done.

__ Ensure that the basic certificates setup is done.

__ 2. Configuring the CIM server runtime

Chapter 6. CIM server security setup
Set the configuration property `enableHttpsConnection` to `true`.

Ensure that the configuration property `httpsPort` is set to `5989`.
This default should not be changed.

Ensure that the `https` port `5989` can be used by the CIM server.

For more information, see "Configuring the ports for the CIM server" on page 45.

Based on this configuration, the CIM server opens a second listener for receiving client connections and ensures that these connections are secured by AT-TLS. The level of protection depends on the configuration of AT-TLS. If a connection on this port is not secured by AT-TLS, the connection is closed and an appropriate error message is issued on the operator console.

3. **Configuring the Policy Agent to secure communication for the CIM server**

   Enable the Policy Agent for AT-TLS.


   Configure the Policy Agent to secure the communication for the CIM server at the configured HTTPS port (configuration property `httpsPort`). For sample Policy Agent polices, see "Example: Configuring AT-TLS for secure communication" to configure either an SSL protection or an SSL protection including a certificate based authentication.

   Optionally you can protect the (outgoing) indication delivery on a specific port range with SSL.

**Example: Configuring AT-TLS for secure communication**

This sample shows the exemplary setup of the Policy Agent to secure communication for the CIM server.

- SSL protection only (see "Prerequisite: Common certificate setup" and "SSL protection only" on page 31)
- SSL protection including certificate based authentication (see "Prerequisite: Common certificate setup" and "SSL protection including certificate based authentication" on page 31)
- SSL protected indication delivery (see "Prerequisite: Common certificate setup" and "SSL protected indication delivery" on page 33)

For a more detailed explanation about Policy Agent AT-TLS policy see `z/OS Communications Server: IP Configuration Reference` about Policy Agent and policy applications and Application Transparent Transport Layer Security (AT-TLS) policy statements.

**Prerequisite: Common certificate setup**

To enable AT-TLS to secure the communication, a valid server certificate, the associated server private key, and the certificate of trusted Certificate Authority’s (CA) are needed. These examples are using a key ring named `CFZCIMServerRing` to store these credentials. This key ring must be accessible by the CIM server user ID (e.g. `CFZSRV`), and the server certificate must be the default certificate.

For a sample setup with RACF, see `z/OS Security Server RACF Security Administrator’s Guide` about RACF and digital certificates, implementation scenario 1 or 2. For handling certificates and key rings, please refer to the documentation of your SAF product.
**SSL protection only**

Simple SSL protection means that the communication between the client and the server is encrypted without having established a trust relationship between the client and the server. So the client still needs to send a user ID and a password for authentication.

To set up AT-TLS with simple SSL protection for the CIM server, a policy for the Communications Server Policy Agent has to be created that restricts AT-TLS to the CIM server port 5989 and to inbound TCP/IP communication.

**Sample Policy Agent policy for a simple SSL protection:**

```
Example
TTLSSRule CFZCIMServerRuleInbound
{
  Jobname CFZCIM*
  LocalPortRange 5989
  Direction Inbound
  TTLSGroupActionRef grp_StartUp
  TTLSEnvironmentActionRef CFZCIMServerEnvActionInbound
}

TTLSEnvironmentAction CFZCIMServerEnvActionInbound
{
  HandshakeRole Server
  TTLSEnvironmentAdvancedParms
  {
    ClientAuthType PassThru
  }
  TTLSKeyRingParms
  {
    Keyring CFZCIMServerRing
  }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TTLSSGroupAction grp_StartUp
{
  TTLEnabled On
  Trace 0    # Log Errors and Info messages to syslogd
}
```

CIM server specific notes to the AT-TLS Policy parameters:

**TTLSSRule: Jobname**

*Jobname* identifies where this rule applies. In the example, it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

**TTLSSRule: LocalPortRange**

This property must match the HTTPS port definition of the CIM server.

**SSL protection including certificate based authentication**

Since the CIM server is aware of AT-TLS, you can use SSL secured communications and certificates based authentication between the CIM client and the CIM server. The CIM server queries AT-TLS if the client is identified by a client certificate and mapped to a local user ID.
Authentication based on SSL certificates means:
- the communication between the client and the server is encrypted,
- the trust relationship is established, and
- the client certificate is matched to a local z/OS user ID.

No user ID and password have to be provided by the client. All subsequent authorization checking is done with the mapped user ID.

The CIM client sends an SSL certificate to AT-TLS, AT-TLS sends the certificate to RACF and RACF associates the certificate to the appropriate user ID, which then can access the CIM server. Vice versa, the CIM server returns its responses to client requests using SSL certificates.

This method of authentication provides more security than sending user IDs and passwords between client and server.

If you want to use this enhanced method based on certificates, you must create the inbound/outbound rules as follows:

To set up AT-TLS with authentication based on SSL certificates for the CIM server, a policy for the Communications Server Policy Agent has to be created that restricts AT-TLS to the CIM server port 5989 and to inbound TCP/IP communication. Also the SAF facility has to be set up to match certificate subjects to local z/OS user IDs.

For setting up the SAF facility to map certificates to local user IDs, see z/OS Security Server RACF Security Administrator's Guide about RACF and digital certificates, Certificate Name Filtering.

Sample Policy Agent policy for authentication based on SSL certificates:

```
Example

TTLSRule CFZCIMServerRuleInbound
{
  Jobname CFZCIM*
  LocalPortRange 5989
  Direction Inbound
  TTLSGroupActionRef grp_StartUp
  TTLSEnvironmentActionRef CFZCIMServerEnvActionInbound
}

TTLSEnvironmentAction CFZCIMServerEnvActionInbound
{
  HandshakeRole ServerWithClientAuth
  TTLSEnvironmentAdvancedParms
  {
    ClientAuthType SAFCheck
  }
  TLSKeyRingParms
  {
    Keyring CFZCIMServerRing
  }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TTLSGroupAction grp_StartUp
{
  TTLSEnabled On
  Trace 0 # Log Errors and Info messages to syslogd
}
```
CIM server specific notes to the AT-TLS Policy parameters:

**TTLSRule: Jobname**

*Jobname* identifies where this rule applies. In this example it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

**TTLSRule: LocalPortRange**

This property must match the HTTPS port definition of the CIM server.

**SSL protected indication delivery**

This sample shows an exemplary setup for the usage of RACF to deliver secured indications with AT-TLS.

Delivering secured indications from the CIM server to an indication listener means that the CIM server establishes an encrypted connection to deliver indications. Whether a trusted relationship is established or not depends on the listener configuration.

In case a trusted relationship is established, the CIM server is a client to the indication listener and therefore an outbound policy has to be specified with AT-TLS. To deliver secured indications, the job name of the CIM server and the port specified in the indication handler destination property must match. An indication is defined by the application programmer so there has to be an agreement between the application programmer and the system programmer that port secured indications are sent from the CIM server to the indication listeners.
Sample Policy Agent policy for the delivery of secured indications:

```cimxml
Example

TTLSRule CFZCIMServerRuleOutbound {
    Jobname CFZCIM*
    RemotePortGroupRef CFZCIMServerRemotePortGroup
    Direction Outbound
    TTLSGroupActionRef grp_StartUp
    TTLSEnvironmentActionRef CFZCIMServerEnvActionOutbound
}

TTLSEnvironmentAction CFZCIMServerEnvActionOutbound {
    HandshakeRole Client
    TTLSKeyRingParms {
        Keyring CFZCIMSServerRing
    }
}

PortGroup CFZCIMServerRemotePortGroup {
    PortRange {
        Port 5989
    }
    PortRange {
        Port 6000-7000
    }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TTLSGroupAction grp_StartUp {
    TTLSenabled On
    Trace 0   # Log Errors and Info messages to syslogd
}
```

CIM server specific notes to the AT-TLS Policy parameters:

**TTLSRule: Jobname**

*Jobname* identifies where this rule applies. In this example it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

**PortGroup**

All indications which do have a port specified within the indication handler destination property and do match to any PortRange defined within the PortGroup are delivered secure via AT-TLS. If the destination property protocol is specified as https and no other port is specified, port 5989 will be used by the CIM server. So please ensure that always port 5989 is within a PortRange. In this example, all indications with port 5989 and port 6000-7000 are delivered in a secured way.
Defining the CFZAPPL profile for the APPL class

If the APPL class for the security product is active, the CFZAPPL profile can be defined to allow only certain users to log on to the CIM server. You can manage access to the CIM server application by a profile for CFZAPPL in the APPL class with an access list that contains only those users who are allowed to use the CIM server.

In general, you need not define a profile for CFZAPPL unless you have a generic profile (*) that prevents access to applications without a more specific profile.

Defining an encryption key for PassTicket validation

The CIM server can alternatively validate a user ID and a PassTicket instead of a user ID and a password for authentication.

For more information about PassTickets, see z/OS Security Server RACF Security Administrator’s Guide.

A PassTicket is validated against an application ID. The application ID for the CIM server is CFZAPPL.

To enable CFZAPPL for the CIM server,

- Define CFZAPPL profile in the PTKTDATA class in RACF.

Example:

```
SETROPTS CLASSACT (PTKTDATA)
SETROPTS RACLIST (PTKTDATA)
RDEFINE PTKTDATA CFZAPPL -
   SSIGNON(KEYMASKED(<key>))
SETROPTS RACLIST(PTKTDATA) REFRESH
```

where <key> is the 16 digit encryption key.

Setting up multilevel security (MLS) support

In a conventional CIM server setup, all providers are processed in the CIM server's address space. If the CIM server is running in a multilevel secure (MLS) z/OS system, providers are executed in several provider agent processes depending on the user's security classification and port of entry, independent of the CIM server configuration.

Additional setup for an MLS environment:

- If the Enhanced Security model is enabled (that is, the CIM server user ID is not privileged), make sure that the CIM server user ID has READ access to security resource BPX.POE in the FACILITY class.
  This allows the CIM server to use the z/OS XL C/C++ Run-Time Library function __poe() to retrieve information on the security classification and the port of entry of a user.
Example for the security product RACF:

```
RDEFINE FACILITY BPX.POE UACC(NONE)
PERMIT BPX.POE CL(FACILITY) ACCESS(READ) ID(CFZSRV)
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) REFRESH
```

where CFZSRV is the CIM server user ID.

For general information on MLS, please refer to z/OS Planning for Multilevel Security and the Common Criteria.

If the CIM server is not running in an MLS z/OS system, and you want to run providers in processes separate from the CIM server process for stability reasons or for debugging purposes, use the out-of-process support for providers. For more information, see “Running providers in separate address spaces” on page 62.

**Considering Automatic Restart Manager security**

The z/OS CIM server is enabled for the Automatic Restart Manager (ARM).

If the CIM server is configured to use ARM in a sysplex, you must ensure that the XCF address space has the proper authorization to perform a restart. ARM must be able to issue operator commands from the XCF address space (XCFAS) to start the CIM server.

The CIM server is not running in supervisor mode. Therefore, the user ID running the CIM server must have proper SAF authorization to be allowed to register to ARM. Therefore the user ID running the CIM server also needs the SAF authorization for UPDATE access to the following FACILITY class resource:

```
Example:
IXCARM.DEFAULT.CFZ_SRV_<system_name>
```

Here is an example for entitling the CIM server user ID CFZSRV to register the CIM server for all machines within a sysplex using RACF:

```
Example:
SETROPTS CLASSACT(FACILITY) GENERIC(FACILITY)
SETROPTS RACLIST(FACILITY)
RDEFINE FACILITY IXCARM.DEFAULT.CFZ_SRV_* UACC(NONE)
PERMIT IXCARM.DEFAULT.CFZ_SRV_* CLASS(FACILITY) + ID(CFZSRV) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```
Chapter 7. CIM provider setup and security

This chapter describes additional security and setup requirements for providers:

__1. RMF provider__
   (see “Setting up the CIM server for RMF monitoring”)

__2. Network providers__
   (see “Setting up the CIM server for network providers” on page 38)

__3. Job, Cluster, and Monitoring providers__
   (see Chapter 14, “z/OS Management Instrumentation for CIM,” on page 121)

__4. Cluster, CoupleDataset, and JES2-JES3Jobs providers__
   (see “Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers” on page 38)

__5. WLM provider__
   (see “Setting up the CIM server for WLM management” on page 40)

__6. Storage management providers__
   (see “Setting up the CIM server for storage management” on page 41)

__7. Optionally, you can run providers in a designated user context__
   (see “Running providers in a designated user context” on page 42)

__8. Optionally, you can choose the provider based authorization model__
   (see “Utilizing the provider based authorization model” on page 43)

Setting up the CIM server for RMF monitoring

If you have installed RMF, you should consider the following setup for the connection of your RMF CIM providers to the RMF Distributed Data Server (DDS).

__1. The CIM monitoring providers can automatically locate an active RMF DDS in the sysplex. When the DDS is restarted on different systems through RMF management, or through manual action, the CIM monitoring providers can connect to an active DDS without additional configuration. To enable this option, comment out or omit the RMF_CIM_HOST environment variable from your cimserver.env file.

For more information on the RMF-managed DDS refer to “Starting the Distributed Data Server” in the z/OS RMF User’s Guide.

__2. The CIM monitoring providers support PassTicket authentication to the DDS. In this case the HTTP_NOAUTH option must be disabled. Secure signon through PassTickets needs to be enabled in your security manager. If you are using z/OS Security Server (RACF), the following commands can be used (for more information about configuring RACF to use PassTicket services, refer to z/OS Security Server RACF Security Administrator’s Guide):

   Activate the PTKTDATA class and the SETROPTS RACLST processing:

   Example for RACF:

   ```
   SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA) GENERIC(PTKTDATA)
   ```

© Copyright IBM Corp. 2005, 2011
Define the application GPMSERVE to your security product. The application is defined through the SAF profile GPMSERVE in class PTKTDATA. `<keymask>` is the secret passkey shared with the application.

**Example for RACF:**
```
RDEFINE PTKTDATA GPMSERVE SSIGNON(KEYMASKED(<keymask>))
SETROPTS RACLIST(PTKTDATA) REFRESH
```

Define an access profile for the PassTicket service.

**Example for RACF:**
```
RDEFINE PTKTDATA IRRPTAUTH.GPMSERVE.* UACC(NONE)
```

Grant the CIM server UPDATE access to the generic profile `IRRPTAUTH.GPMSERVE.*` in class PTKTDATA. This enables the CIM server user to create PassTickets on behalf of other users for authentication with GPMSERVE.

**Example for RACF:**
```
PERMIT IRRPTAUTH.GPMSERVE.* CL(PTKTDATA) ID(CFZSRV) ACCESS(UPDATE)
```

Activate the changes.

**Example for RACF:**
```
SETROPTS RACLIST(PTKTDATA) REFRESH
```

### Setting up the CIM server for network providers

Access to TCP/IP stack data is controlled by a security resource. Such a security resource is required if a user ID, associated with the client of the CIM server, is not defined as a z/OS UNIX superuser. The resource name is `EZB.CIMPROV.sysname.tcpname`. It is defined in the SERVAUTH class. Access is granted if the user ID associated with the client of the CIM server is permitted for READ access to the resource.

### Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers

For using the Job and Cluster providers, some additional setup has to be done.

1. Configure the Common Event Adapter (CEA):
   a. Define additional parameters in PARMLIB (see "PARMLIB updates" on page 39)
   b. Prepare RACF for CEA (see "RACF setup" on page 39)
2. When running in a sysplex, format the sysplex couple dataset to allow it to be cluster capable (see "Sysplex couple dataset formatting" on page 39).
PARMLIB updates
To enable the Job and Cluster providers, define the following PARMLIB parameters:

MAXCAD limit
This parameter defaults to 50. If the installation sets a lower limit, it may be necessary to increase this setting to accommodate the Common Event Adapter (CEA) Common Area Data Space (CADS).

APF Authorize SYS1.MIGLIB
To enable the CFRM-related CIM providers, add the following to the installation’s PROGxx PARMLIB member:

```
APF ADD DSNAME(SYS1.MIGLIB) VOLUME(******)
```

REXX Alternate Library
The Couple Dataset providers require the use of compiled REXX execs provided as part of the z/OS 1.9 SYSREXX support. These execs require the use of the REXX alternate library. The following addition to the installation’s PROGxx PARMLIB member is one way to accomplish this:

```
LNKLST ADD,NAME(LNKLST00),DSN(REXX.V1R3M0.SEAGALT),ATTOP
```

RACF setup
For using the Job and Cluster providers, RACF has to be prepared for CEA:

1. For the necessary RACF setup to permit CEA to use Automatic Restart Manager (ARM), see [z/OS Planning for Installation](#), chapter “Customizing for CEA”.

2. To configure CEA for the Cluster, Couple Dataset and JES2/JES3 Jobs CIM providers, use job CFZSEC from the installation SAMPLIB as described in Chapter 5, “Quick guide: CIM server setup and verification,” on page 19. For details see job steps PECEA and ENCLCDS in “Appendix B: Step-by-step explanation of the CFZSEC job” on page 327.

Sysplex couple dataset formatting
To format the sysplex couple dataset, use the IXCL1DSU format utility by specifying:

```
ITEM NAME(CLUSTER) NUMBER(1)
```

The following table shows a sample JCL formatting the sysplex couple dataset for enabling cluster functions. The IXCSYSPF member has been updated to indicate the new CLUSTER keyword.
Table 3. Sample sysplex couple dataset formatting JCL

```
IXCSYSPF JOB
*  
* SAMPLE JCL TO FORMAT THE PRIMARY AND/OR ALTERNATE COUPLE DATA SETS
* - SYSPLEX COUPLE DATA SETS
*  
* 1. SYSPLEX NAME IS REQUIRED AND IS 1-8 CHARACTERS
* 2. SYSPRINT DD IS A REQUIRED DD STATEMENT FOR FORMAT UTILITY
* MESSAGES
* 3. SYSIN DD IS A REQUIRED DD STATEMENT FOR FORMAT UTILITY CONTROL STATEMENTS
*  
//STEP1 EXEC PGM=IXCL1DSU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN   DD *  
  DEFINEDS SYSPLEX(PLEX1)
    DSN(SYS1.XCF.CDS01) VOLSER(CDSPK1)
    MAXSYSTEM(8)
    CATALOG
    DATA TYPE(SYSPLEX)
      ITEM NAME(GROUP) NUMBER(50)
      ITEM NAME(MEMBER) NUMBER(120)
      ITEM NAME(GRS) NUMBER(1)
      ITEM NAME(CLUSTER) NUMBER(1)
  DEFINEDS SYSPLEX(PLEX1)
    DSN(SYS1.XCF.CDS02) VOLSER(CDSPK1)
    MAXSYSTEM(8)
    CATALOG
    DATA TYPE(SYSPLEX)
      ITEM NAME(GROUP) NUMBER(50)
      ITEM NAME(MEMBER) NUMBER(120)
      ITEM NAME(GRS) NUMBER(1)
      ITEM NAME(CLUSTER) NUMBER(1)
/*
```

Setting up the CIM server for WLM management

The z/OS Workload Manager (WLM) subsystem is represented in z/OS CIM through class IBMzOS_WLM.

The provider serving class IBMzOS_WLM requires UPDATE access to resources which are protected by profile MVSADMIN.WLM.POLICY in class FACILITY.

- Permit access to MVSADMIN.WLM.POLICY by either permitting the requestor’s user ID access to the discrete profile MVSADMIN.WLM.POLICY, or to one of the generic umbrella profiles MVSADMIN.WLM.* or MVSADMIN.*, depending on your system’s current security definitions.
- Either, grant the requestor’s user ID UPDATE access to the discrete RACF profile MVSADMIN.WLM.POLICY in class FACILITY.

Example:

```
SETROPTS CLASSACT(FACILITY) RACLST(FACILITY)
PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(<client-ID>) ACCESS(UPDATE)
SETROPTS RACLST(FACILITY) REFRESH
```

- Or, grant the requestor’s user ID UPDATE access to generic RACF profile MVSADMIN.WLM.* in class FACILITY.
If your system's environment is set up for program control, the load module BLDUXTID in SYS1.MIGLIB needs to be program controlled. The following example shows how you can enable program control for load module BLSUXTID.

Example:

```plaintext
RDEFINE PROGRAM BLSUXTID
RALT PROGRAM BLSUXTID ADDMEM('SYS1.MIGLIB'/'******'/NOPADCHK) + UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH
```

A complete example for the security setup required by the CIM provider for class IBMzOS_WLM is provided in the z/OS CIM sample security setup job CFZSEC, step ENWLM.

More information:

- Chapter 15, “WLM classes,” on page 261
- “Step ENWLM” on page 336

### Setting up the CIM server for storage management

The IOS services IOSCDR and IOSCHPD have been extended for z/OS 1.13 to facilitate the retrieval of the world wide port number (WWPN) for the Initiator (IOSCHPD) and Target (IOSCDR) protocol endpoints of IBMzOS_SBProtocolEndPoint. The retrieval of the WWPN through IOSCDR is only possible under the following conditions:

__1. The used hardware is at least an IBM System z10™.

__2. The requestor or CIM client has UPDATE access to the IOSCDR profile in the FACILITY class.

Example:

```plaintext
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)
RDEFINE FACILITY IOSCDR UACC(NONE)
PERMIT IOSCDR CL(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

The SMI-S CIM life cycle indications are using the Common Event Adapter (CEA) to be notified for device path changes and insertions or deletions of FICON® channel ports.

The following setup has to be done to grant the CIM server access to CEA for the retrieval of events and IOS information:

__1. Ensure that the CEA is running in full function mode.
2. Grant the CIM server user ID `UPDATE` access to the IOSCDR profile in the
   FACILITY class.

   **Example:**
   ```
   SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) GENERIC(FACILITY)
   RDEFINE FACILITY IOSCDR UACC(NONE)
   PERMIT IOSCDR CL(FACILITY) ID(CFZSRV) ACCESS(UPDATE)
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

3. The SMI-S CIM life cycle indications are using CEA to be notified of device
   path changes and insertions or deletions of FICON channel ports. Event
   notification from CEA is protected through the following profiles in the
   RACF class SERVAUTH:
   - CEA.CONNECT
   - CEA.SUBSCRIBE.ENF_0009*
   - CEA.SUBSCRIBE.ENF_0027*
   - CEA.SUBSCRIBE.ENF_0033*

   To be permitted to subscribe for event notification by CEA the CIM server
   user ID requires READ access to these mentioned profiles. To keep your
   security setup simpler it is recommended to protect the CEA resources using
   the generic profile CEA.* instead of defining the several discrete profiles.

   Grant the CIM server user ID `READ` access to the generic profile CEA.* in
   RACF class SERVAUTH:

   **Example:**
   ```
   SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH) GENERIC(SERVAUTH)
   RDEFINE SERVAUTH CEA.* UACC(NONE)
   PERMIT CEA.* CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ)
   SETROPTS RACLIST(SERVAUTH) REFRESH
   ```

---

**Running providers in a designated user context**

Generally, the vendor of a provider (implementing a certain CIM class) defines if a
provider should run under a designated user context and also supplies the
according documentation describing the specific setup steps.

When an invocation is caused by an external CIM operation, by default the
provider is processed in the context of the *requestor's user ID*. As the provider runs
under the identity of the requestor's user ID, all resource access authorization
occurs against this user ID. So the requestor must be authorized for all resources
that a provider accesses during a request.

To avoid that a CIM client user ID needs global access to all the resources that a
provider uses for gathering data, a provider can be registered with a *designated user
ID*. The designated user ID specifies a separate security context which is used to
process the provider. The designated user ID must be authorized to access all the
resources accessed by the provider. Instead of directly using a requestor's user ID
when accessing the resource, the provider code now has to perform custom
authorization checks based on the requestor's user ID, to prevent unauthorized
access to resources. The security definitions for the designated user ID should be
similar to those of regular client users, as described in “Switching identity (surrogate)” on page 29, but it is recommend to make the designated user ID a protected user ID by disabling password, passphrase and oidcard.

Example:

```
ALTUSER <designated-user-ID> NOPASSWORD NOOIDCARD NOPHRASE
```

The properties `UserContext` and `DesignatedUserContext` of CIM class `PG_ProviderModule` specify the provider’s processing context. You can specify the values for these properties in the provider registration MOF file for each provider module. By default, it is installed at `/usr/lpp/wbem/provider/schemas/...`. For further details, see “`PG_ProviderModule`” on page 281.

**Utilizing the provider based authorization model**

When the provider based authorization model is enabled for a provider, a provider-specific profile in SAF class `WBEM` restricts the access to the provider. In this case, the requesting user ID needs special authorization before it can invoke the provider. These checks are strongly recommended for providers which use a designated user ID.

Each CIM operation needs, depending on its type, a different level of access to the security profile. For example, in order to access CIM operations that change the states of objects, WRITE access to the SAF profile defined for a provider is required. Schema manipulation is only available to users with CONTROL access to SAF profile `CIMSERV` in class `WBEM`.

You can define provider based authorization by relating a SAF profile in class `WBEM` to a single provider library. The specific SAF requirements of the provider should be documented. Unless instructed to do so, there is no need to take any configuration action for this.

To correlate a provider and a SAF profile, define a security access profile. The OpenPegasus CIM class `PG_Provider` contains a string type attribute named `SecurityAccessProfile`. Providers that register with an instance of class `PG_Provider` containing the `SecurityAccessProfile` property, must specify their SAF profile with this property in order to define it to the system. In addition, requesting users must have the according level of authorization for the named profile.

If you want to have an existing provider exploit this feature,

1. remove (unregister) the provider using the `cimprovider` utility
2. add the security profile name in property `SecurityAccessProfile` in the provider registration MOF file
3. register the provider again

The existence of a specified security profile is not checked during provider registration, but during runtime, when a request is received for the according provider.

More information:

* Table 2 on page 28 lists the type of access required for the different types of CIM operations
* “`cimprovider`” on page 81
Chapter 8. Customization

This chapter describes the customization tasks you should consider before you start the CIM server for the first time:

1. Make sure that the CIM server can use the configured HTTP and HTTPS ports (usually, port numbers 5988 and 5989)
   (see "Configuring the ports for the CIM server").

2. If you have installed z/OS CIM for the very first time, ensure that CFZRCUST has been customized during CIM server setup. If you have not already done so, it is now time to customize CFZRCUST
   (see "Customizing CFZRCUST" on page 46).

3. Ensure that you have run CFZRCUST during CIM server setup. If you have not already done so, it is now time to run CFZRCUST.

4. Customize the CIM server startup
   (see "Customizing the CIM server startup" on page 49).

5. Customize the UNIX System Services shell to be able to run CIM server commands
   (see "Customizing the UNIX System Services shell" on page 50).

6. Customize the environment variables
   (see "Setting the CIM server environment variables" on page 51).

7. Select a WLM service class for z/OS CIM priority
   (see "Selecting a WLM service class for z/OS CIM priority" on page 52).

Configuring the ports for the CIM server

Ensure that the CIM server can use the default port 5988 for HTTP or 5989 for HTTPS. You can change the default values for the ports using the httpPort and httpsPort CIM server configuration properties.

When the CIM server cannot listen to one of the ports, the CIM server startup will fail. Then check if another server is listening to the ports, your security product is protecting the ports, or the ports are blocked by the TCP/IP configuration.

To identify your currently configured port for HTTP and HTTPS, see the configuration properties httpPort and httpsPort as described in Chapter 9, "CIM server configuration," on page 53.

Use the TCP/IP NETSTAT ALLCONN PORT command to check for servers using the specified ports.

Example:

```
TSO NETSTAT ALLCONN (PORT 5988)
```

Your security product may also need to be configured to allow access to the HTTP port. For example, OEM security product ACF2 may require "Stack & Port security authorization" for the CIM server.

Please refer to your security product’s documentation for additional information.
The TCP/IP PORT and PORTRANGE statements in the TCP/IP profile may be used to make the configured HTTP port available for the CIM server's use.

For more information, refer to [z/OS Communications Server: IP Configuration Reference](#), chapter "TCP/IP profile (PROFILE.TCPIP) and configuration statements".

---

**Customizing CFZRCUST**

The job CFZRCUST installs and migrates the z/OS CIM server configuration and repository on each target machine. A sample of CFZRCUST is shipped with the default SAMPLIB.

If you have installed z/OS CIM for the first time, you need to customize CFZRCUST.

**Prerequisites**

1. The target system is running with configured UNIX System Services.
2. The CIM server is stopped.
3. The user running this job:
   - must either have UNIX user ID 0
   - or must be able to copy files and set the program control bit on files.
4. If you intend to mount the data set on a separate file system - which is recommended - this user must be entitled to allocate a 100 MB zFS data set (if not yet allocated), and must be authorized to mount file systems.

Now you have to adjust the sample job CFZRCUST, which is located in the SAMPLIB, to fit your environment. There are two options you can choose; it depends on whether you want to place the CIM server repository and the log files in a separate file system or not.

**Option 1: Placing /var/wbem in a separate file system**

To place the CIM server repository and the log files in a separate file system, perform the following steps. For a better maintainability, it is recommended to mount a separate file system on /var/wbem for the CIM server's data repository. The recommended size is 100 MB.

1. Adjust the job card.
2. Adjust STEP 1 of the JCL to create a file system data set. Choose this step to create a data set. You must provide the name in the JCL for further processing the selected sample job.
   - As an alternative, you can also create the file system outside of this JCL.
   - **STEP 1** is a sample to allocate a zFS file system dataset:
**JCL - sample STEP 1**

```plaintext
//*****************************************************************/
//* STEP 1 - Create zFS DataSet for /var/wbem */
//*****************************************************************/
//DEFZFS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//DASD0 DD DISP=(NEW,CATLG),UNIT=unit, VOL=SER=volser 
//SYSIN DD *

DEFINE CLUSTER( -
    NAME(%CFZVARWBEMDS%) -
    VOLUMES(volser) -
    STORAGECLASS(OMVS) -
    LINEAR -
    CYLINDER(150 15) -
    SHAREOPTIONS(3) - 
)

//FRMZFS EXEC PGM=IOEAGFMT,REGION=0M, 
// SYSPRINT DD SYSOUT=* 
//STDOUT DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 

__ 3. If you are using an extensible file system, you can suppress the check for 
   enough free space by specifying the parameter -noSpaceCheck in the 
   installation/migration utility at STEP 2 of the JCL. The system administrator 
   is responsible to ensure that there is enough free space (60 MB) available for 
   installation or migration, otherwise the job will fail. This will not suppress 
   the check if you use a separate file system data set. 

   The beginning of STEP 2 will then look like:

**JCL - sample STEP 2**

```plaintext
//*****************************************************************/
//* STEP 2 - Run customization/migration utility */
//*****************************************************************/
//CFZRCUST EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M, 
// SYPARM=PGM /usr/lpp/wbem/install/CFZRCUST.sh -noSpaceCheck 
//*
```

__ 4. Replace the place holder %CFZVARWBEMDS% in the JCL with the name of 
   the file system data set, for example: OMVS.VARWBEM.ZFS.

When you have submitted the job, a return code (MAXACC) 0 or 4 indicates a 
successful installation or migration. If the return code is 12, look at the job output, 
correct the error and submit the job again.

__ 5. To mount the file system for the CIM server's data repository, you can add 
a mount statement in your BPXPRMxx PARMLIB member:

**Mount statement:**

```plaintext
MOUNT FILESYSTEM(OMVS.VARWBEM.ZFS) 
   TYPE(ZFS) 
   MOUNTPOINT('/var/wbem') 
   MODE(RDWR)
```
**Option 2: Using an existing file system for /var/wbem**

To use an existing file system for the CIM server repository and the log files, perform the following steps:

1. Adjust the job card.
2. Omit STEP 1 of the sample job and specify the parameter `-noDS` in the installation/migration utility at STEP 2 of the JCL. `-noDS` disables the use of a separate file system dataset for /var/wbem.

```
JCL - sample STEP 2

//*************************************************************/
//* STEP 2 - Run customization/migration utility */
//*************************************************************/
//CFZRCUST EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
// PARM='PGM /usr/lpp/wbem/install/CFZRCUST.sh -noDS'

```

3. If you are using an extensible file system, you can suppress the check for enough free space by specifying the parameter `-noSpaceCheck` in the installation/migration utility at STEP 2 of the JCL. The system administrator is responsible to ensure that there is enough free space (60 MB) available for installation/migration, otherwise the job will fail. This will not suppress the check if you use a separate file system data set.

When you have submitted the job, a return code (MAXACC) 0 or 4 indicates a successful installation or migration. If the return code is 12, look at the job output, correct the error and submit the job again.

**System specific directories**

After successfully running CFZRCUST, the following files are located on your system:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
<th>Owner</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/wbem</td>
<td>This directory is system specific and used by the CIM server to store its configuration files and environment for the started task. It has to be owned and writable by the CIM server user (e.g. CFZSRV)</td>
<td>CIM server user</td>
<td>rwxr-xr-x</td>
</tr>
<tr>
<td>/var/wbem</td>
<td>This directory is system specific. The CIM server uses it to store its data repository for CIM classes and instances as well as for various files used at runtime, such as the special file required for connecting to the CIM server through UNIX Domain Sockets (cimxml.socket). This directory has to be owned by the CIM server user and only the CIM server user must have write access to it.</td>
<td>CIM server user</td>
<td>rwxr-xr-x</td>
</tr>
<tr>
<td>/var/wbem/logs</td>
<td>Used by the CIM server to log the stdout and stderr output when running as a started task. See &quot;Customizing the started task procedure CFZCIM&quot; on page 49 for details.</td>
<td>CIM server user</td>
<td>rwxr-xr-x</td>
</tr>
</tbody>
</table>
If the CIM server user ID is not privileged (UID ≠ 0), ensure that the directories /etc/wbem and /var/wbem are owned by this user ID.

The following example shows how to change ownership:

**Example:**

```
chown -R <Server UserID>:<Server GroupID> /etc/wbem
chown -R <Server UserID>:<Server GroupID> /var/wbem
```

### Customizing the CIM server startup

There are two ways to start the CIM server:

- either from the started task procedure CFZCIM (recommended)
- or from a UNIX System Services shell.

**If you want to start the CIM server as started task,**

- Customize the JCL procedure CFZCIM and the according environment variable file /etc/wbem/cimserver.env.
  - "Customizing the started task procedure CFZCIM" describes how to perform these steps.

**If you want to start the CIM server from a UNIX System Services shell or a remote UNIX session (telnet, SSH),**

- Customize the UNIX System Services shell
  - (see "Customizing the UNIX System Services shell" on page 50)
- Set the environment variable _BPX_JOBNAME to CFZCIM

### Customizing the started task procedure CFZCIM

You can start the CIM server via started task procedure CFZCIM. A sample of CFZCIM is shipped with the default PROCLIB.

To customize CFZCIM,

- Include CFZCIM in your PROCLIB concatenation.
- When you use the default installation directory /usr/lpp/wbem, you need not modify CFZCIM or cimserver.env. Else, you need to customize the procedure in the DD statements and also update the cimserver.env file installed in /etc/wbem to match the correct installation paths for the CIM server.
- The DDNAMEs STDOUT and STDERR in path /var/wbem/logs are used to redirect the output from the console into the UNIX file system files cimserver.out and cimserver.err. When the started task is ended, job steps two and three copy the console output to the JCL job log.
- The DDNAME STDENV points to the hierarchical file system file containing environment variables required to run the CIM server. For running the CIM server as a started task, the environment variables are set in file cimserver.env located in the /etc/wbem hierarchical file system directory. See "Setting the CIM server environment variables" on page 51 for details on how to set environment variables for the z/OS CIM server.
- To run the CIM server with a user ID for which the security setup has been completed, either set up the STARTED class or use the started procedures table (ICHRIN03).
For further details refer to the z/OS Security Server RACF Security Administrator's Guide chapter "Assigning RACF User IDs to Started Procedures".

Example of the RACF commands required to set up the CIM server for the STARTED class:

```plaintext
Example:

SETROPTS RACLIST(STARTED)
RDEFINE STARTED CFZCIM.* STDATA(USER(CFZSRV) GROUP(CFZSRVGP))
SETROPTS RACLIST(STARTED) REFRESH
```

### Customizing the UNIX System Services shell

You need to customize the UNIX System Services shell, not only if you want to start the CIM server from here.

All commands of the z/OS CIM server are UNIX style programs running in a UNIX System Services shell and executing in the Enhanced ASCII mode. This means that all string data is represented in ASCII rather than in EBCDIC encoding. To be able to execute z/OS CIM server commands, a UNIX System Services shell has to be started and the environment has to be set up to enable automated ASCII-EBCDIC translation and to find the necessary libraries and executables.

There are two ways to set up a shell for CIM server commands:

- In the UNIX System Services, or
- Using BPXBATCH in a JCL job

**Setting up a shell in the UNIX System Services:**

The file `/usr/lpp/wbem/install/profile.add` contains the basic settings to enable z/OS CIM server commands. You can add the contents of `profile.add` to `/etc/profile` to set up the z/OS CIM server environment for all users of the UNIX System Services shell or to the individual profile in the home path of each user who wants to use the commands.

**Setting up a shell using BPXBATCH in a JCL job:**

Use the utility BPXBATCH to run CIM server commands using a JCL job.

**Example to run the cimivp utility:**

```plaintext
//STEP1 EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
//       PARM='PGM /usr/lpp/wbem/bin/cimivp 127.0.0.1'
//       ENV DD /etc/wbem/cimserver.env'
//       STDENV DD /etc/wbem/cimserver.env'
//       STDOUT DD SYSOUT**
//       STDERR DD SYSOUT**
//       CEEDUMP DD SYSOUT**
//       SYSPRINT DD SYSOUT**
//       SYSDUMP DD SYSOUT**
//       SYSMKDUMP DD SYSOUT**
```

The file `/etc/wbem/cimserver.env` contains the basic settings for the BPXBATCH environment. You can find an alternative example for the usage of BPXBATCH in job CFZRCUST in SYS1.SAMPLIB.

More information:

See "Setting the CIM server environment variables" on page 51 for details on CIM server specific environment variables.
Setting the CIM server environment variables

Environment variables are set in file cimserver.env, if the CIM server runs as started task. If you use the CIM server from the UNIX System Services command prompt, the environment variables are set in UNIX System Services .profile in the home path of the user ID which starts the CIM server.

Setting the trace variables is not required for normal operation.

Note that changes to the environment variables become effective only after a restart of the CIM server.

The environment variable file cimserver.env is located in the hierarchical file system at /etc/wbem/. After installation, you can still find the originally shipped version in /usr/lpp/wbem/. 

Set the following environment variables contained in this file to start the CIM server:

PEGSUS_HOME
Must be set to the hierarchical file system directory where the CIM server is installed. By default this is /usr/lpp/wbem.

LIBPATH
Must include the CIM server's lib and provider hierarchical file system directory paths. By default this is set to /usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib

OSBASE_TRACE
Defines the trace level for the z/OS OS management CIM instrumentation. Valid values range from 0 through 4, where 4 provides the most details.

OSBASE_TRACE_FILE
Defines the filename for the z/OS CIM instrumentation traces.

PATH
Only for running the CIM server or any of the CIM server command-line utilities in UNIX System Services. Must include the CIM server's bin hierarchical file system directory path so that the CIM server's executable programs are automatically found when you enter the according command at the UNIX System Services command prompt. By default this is set to /usr/lpp/wbem/bin.

The following variables starting with RMF_ only apply when RMF is installed and you use the RMF monitoring providers:

RMF_CIM_PROVIDER
Used to control the behavior of the RMF CIM providers when RMF is installed. By default, the RMF CIM provider is enabled. To disable the RMF CIM provider, set the environment variable RMF_CIM_PROVIDER=DISABLE.

RMF_CIM_HOST
Defines the target TCP/IP address or hostname of the z/OS MVS image on which the DDS responsible for this system is running. Beginning with z/OS 1.11 the use of this environment variable is no longer required, but it will be used if defined. If omitted, the CIM monitoring providers can...
automatically locate an active RMF DDS in the sysplex, provided all systems in the sysplex run z/OS 1.10 or higher.

**RMF_CIM_PORT**
Defines the TCP/IP port number of the DDS (default: 8803). Starting with z/OS 1.11 no longer required, but used when defined.

**RMF_CIM_TRACE**
Defines the trace level of the RMF CIM provider. Valid values range from 0 through 4, with 0 providing no trace and 4 providing all information possible.

**RMF_CIM_TRACE_FILE**
Defines the file name for storing the trace data for the z/OS RMF CIM instrumentation.

**RMF_CIM_BENCH**
Is used for performance benchmarks, for example, to identify the response time of the underlying RMF infrastructure. If this variable is set to 1, the RMF CIM provider will print some benchmarking information about various RMF operations, suitable for RMF development.

**RMF_INDICATION_RESTTIME**
Specifies the time in seconds that a provider should sleep after checking an indication subscription and before checking the next one, in order to reduce system load. The default is one second.

**RMF_INDICATION_POLLING_INTERVAL**
Specifies the time interval in which every single subscription is checked at most once. The default is 100 seconds.

The following variables starting with **WLM_** only apply when Workload Manager (WLM) is installed and you use the WLM providers:

**WLM_CIMPROVIDER_TRACE_FILE**
Defines the output file name for z/OS WLM provider traces. The default trace file is `/var/wlmprovider.trc`.

**WLM_CIMPROVIDER_TRACE_LEVEL**
Defines the trace level for the z/OS WLM provider. Valid values range from 0 through 5, where 5 provides the most details. The default is 0, meaning that no trace is written.

---

### Selecting a WLM service class for z/OS CIM priority

If you plan to use the z/OS CIM server as part of your monitoring or management infrastructure, it should run at a priority higher than the work to be managed. You should classify the CIM server into a single period service class with a velocity goal at an appropriate importance level.
Chapter 9. CIM server configuration

Configuration properties are used to control the behavior of the CIM server. The default configuration setting for the CIM server works for the majority of environments. Table 5 describes the configuration properties.

You can display or change the configuration settings using
- the cimconfig UNIX System Services command
- or the MODIFY console command

Column “dynamic Y/N” indicates if a configuration property is dynamic or not.
- Dynamic configuration properties can be changed while the CIM server is running.
- For those properties which you cannot dynamically change, use
  - either the -p parameter of the cimconfig command,
  - or the PLANNED option of the MODIFY command.

to indicate your change. Then stop and restart the CIM server.

More information:
- “cimconfig” on page 79
- “MODIFY console command” on page 113
- “Changing current configuration properties” on page 63
- “Changing planned configuration properties” on page 63

Table 5. CIM server configuration properties

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default value</th>
<th>dynamic Y/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>daemon</td>
<td>The foreground/background process property. Set daemon to 'false' to run the CIM server as foreground process or as a started task.</td>
<td>true</td>
<td>N</td>
</tr>
<tr>
<td>enableAuditLog</td>
<td>When this option is set to true, the CIM server is writing SMF records 86. For details see “Audit logging with SMF record 86” on page 69.</td>
<td>false</td>
<td>Y</td>
</tr>
<tr>
<td>enableHttpConnection</td>
<td>The HTTP connection to the CIM server. Enables and disables connections to the CIM server over HTTP. When turned off only local connections are accepted.</td>
<td>true</td>
<td>N</td>
</tr>
</tbody>
</table>
Table 5. CIM server configuration properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default value</th>
<th>dynamic Y/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableHttpsConnection</td>
<td>The HTTPS connection to the CIM server. Enables and disables secure connections to the CIM server via HTTPS. Note that it is not sufficient to turn on this option, but you must also enable an SSL connection through the AT-TLS feature at the z/OS Communications Server as described in “Configuring the CIM server HTTPS connection using AT-TLS” on page 29. Note: When set to true, ensure that the configured httpsPort can be used by the CIM server.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>enableIndicationService</td>
<td>‘true’ means the indication service is enabled. ‘false’ will disable the indication service.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>enableRemotePrivilegedUserAccess</td>
<td>The remote privilege for users. Enables and disables remote access for users with UID 0.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>forceProviderProcesses</td>
<td>When this option is set to ‘true’, providers will run in one or more separate address spaces. For details see “Running providers in separate address spaces” on page 62. This option is ignored when MLS support is activated. The out-of-process provider support uses then one address space per security label for full protection of classified documents and information.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>httpPort</td>
<td>The port to listen for HTTP requests. It is recommended not to change this value. Note: Make sure that the configured httpPort can be used by the CIM server.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>httpsPort</td>
<td>The port to listen for HTTPS requests. AT-TLS should be configured to use this port. It is not recommended to change this value. This value is only active if enableHttpsConnection is set to true. Note: Make sure that the configured httpsPort can be used by the CIM server.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>idleConnectionTimeout</td>
<td>The timeout value in seconds that the CIM server uses to wait for idle client connections to close. A client connection is considered as idle when it is not in the process of sending a request and when the CIM server is not processing a request from that connection. If the value is set to 0, no timeout is used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>logLevel</td>
<td>The detail level for logging. Possible values are INFORMATION, WARNING, SEVERE, FATAL, or TRACE (see also “Logging” on page 67).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: When set to true, ensure that the configured httpsPort can be used by the CIM server.
<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default value</th>
<th>dynamic Y/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxFailedProviderModuleRestarts</td>
<td>The number of times a failed provider module with indications enabled is restarted automatically before it is moved to the state Degraded. If this value is zero, the failed provider module is moved to the state Degraded immediately.</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>maxProviderProcesses</td>
<td>The maximum number of separate address spaces for running providers. Only in effect if <code>forceProviderProcesses</code> is set to <code>TRUE</code>. If the value is set to 0, the number is unlimited.</td>
<td>0</td>
<td>Y</td>
</tr>
<tr>
<td>messageDir</td>
<td>The message bundle directory. Do not change the default.</td>
<td>msg</td>
<td>N</td>
</tr>
<tr>
<td>providerDir</td>
<td>The name of the directory where the provider libraries reside. You can specify multiple directories here, separated by a colon (':'). Provide the full path for all directories when changing the default. Since the CIM server has its own set of providers, its lib directory always needs to be present in the list of provider directories. When adding new provider directories, it is also recommended to update the LIBPATH environment variable according to the new values of providerDir. This is required, because a provider may need other supplemental dynamic load libraries, which the CIM server is not aware of and therefore would otherwise fail to load.</td>
<td><code>lib:provider</code></td>
<td>Y</td>
</tr>
<tr>
<td>repositoryDir</td>
<td>The name of the directory for the repository. /var/wbem/repository</td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>repositoryIsDefaultInstanceProvider</td>
<td>The CIM server repository serves as the default provider for CIM instances when no dynamic provider has been registered for a CIM class.</td>
<td>true</td>
<td>N</td>
</tr>
<tr>
<td>shutdownTimeout</td>
<td>The timeout value in seconds that the CIM server uses to wait for the shutdown process to complete. This value includes terminating active providers.</td>
<td>30</td>
<td>Y</td>
</tr>
<tr>
<td>slp</td>
<td>The CIM server uses the SLP Protocol to announce itself over the network.</td>
<td>false</td>
<td>N</td>
</tr>
<tr>
<td>socketWriteTimeout</td>
<td>The timeout value in seconds that the CIM server uses to wait for a client to receive data from the socket. After the timeout the CIM server will close the socket.</td>
<td>20</td>
<td>Y</td>
</tr>
<tr>
<td>Property name</td>
<td>Description</td>
<td>Default value</td>
<td>dynamic Y/N</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>traceComponents</td>
<td>This option specifies the component(s) that you want to trace. The value ALL enables tracing for all components. For more information refer to section &quot;Tracing&quot; on page 64, which also lists the valid components.</td>
<td>All</td>
<td>Y</td>
</tr>
<tr>
<td>traceFacility</td>
<td>This option specifies the trace destination.</td>
<td>Memory</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td><strong>FILE</strong> saves the tracing messages to the file specified in traceFilePath</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>LOG</strong> saves the tracing messages to the logging facility, if logLevel is set to TRACE (see &quot;Logging&quot; on page 67). This alternative combines the tracing message stream with the log messages.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>MEMORY</strong> saves tracing messages in a wrap around memory buffer. This buffer is included in memory dumps. Specify the size of the allocated memory with the traceMemoryBufferKbytes property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>traceFilePath</td>
<td>This property specifies the fully qualified file which saves the trace data.</td>
<td>/tmp/cimserver.trc</td>
<td>Y</td>
</tr>
<tr>
<td>traceLevel</td>
<td>Switches tracing on or off, and sets the trace level of detail. Choose one of the following trace levels:</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>0    Tracing is off</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1    Severe errors</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2    Warning level error messages</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3    Inter-function logic flow, medium data detail</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4    High data detail</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5    High data detail, method enter and exit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: This does not include tracing for the providers. See also &quot;Tracing&quot; on page 64.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>traceMemoryBufferKbytes</td>
<td>Specifies the size of the memory area which is reserved for tracing messages in kB (1kB=1024B). The value must be at least 16. This value only becomes valid when traceFacility=MEMORY.</td>
<td>10240</td>
<td>N</td>
</tr>
</tbody>
</table>
Chapter 10. Setup verification

After performing the customization actions, you can start the CIM server as described in "Step 4: Starting the CIM server" on page 21 and run the sample application CIMIVP delivered with the product as an installation verification program.

The client application CIMIVP is delivered as executable with the product in file /usr/lpp/wbem/bin/cimivp. It displays some of the information about the z/OS system which is available through CIM.

You invoke this program as job CFZIVP contained in SYS1.SAMPLIB or from the UNIX System Services command line as cimivp.

On successful completion, it generates an output similar to the one shown hereafter.

```
cimivp Main started ...
Connecting to local CIM Server ...
... success
> Found Computer System : BOEPEG4.boeblingen.de.ibm.com
  (CPUID: 0C0B822097, VMGuestID: PEG4)
> Found Operating System : PEG4 (Version: 01.12.00,
  Sysplex: PEG4PLEX, FreeMem: 1303824)
> Number of active UNIX System Services processes: 17
> Number of active address spaces: 110
> Number of FC ports: 20
> Number of online processors: CP(3) zAAP(0) zIIP(1)
> Number of online disk volumes: 80
```
cimivp - All tests completed successfully.

If the execution of cimivp times out, this may be caused by a slow IP hostname resolution or a large amount of managed resources, like for example disks. To override the default timeout, you can set the environment variable CIM_IVP_TIMEOUT to the amount of seconds that cimivp should wait for a response from the CIM server before it fails with a timeout. When you run cimivp by submitting the CIMIVP sample job, you can add the CIM_IVP_TIMEOUT variable to file /etc/wbem/cimserver.env like this:

```
CIM_IVP_TIMEOUT=300
```

This sets the timeout for cimivp to 5 minutes.
Part 3. Administration and operation
Chapter 11. CIM server administration

While you must set up the CIM server only once to make it ready to use, you can configure your CIM server environment as often as you want during operation to best meet your requirements. The CIM server provides the ability to set a number of configuration options. Many tasks and operations for the CIM server are performed under z/OS UNIX System Services, ideally within a telnet session.

More information:
- To configure the CIM server, you can use the commands described in Chapter 12, “CIM server command-line utilities and console commands,” on page 75.
- To use the command line tools, be sure that you have set up the UNIX System Services environment as described in “Customizing the UNIX System Services shell” on page 50.
- If you run into problems while setting up or using the CIM server you can find information for problem solving in “Appendix A. Troubleshooting” on page 323.

Starting and stopping the CIM server

Start the CIM server either as a started task or from the UNIX System Services command prompt, as described in the following sections.

Running the CIM server as started task

The standard way to start the CIM server on z/OS is through the started task CFZCIM.

Before the first start:
- Make sure that you have customized the procedure CFZCIM and cimserver.env before you start the CIM server for the first time as described in “Customizing the started task procedure CFZCIM” on page 49 and “Setting the CIM server environment variables” on page 51.

Starting the CIM server:
- Enter the following command from the z/OS console:
  
  S(TART) CFZCIM

Verifying a successful start:
- After a successful start of the CIM server, the following message is shown on the console and issued to the syslog:
  
  CFZ10030I: Started CIM server version 2.11 for z/OS.

Stopping the CIM server:
- When the CIM server was started through CFZCIM, you can also stop it from the console by entering
  
  (STO)P CFZCIM
Running the CIM server from the UNIX System Services
command prompt

Before the first start:

- Make sure you have completed the configuration steps described in
  “Customizing the UNIX System Services shell” on page 50.
- Ensure that you have set the environment variable _BPX_SHAREAS to NO in
  your z/OS UNIX System Services shell to run the CIM server runtime
  environment in its own address space.

Starting the CIM server:
- Type the cimserver command at the command prompt of a z/OS UNIX
  System Services shell.

Verifying a successful start:
- After a successful start of the CIM server, the following message is shown on
  the console and issued to the syslog:
  CFZ10030I: Started CIM server version 2.11 for z/OS.

Stopping the CIM server:
- At the command line, enter: cimserver -s

Running providers in separate address spaces

In a conventional CIM server setup, all providers are processed in the CIM server's
address space. Only when the CIM server is running in a multi level secured (MLS)
z/OS system, providers are executed in several provider agent processes
depending on the user's security classification and port of entry, independent of
the CIM server configuration.

If the CIM server is not running in an MLS system, you may want to run CIM
providers in separate processes to protect the CIM server from failing CIM
providers or to protect the CIM providers from each other. Rather than loading
and calling CIM provider libraries directly within the CIM server process one or
more provider agent processes are then started that will run the CIM provider
code. In this case you can enable the out-of-process support (OOP) for providers.
This is an enhanced version of the OpenPegasus out-of-process provider feature

To turn on out-of-process support,
- Set the configuration property forceProviderProcesses to true.
  (See “Changing planned configuration properties” on page 63)

If the Enhanced Security model is enabled (that is, the CIM server is not
privileged),
- Grant the CIM server user ID READ access to the profile BPX.JOBNAME
defined in the FACILITY class.
  This allows the CIM server to set the job name of the out-of-process agent to
  CFZOOPA.
Example for the security product RACF:

```
RDEFINE FACILITY BPX.JOBNAME UACC(NONE)
PERMIT BPX.JOBNAME CL(FACILITY) ACCESS(READ) ID(CFZSRV)
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) REFRESH
```

where CFZSRV is the CIM server user ID.

When the out-of-process support is enabled, the z/OS-specific provider property `ShareAS` and the property `ModuleGroupName` for class PG_ProviderModule are used. These properties specify whether a provider should run in its own address space, optionally grouped with other providers, or should be processed in the CIM server address space. They are set during provider registration via the registration MOF file. `ModuleGroupName` can also be set dynamically at runtime using the `-g` option of the `cimprovider` command.

### Changing current configuration properties

You can update the current configuration while the CIM server is running for dynamic properties.

Use the `cimconfig` UNIX System Services shell command or the `MODIFY` console command to dynamically change the current configuration properties of the CIM server.

Using the `cimconfig` command without the `-p` option or the `MODIFY` console command without the PLANNED option results in a non-permanent change. With a restart of the CIM server these changes are reset to the planned configuration values. For making permanent changes, change the planned configuration values.

More information:

- Chapter 9, “CIM server configuration,” on page 53
- "cimconfig" on page 79
- “MODIFY console command” on page 113

### Changing planned configuration properties

To change the values of the planned configuration properties - these are the permanent values of configuration properties which are used at the CIM server startup - use

- the `cimconfig` UNIX System Services shell command with the `-p` option or
- the `MODIFY` console command with the PLANNED option.

The use of the `cimconfig` command is independent of whether the CIM server is running or stopped. If you change the planned configuration properties while the CIM server is running, those changes do not take effect until the CIM server is restarted. Then the planned configuration properties become the current configuration properties.

In order to use the `MODIFY` console command, the CIM server must be running. When you use the `MODIFY` console command with the PLANNED option, your changes do not take effect until the CIM server is restarted. Then the planned configuration properties become the current configuration properties.
Tracing

To enable or to modify tracing, use the `cimconfig` command or the MODIFY console command. You can modify the tracing configuration properties while the CIM server is running.

See also “cimconfig” on page 79 and “MODIFY console command” on page 113.

You can modify the following tracing configuration properties:

- **traceLevel**
  - turns tracing on and off and specifies the trace level. You can choose among the following trace levels:
  - 0: Tracing is off
  - 1: Severe errors
  - 2: Warning level error messages (default)
  - 3: Inter-function logic flow, medium data detail
  - 4: High data detail
  - 5: High data detail, method enter and exit

- **traceComponents**
  - specifies the components that you want to trace.
  - You can choose one or more of the following components, separated by comma:

    | All                      | Authentication          |
    |-------------------------|-------------------------|
    | Authorization           | BinaryMessageHandler    |
    | CIMExportRequestDispatcher | CIMOMHandle            |
    | CMPIProvider            | CMPIProviderInterface   |
    | Config                  | ControlProvider         |
    | CQL                     | DiscardedData           |
    | Dispatcher              | ExportClient            |
    | Http                    | IndicationFormatter     |
    | IndicationGeneration    | IndicationHandler       |
    | IndicationReceipt       | IndicationService       |
    | IPC                     | L10N                    |
    | Listener                | LogMessages             |
    | MessageQueueService     | ObjectResolution        |
    | OsAbstraction           | ProviderAgent           |
    | ProviderManager         | Repository              |
    | Server                  | Shutdown                |
    | SSL                     | StatisticalData         |
    | Thread                  | UserManager             |
    | WsmServer               | WQL                     |
    | Xml                     | XmlIO                   |


The following components have a special purpose:

<table>
<thead>
<tr>
<th>Special purpose trace components</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Traces all available components</td>
</tr>
<tr>
<td>DiscardedData</td>
<td>Issues a trace message when information is discarded or an operation is cancelled</td>
</tr>
<tr>
<td>LogMessages</td>
<td>Traces all messages written to the logging facility</td>
</tr>
<tr>
<td>StatisticalData</td>
<td>Prints statistical data to the trace at level 4</td>
</tr>
<tr>
<td>XmlIO</td>
<td>Prints the complete CIM-XML messages</td>
</tr>
</tbody>
</table>

**traceFacility**

specifies the destination of the trace messages.

**FILE** saves the trace messages to the file specified in `traceFilePath`.

This file is continuously growing. You can remove it while the CIM server is running. It will be recreated automatically.

**LOG** saves the trace messages to the logging facility, if the `logLevel` is set to TRACE (see “Logging” on page 67). This alternative combines the log messages and the trace messages to one message stream.

**MEMORY** saves trace messages in a wrap around memory buffer. This buffer is included in memory dumps. (default).

To find the trace in a memory dump, the top of the allocated memory block is flagged with “PEGASUSMEMTRACE”. The last trace message is flagged with the suffix "EOTRACE". The flags are encoded in ASCII.

Specify the size of the memory buffer with the static `traceMemoryBufferKbytes` property.

**traceFilePath**

if `traceFacility`=FILE, this property specifies the file which saves the trace data. The default is `/tmp/cimserver.trc`.

**traceMemoryBufferKbytes**

specifies the size of the memory area which is reserved for trace messages in kB (1kB=1024B). The default is 10240. The value must be at least 16. `traceMemoryBufferKbytes` is a planned configuration property (see "Changing planned configuration properties" on page 63).

This area is allocated when `traceFacility`=MEMORY.

**Tracing providers running out-of-process:**

When tracing is enabled in the CIM server, it is also enabled in the provider agent processes. For reasons of trace data integrity and regarding performance aspects, a separate trace file is used for each provider agent process.
Each provider agent is uniquely identified by the name of the shared provider agent executable. Each non-shared instance of a provider agent corresponds with a single provider module. This name is used as an extension to the trace file name specified by the traceFilePath configuration property. For example, if traceFilePath is defined as /tmp/cimserver.trc, the non-shared provider agent for the OperatingSystemModule would direct its trace output to the file /tmp/cimserver.trc.OperatingSystemModule.

Examples:

To set the trace level to trace all information with high data detail in the Thread and ProviderManager components,
type the following commands into the UNIX System Services shell:
cimconfig -s traceLevel=4
cimconfig -s traceComponents=Thread,ProviderManager

or
F CFZCIM,APPL=CONFIG,traceLevel=4
F CFZCIM,APPL=CONFIG,traceComponents='Thread,ProviderManager'

on the console.

To disable all tracing,
type the following command into the UNIX System Services shell:
cimconfig -s traceLevel=0

To route both trace and log messages to a file:
type the following commands into the UNIX System Services shell:
cimconfig -s traceLevel=1

cimconfig -s traceComponents=Thread,ProviderManager,LogMessages
cimconfig -s traceFacility=FILE
cimconfig -s traceFilePath=/tmp/cimservr1.trc

The CIM server now saves severe trace messages in the Thread and ProviderManager components and all log messages to the file /tmp/cimservr1.trc.

To route both trace and log messages to memory:
type the following commands into the UNIX System Services shell:
cimconfig -s traceLevel=1

cimconfig -s traceComponents=Thread,ProviderManager,LogMessages
cimconfig -s traceFacility=MEMORY

The CIM server now saves severe trace messages in the Thread and ProviderManager components and all log messages to the default memory space of 10240kB.

To route both trace and log messages to the z/OS Communications Server's system logger (syslog) daemon:
1. configure the syslog daemon as described in z/OS Communications Server: IP Configuration Reference and z/OS Communications Server: IP Configuration Guide
2. type the following commands into the UNIX System Services shell:
cimconfig -s logLevel=TRACE
cimconfig -s traceLevel=1
cimconfig -s traceComponents=Thread,ProviderManager
cimconfig -s traceFacility=LOG
The CIM server now writes severe trace messages in the *Thread* and *ProviderManager* components and all log messages to the syslog daemon.

See also “Logging.”

### Logging

The CIM server sends log messages

- to the *z/OS system console*,
- to `stderr`,
  - if the CIM server is run as a started task. The logs are captured in `/var/wbem/logs/cimserver.err`.
- to the *z/OS Communications Server’s system logger (syslog) daemon*,
  - if the syslog daemon is configured as described in *z/OS Communications Server: IP Configuration Reference* and *z/OS Communications Server: IP Configuration Guide*,
- and to the *trace facility*,
  - if `traceComponents` includes the element `LogMessages`, (see also “Tracing” on page 64).

Generally logging for the CIM server is enabled and cannot be turned off. However, you can configure the level of logging.

**To modify the log level**

use the `cimconfig` command or the `MODIFY` console command to change the `logLevel` configuration property.

**Examples**

- type the following command into the UNIX System Services shell while the CIM server is running:
  
  `cimconfig -s logLevel=INFORMATION`
- or type the following command into the *z/OS system console*:
  
  `F CFZCIM,APPL=CONFIG,LogLevel=INFORMATION`

See also “`cimconfig`” on page 79 and “`MODIFY console command`” on page 113.

**Log levels**

You can choose between five different log levels:

**INFORMATION (default)**

The default setting for `logLevel` is `INFORMATION`. This setting should not be changed unless there is a specific need for a more or less detailed logging.

**WARNING**

returns log messages for warnings, severe and fatal errors

**SEVERE**

returns log messages for severe and fatal errors

**FATAL**

returns log messages only for fatal errors

**TRACE**

returns all log messages and all trace messages

  - trace messages are only routed to the *z/OS Communications Server’s system logger (syslog) daemon* - never to the system
Using the syslog daemon for CIM server logging

The z/OS CIM server will connect to the syslog daemon and send all of its log messages to it, where the filtering according to the logLevel configuration property applies as described above. Therefore no messages will be submitted to the syslog daemon which have a higher log level than what’s specified in the current value of the logLevel configuration property.

Messages that go to the syslog daemon are prepended with an according z/OS message number, which is either one of the generic CFZ00001E, CFZ00002W or CFZ00004I messages followed by a PGSxxxxx message number, or one of the directly mapped z/OS specific CFZxxxxx message numbers.

Syslog messages from the z/OS CIM server will have an identifier of “CFZCIM” and also contain the CIM server’s process ID.

The log levels of the z/OS CIM server are mapped to the following syslog levels:

<table>
<thead>
<tr>
<th>Log level</th>
<th>Syslog level</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFORMATION</td>
<td>LOG_INFO</td>
</tr>
<tr>
<td>WARNING</td>
<td>LOG_ERR</td>
</tr>
<tr>
<td>SEVERE</td>
<td>LOG_WARNING</td>
</tr>
<tr>
<td>FATAL</td>
<td>LOG_ERR</td>
</tr>
<tr>
<td>TRACE</td>
<td>LOG_DEBUG</td>
</tr>
</tbody>
</table>

The syslog service must be properly configured for CIM, and the syslog daemon must be started as described in z/OS Communications Server: IP Configuration Reference and z/OS Communications Server: IP Configuration Guide.

Following is a sample syslog configuration file (/etc/syslog.conf) entry for the CIM server, which tells the syslog daemon to create log files:

```plaintext
Example:

... *
.*.CFZ*.debug /var/wbem/logs/cimserver_%Y.%m.%d.syslog
...
```

When configured like this, the CIM server log messages will be displayed in the format shown by the following example:
Example:

Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10025I: The CIM server is listening on HTTP port 5,988.
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10028I: The CIM server is listening on the local connection socket.
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10030I: Started CIM Server version 2.11
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ12533I: The CIM server failed to register with ARM using
element name CFZ_SRV_PEG2: return code 0x0C, reason code 0x0168.
Nov  7 12:49:01 BOECFZ1 CFZCIM[33557318]: CFZ10031I: CIM Server stopped.

Except for the CIM server's logLevel property, all configuration now occurs through
the syslog service as described in z/OS Communications Server: IP Configuration
Reference and z/OS Communications Server: IP Configuration Guide.

Configuration of the syslog daemon for specific processes/daemons is done based
on the job name of the process writing the logs.

When you run the CIM server as started task,
the job name is always CFZCIM.

When you have started the CIM server from the UNIX System Services command
prompt,
the job name of the CIM server is the user ID that started the CIM server.
Be sure that you have set environment variable _BPX_JOBNAME to CFZCIM
in order to set the job name of the CIM server correctly. Otherwise it will
be difficult to create a syslog configuration for the CIM server.

Audit logging with SMF record 86

The CIM server can file audit log records to SMF record 86. These records contain
information about authentication, configuration, provider status, and CIM
operations. For details of SMF record 86, see z/OS MVS System Management
Facilities (SMF).

To enable writing audit SMF record 86, modify the SMF, the CIM server, and the
security configuration:

SMF configuration:

__ Ensure that record 86 is part of your active SMF configuration
SMFPRMXX PARMLIB member.

CIM server configuration:

__ To enable the CIM server to write audit records, set the configuration
property enableAuditLog to true.

When recording is switched on, the current CIM server configuration and
the status of the currently loaded providers is recorded. To disable
recording, set the configuration property to false. This property can be
changed dynamically during CIM server runtime.

Security configuration:

__ In order to write SMF records, the CIM server needs at least READ
access to the BPX.SMF profile of the FACILITY class at your SAF
product.
Example for RACF:

```
RDEFINE FACILITY BPX.SMF UACC(NONE)
PERMIT BPX.SMF CL(FACILITY) ACCESS(READ) ID(CFZSRV)
SETROPTS RACLIST(FACILITY) REFRESH
```

If the CIM server audit logging is enabled, but SMF does not collect SMF record 86 or subtypes, or SMF is not enabled at all, no records are written.

### Backing up the CIM server configuration

After you have set up and configured the z/OS CIM server, you should back up the following CIM server property configuration files located in `/etc/wbem`:

- **cimserver_planned.conf**
  containing planned values which have been modified but are not yet in effect. They will be picked up at the next CIM server restart.

- **cimserver.env**
  containing the environment variables for the started task CFZCIM

How to backup the CIM server repository is described in “Backing up the CIM server repository” on page 73.

### Automatically restarting the CIM server

Since the CIM server serves as a primary system management interface for a system, it should be continuously available.

To support the CIM server availability, startup and shutdown messages are logged to the z/OS console to be used with a systems management program such as IBM Tivoli® System Automation.

The z/OS CIM server is enabled for the Automatic Restart Manager (ARM). The CIM server needs no additional configuration to use ARM, it always registers itself to ARM. When ARM is active and the CIM server is authorized to register with ARM, then success message CFZ12532I is displayed in the system log. Otherwise, information message CFZ12533I is displayed in the system log to inform you that the CIM server is not registered to ARM.

You can use ARM only for started task procedures or batch jobs. So if you start the CIM server from the UNIX System Services shell, you also get the message CFZ12533I. If you do not plan to use ARM, you can ignore this message, which is issued every time when the CIM server is started.

The CIM server issues the registration and the ready request after a successful bind to the communication socket/s (HTTP, HTTPS, and/or Local). It is deregistered from ARM during its normal shutdown procedure. In all other cases, the CIM server remains registered and is restarted based on the active ARM policy.

In a sysplex, you can start only one CIM server per OS image. Therefore ARM can only be used to restart after an application ABEND and not for cross-system restarts. You must use other facilities to start the CIM server during an IPL.

More information:
ARM policy considerations

The CIM server has the following requirements for exploiting the ARM restart policy:

- The ARM element name used for the CIM server is CFZ_SRV_<system_name>, where <system_name> is substituted by the value of the system symbol SYSNAME.
- The CIM server can only be restarted on the system where it failed. A cross-system restart within a sysplex is not possible. Therefore the termination type has to be ELEMTERM.
- The restart occurs through starting the CIM server started task procedure CFZCIM.

The sample JCL CFZARMP is installed to the SYS1.SAMPLIB during SMP/E z/OS installation of the CIM component.
SAMPLE JCL TO UPDATE THE ADMINISTRATIVE POLICY DATA FOR CIM
SERVER IN THE COUPLE DATA SET FOR ARM (AUTOMATIC RESTART MANAGER)

NOTES:

1. SYSPRINT DD IS A REQUIRED DD STATEMENT FOR THE UTILITY OUTPUT.
2. SYSIN DD IS A REQUIRED DD STATEMENT FOR THE UTILITY CONTROL STATEMENTS.
3. DATA TYPE(ARM) STATEMENT IS REQUIRED TO SPECIFY WHAT TYPE OF COUPLE DATA SET IS TO BE UPDATED.
4. REPORT KEYWORD IS OPTIONAL. WHEN REPORT(YES) IS SPECIFIED, AN ARM ADMINISTRATIVE POLICY REPORT WILL BE GENERATED IN THE OUTPUT. THE DEFAULT VALUE FOR REPORT IS YES.
5. REPLACE KEYWORD IS OPTIONAL. WHEN REPLACE(YES) IS SPECIFIED FOR A POLICY, THE POLICY WILL BE REPLACED IF IT ALREADY EXISTED IN THE COUPLE DATA SET. IF REPLACE(NO) IS SPECIFIED FOR AN EXISTING POLICY, THE UPDATE JOB WILL BE FAILED AND NO CHANGES WILL BE MADE TO THE COUPLE DATA SET.
6. TO DELETE AN EXISTING POLICY IN A COUPLE DATA SET, INCLUDE THE FOLLOWING LINE IN THE SYSIN DD CARD:
DELETE POLICY NAME(CFZARMPO)
WHERE POLNAME IS THE NAME OF THE POLICY TO BE DELETED.

STEP1 EXEC PGM=IXCMIAPU
STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
SYSPRINT DD SYSOUT=A
SYSABEND DD SYSOUT=A
SYSIN DD *

DATA TYPE(ARM)
REPORT(YES)

DEFINE POLICY NAME(CFZARMPO) REPLACE(YES)

RESTART_GROUP(CFZCIMRESGRP)
/* List all systems where the CIM Server can be started */
TARGET_SYSTEM(SYS1)
/* Wait 10 sec before restarting to free resources */
RESTART_PACING(10)

ELEMENT(CFZ_SRV *)
RESTART_ATTEMPTS(3,300)
RESTART_TIMEOUT(300)
READY_TIMEOUT(300)
/* coss-system restart is not allowed. */
/* No restart after system failure */

TERMTYPE(ELEMTERM)
RESTART_METHOD(ELEMTERM,STC,'S CFZCIM')
Backing up the CIM server repository

The CIM server keeps definitions of the data about managed objects and their providers in its repository located in /var/wbem.

It is important to schedule backups of the repository directories and files. If the repository is deleted or corrupted, backups of the repository files need to be restored. If the repository files cannot be restored from a backup, refer to section “Migration from z/OS 1.11 or z/OS 1.12 to z/OS 1.13” on page 18 for information about how to recover the repository.

As recommended in the z/OS Program Directory, the path /var/wbem should be mounted as a separate data set to simplify backing up. It is also recommended to stop the CIM server during backup to avoid data corruption.

Note: If the repository was backed up from a prior z/OS release, it should not be restored onto a system that runs a later version of z/OS. Once a new version of z/OS was installed and the CIM server has been initially started, you should immediately back up the upgraded repository and discard old repository backups.
Chapter 12. CIM server command-line utilities and console commands

The CIM server includes a set of command-line utilities and console commands that you can use to control or change the CIM server environment or to send CIM requests to CIM servers on z/OS or non-z/OS systems. You run most of the command-line utilities from a z/OS UNIX System Services shell.

Prepare the UNIX System Services shell as follows:

- Be sure that your environment is set up as described in “Step 5: Customizing the UNIX System Services shell” on page 21 or “Customizing the UNIX System Services shell” on page 50
- Grant system administrators using the command-line utilities CONTROL access to profile CIMSERV in class WBEM

CIM server utilities and commands:

**cimmof**

These commands are used to compile provider registrations and to compile CIM class descriptions written in the managed object format (MOF) language. The compiled information is put into the class schema stored in the repository.

The cimmof command is described in “cimmof” on page 77.

**cimconfig**

This command configures the options for the CIM server. Depending on the property being configured, the CIM server may need to be restarted after using this command.

The cimconfig command is described in “cimconfig” on page 79.

**cimprovider**

This command can be used to control the registered providers. The CIM server must be running to use this command.

The cimprovider command is described in “cimprovider” on page 81.

**cimcli**

This command lets you perform CIM client requests/operations against the local or remote CIM servers. It implements most of the DMTF CIM operations.

Each call of cimcli invokes a CIM operation with the corresponding parameters equivalent to the CIM operations defined in the CIM Operations over HTTP specification. Additionally, the cimcli command-line interface implements a number of other specific operations that support testing and querying CIM servers, including operations to query for namespaces and to get all instances in a namespace.

The cimcli command is described in “cimcli” on page 84.

**cimsub**

This command lets you manage CIM indications on the local CIM server. The command can list, enable, disable and remove indication subscriptions, filters and handlers.

The cimsub command is described in “cimsub” on page 110.
**MODIFY console command**

Like the cimconfig command, the MODIFY console command configures the options for the CIM server while the CIM server is running. Depending on the property being configured, the CIM server may need to be restarted after using this command.

The **MODIFY console command** is described in [“MODIFY console command” on page 113](#).

**Note:** The **wbemexec** utility is also included with CIM. It is used to directly send CIM-XML requests to a CIM server. However, this tool is not supported, but just supplied on an 'as-is-base'.

You can specify most options provided by the utilities in two ways:
- a short form introduced by a single dash, for example `-f<file>`
- a long form introduced by a double dash, for example `--file=<file>`

To understand the syntax of the description of the command-line utilities, see [“How to read syntax diagrams” on page 347](#).
Purpose

These commands are used to compile provider registrations or to compile CIM class descriptions written in the MOF language and store the information in the repository. For cimmof, the CIM server must be started before using this command.

The CIM server MOF compiler is a command-line utility that compiles MOF files (using the MOF format defined by the DMTF CIM Specification) into a CIM server repository. It allows compiling from structures of MOF files using the include #pragma and can either compile into a CIM server repository or check the syntax of the MOF files. The compiler requires that the input MOF files are in the current directory or that a fully qualified path is given. MOF files that are included using the include #pragma must be in the current directory or in a directory specified by a -I command-line switch.

For using the cimmof command against the CIM server namespaces (root/PG_Internal, root/PG_InterOp), a user needs to have CONTROL access to profile CIMSERV in class WBEM.

Syntax

Main diagram:

Options:

mof_file

Specifies the MOF file or MOF files to compile.

--version

Displays the CIM server version.
-h, --help, or no specified option
  Prints out a usage message with command definitions.

-I path  Specifies a path to the included MOF files.

-n path, --namespace=path
  Overrides the default CIM repository namespace path. The default is root/cimv2.

--xml
  Generates XML to standard output. This option does not update the repository.

--trace, --trace=filename
  Writes trace information to a file. If filename is omitted, the output destination is standard output. Those files are written with ASCII encoding.

-E
  Performs a syntax check on the input. This option does not update the repository.

-w
  Suppresses warning messages.

-uc
  Allows the update of an existing class definition. This option lets you update a leaf class. It does not allow updates of superclasses or classes that have subclasses.

-aE
  Allows the addition or modification of classes with the experimental qualifier.

-aV
  Updates a class that results in a version change. The version must be specified in a valid format. The format is m.n.u where m is major version, n is minor release and u is update. For example, 2.7.0 is a valid format for CIM Schema 2.7.0. If the input class has the same version as the class in the repository, the class is not updated.

-aEV
  Allows both Experimental and Version Schema changes.

Examples

cimmof -w -I./myDir myDir/CIM_Schema211.mof
  In this example, the managed object format (MOF) file that is located in directory myDir with the name CIM_Schema211.mof is compiled into the default namespace root/cimv2. CIM_Schema211.mof includes #pragmas for other MOF files that are also in the myDir directory. Therefore an include (-I) option is required for the myDir directory. The -w option suppresses warning messages.
cimconfig

Purpose
Use the cimconfig command to manage CIM server configuration properties. You can get, set, unset, or list these properties. See Chapter 11, “CIM server administration,” on page 61 for more information.

You can use the cimconfig command to set the current or planned configuration properties of the CIM server.

Current configuration properties:
You can update the current configuration properties only while the CIM server is running. After a restart of the CIM server, these changes will be reset to the planned or default configuration values. For making permanent changes, you must change the planned configuration values.

Planned configuration properties:
Planned configuration properties can be modified even if the CIM server is stopped. If the planned configuration properties are changed when the CIM server is running, those changes do not take effect until the CIM server is restarted.

For using the cimconfig command, a user needs to have CONTROL access to profile CIMSERV in class WBEM.

Syntax

```
cimconfig -g property-name -c
-s property-name=value -c
-u property-name -c
--version
-h --help
```

Options
The cimconfig command recognizes the following options:

- **-g property-name, -g property-name -c**
  Gets the current value of the configuration property property-name. Returns an error when the CIM server is not running.

- **-g property-name -p**
  Gets the planned value of the configuration property property-name.

- **-g property-name -d**
  Gets the default value of the configuration property property-name. Returns an error when the CIM server is not running.

- **-s property-name=value, -s property-name=value -c**
  Sets the current configuration property property-name to the value value. Returns an error when the CIM server is not running or the specified property cannot be updated dynamically.
-s property-name=value -p
Sets the planned configuration property property-name to the value value.

-u property-name, -u property-name -c
Unsets the value of the current configuration property property-name to its default value. Returns an error when the CIM server is not running or the specified property cannot be updated dynamically.

-u property-name -p
Unsets the value of the planned configuration property property-name to its default value.

-l Lists the names of all configuration properties. Returns an error when the CIM server is not running.

-l-c Lists the name and value pairs of all current configuration properties. Returns an error when the CIM server is not running.

-l -p Lists the name and value pairs of all planned configuration properties.

--version Displays the CIM server version.

-h, --help, no options specified Displays the command help information.

Examples

cimconfig -s traceLevel=4

cimconfig -s traceComponents=XmlIO,Http
Sets the trace level to trace all information with high data detail in the XmlIO and Http components.

cimconfig -s logLevel=WARNING -p
Sets the logLevel configuration property to the value WARNING in the cimserver_planned.conf file.
Purpose
The cimprovider command lets you disable, enable, remove, and list registered CIM providers or CIM provider modules and the according module status. In addition, it allows you to define groups of provider modules to be run in the same provider agent process.

disable
When a CIM provider is disabled, the CIM server rejects any requests to the provider. When a provider module is disabled, any new requests to the providers that are contained in the specified provider module are rejected.

enable
When a CIM provider is enabled, the CIM server forwards requests to the provider. When a provider module is enabled, the providers that are contained in the provider module are ready to accept a new request.

remove
When a CIM provider is removed (unregistered), the CIM server will no longer have any information about the provider. When a CIM provider module is removed (unregistered), the CIM server will no longer have any information about any provider contained in the module. If you want to address requests to a provider after removal, the provider or provider module must be registered again (typically by loading its registration schema using the cimmof command).

list
You can list all registered provider modules and the according module status or all providers in the specified provider module.

group
Allows grouping of provider modules in a single provider agent process when running the CIM server in out-of-process mode, that is, configuration property forceProviderProcesses is true.

For using the cimprovider command, the CIM server must be running, and the user needs to have CONTROL access to profile CIMSERV in class WBEM.

Syntax
The cimprovider command recognizes the following options:
-d -m module-name
Disables the CIM provider module module-name. If the module is already disabled, an error message is returned.

-e -m module-name
Enables the CIM provider module module-name. If the module is already enabled or is currently being disabled, an error message is returned.

-g group-name -m module-name
Sets the CIM provider module group. To remove a provider module from grouping, specify an empty string. If the provider module is active, it will be disabled before the group is set and then enabled again. All provider modules with the same group name are loaded into a single agent address space. If CIMServer is specified as group name, the provider module is loaded into the CIM server address space. Provider module groups are only in effect when running the CIM server in out-of-process mode.

-r -m module-name
Removes the provider module module-name and all of its contained providers.

-r -m module-name -p provider-name
Removes the provider provider-name in the provider module module-name without affecting any other providers in that module.

-l
Displays all registered provider modules.

   To list all providers in all modules, type a cimprovider -l command, followed by cimprovider -l -m for each listed module.

-l -s
Lists the status of all registered provider modules.

-l -f
Lists the full status of all registered provider modules and their module group name.

-l -m module-name
Lists all registered providers in module module-name.

--version
Displays the CIM server version.

-h, --help, no option specified
Displays the command help information.

Limitations
This command disables, enables, or removes one CIM provider module or CIM provider at a time.

Examples

cimprovider -d -m myProviderModule
Disables provider module myProviderModule and all of its contained providers (placing them in a stopped state).

cimprovider -e -m myProviderModule
Enables provider module myProviderModule and all of its contained providers (placing them in an OK state).

cimprovider -r -m myProviderModule
Removes (unregisters) the myProviderModule provider module and all of its contained providers.

cimprovider -r -m myProviderModule -p MyProvider
Removes (unregisters) the MyProvider provider contained in the myProviderModule provider module.
cimprovider -l
   Lists the registered provider modules.

cimprovider -l -s
   Lists the registered provider modules and their status (such as OK, Stopping, Stopped).

cimprovider -l -m myProviderModule
   Lists the registered providers, which are in the myProviderModule provider module.

cimprovider -g myProviderGroup -m myProviderModule
   Adds provider module myProviderModule to the group myProviderGroup. Module myProviderModule will be processed in the same provider agent process as all other providers in the group myProviderGroup.
cimcli

Purpose

z/OS provides a command-line interface called cimcli through which you can perform CIM client requests/operations. It implements most of the DMTF CIM operations except for the modifyClass, modifyInstance and createClass operations.

Each execution of cimcli invokes a CIM operation with the corresponding parameters equivalent to the CIM operations defined in the CIM Operations over HTTP specification.

In addition to the basic CIM operations defined in this specification, the cimcli command-line interface implements a number of other specific operations that support testing and querying CIM servers, including operations to query for namespaces and to get all instances in a namespace.

The command-line client is invoked from the UNIX System Services shell.

Syntax

Main diagram:

```
cimcli  Operation
   -h
   -hc
   --help
   -ho
   --version
```

Operation:
Defines the operation to be performed. cimcli performs all of the DMTF CIM operations (for example, getClass) and a set of compound operations (for example, enumerateNamespaces).

There are two forms for each operation: a long form which is the full name of the operation (for example, getClass), and a short form, typically two characters (for example, gc for getClass).

Options

```
-h          Prints help usage message.
-hc         Prints CIM operation command list.
--help      Prints full help message with commands, options, and examples.
-ho         Prints list of options.
--version   Displays the software version.
```
cimcli a (associators)

**Purpose**
Enumerates the classes or instances linked (associated) to a CIM class or a CIM instance.

**Operation**
```
  /SM590000/SM590000
    a
      classname
```

**Common options**
```
  -ac  assocClass
  -i   
  -ic  
  -iq  
  -n   
  -p   
  -pl  
  -r   
  -rc  
  -rr  
```

For "Instance name", see “cimcli Instance name” on page 109.

**Options**
For special options and "Common options", see “cimcli Options” on page 107.

**Examples**
cimcli a IBMzOS_Process

**Results**
0  Successful execution of the operation
all values other than 0
  The execution on the operation returned an error.

For a given class, the list of associated classes is returned.

For a given instance name, the list of associated instances is returned.
cimcli an (associatorNames)

Purpose
Enumerates the class or instance names linked (associated) to a CIM class or a CIM instance.

Operation

\[
\text{an} \quad \text{classname} \quad \text{associatorNames} \quad \text{Instance name}
\]

For "Instance name", see "cimcli Instance name" on page 109.

Options
For special options and "Common options", see "cimcli Options" on page 107.

Examples

\[
cimcli an IBMzOS_Process
\]

Results

0  Successful execution of the operation

all values other than 0

The execution on the operation returned an error.

For a given class, the list of associated class names is returned.

For a given instance name, the list of associated instance names is returned.
cimcli ci (createInstance)

**Purpose**
Creates one instance of the specified class with the provided properties in the repository.

**Operation**

```
cimcli ci classname createInstance /SV040000 propertyName = value
```

- `classname` parameter defines the class for which the instance is to be created.
- The optional set of parameters defines the properties to be provided (see also "cimcli Instance name" on page 109). The command reads the specified class and inserts the properties. The command will be rejected if the class does not exist in the namespace.

Specify a `value` for a property name according to its type. Follow the syntax rules as specified in Common Information Model Specification, DSP0004, Version 2.3 by the DMTF. Note special syntax rules to define:

- the current date and time with the keyword `now` for values of the type `Datetime`
- an empty string with the property name followed by a `!` for values of the type `string`
- an NULL string with the property name followed by a `=` for values of the type `string`

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli ci CIM_Person Name=Michael Title=Engineer

Creates an instance of the class CIM_Person.

**Results**
The command returns the object path of the created instance if the call to the CIM server was performed. Otherwise it returns the exception received.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution of the operation</td>
</tr>
<tr>
<td>all values other than 0</td>
<td>The execution on the operation returned an error</td>
</tr>
</tbody>
</table>
cimcli dc (deleteClass)

**Purpose**
Deletes the CIM class specified by `classname`.

**Operation**
```
>> dc deleteClass classname [ -n path ] Common options <<<
```

**Options**
For special options and "Common options", see ["cimcli Options" on page 107](#).

**Examples**
cimcli dc CIM_Person

Deletes the class CIM_Person and all sub-classes when there are no instances.

**Results**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution of the operation</td>
</tr>
<tr>
<td>all values other than 0</td>
<td>The execution on the operation returned an error.</td>
</tr>
</tbody>
</table>
cimcli di (deleteInstance)

**Purpose**
Deletes the specified instance or interactively one instance from the specified class.

**Operation**
```
+---+  +---+      
| di| deleteInstance| Instance name |
      +---+      
        |  
     -n--path--| Common options |
```

For "Instance name", see "cimcli Instance name" on page 109.

**Usage**
If the instance name is specified, the operation is performed directly. If a class name is specified, the enumerateInstanceNames command is performed with the class name and the list of returned instance names is presented to the user to select one to delete. cimcli then performs deleteInstance with the selected instance name.

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli di CIM_Person

Interactively deletes an instance of class CIM_Person.

**Results**
0 Successful execution of the operation
all values other than 0 The execution on the operation returned an error.

There is no response if the instance was successfully deleted, or an exception returned if there were any errors.
cimcli dq (deleteQualifier)

Purpose
Deletes the CIM qualifier specified by qualifiername.

Operation
```
dq qualifiername deleteQualifier -n path
```

Common options
For special options and "Common options", see "cimcli Options" on page 107.

Examples
```
cimcli dq ASSOCIATION
```
Deletes the qualifier Association (generally not recommended).

Results
```
0 Successful execution of the operation
all values other than 0 The execution on the operation returned an error.
```
cimcli ec (enumerateClasses)

Purpose
Enumerates the classes starting at the level defined by classname.

Operation
```
ec
    enumerateClasses
    classnam
```

Common options
```
-n path
-di
-ic
-niq
```

Usage
If the class name is omitted, cimcli inserts an empty class name.

Options
- `di` enumerates all inherited classes
  If you do not specify this parameter, only the child classes are enumerated.

For all other special options and "Common options", see "cimcli Options" on page 107.

Examples
```
cimcli ec -n root/cimv2 -niq
```
Enumerates classes from the root of the root/cimv2 namespace.

Results
```
0  Successful execution of the operation
all values other than 0
The execution on the operation returned an error.
```
cimcli ei (enumerateInstances)

Purpose
Enumerates the instances of the specified CIM class.

Operation
```
>> ei <classname>
```

Options
For special options and "Common options", see "cimcli Options" on page 107.

Examples
```
cimcli ei CIM_ComputerSystem -di
```
Enumerates the instances of class CIM_ComputerSystem, listing properties of inherited classes (-di).

Results
- `0` Successful execution of the operation
- `all values other than 0` The execution on the operation returned an error.
cimcli eq (enumerateQualifiers)

**Purpose**
Enumerates all qualifiers in the specified or default namespace.

**Operation**
```
  eq
  enumerateQualifiers [-n path] Common options
```

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli eq

Enumerates qualifiers in the default root/cimv2 namespace.

**Results**
```
0  Successful execution of the operation
all values other than 0  The execution on the operation returned an error.
```
cimcli gc (getClass)

**Purpose**
Gets the class of `classname`.

**Operation**
```
SM590000/SM590000
gc classname
getClass
```

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli gc IBMzOS_Process

Gets the definition for class IBMzOS_Process.

**Results**
0  Successful execution of the operation

all values other than 0
The execution on the operation returned an error.
cimcli gi (getInstance)

**Purpose**
Displays the specified instance.

**Operation**
```
cimcli gi [classname] <getInstance> [Instance name]
```

**Common options**
- `-n path`
- `-iql`
- `-pl propertyNameList`

For "Instance name", see "cimcli Instance name" on page 109.

**Usage**
If the instance name is specified, the operation is performed directly. If a class name is specified, the enumerateInstanceNames command is performed with the class name and the list of returned instance names is presented to the user to select one to display. `cimcli` then performs getInstance with the selected instance name.

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli gi IBMzOS_UnixLocalFileSystem

Interactively returns a list of instances from class IBMzOS_UnixLocalFileSystem. The user can select one instance to be displayed.

**Results**
- `0` Successful execution of the operation
- `all values other than 0` The execution on the operation returned an error.

If an instance is specified, the operation displays the result from the CIM server.

If a class is specified, an enumerateInstanceNames CIM operation is performed, and if any instance names are returned, the result is presented for the user to select one of the instances to be displayed.

If there are no instances, the command returns an empty response.
cimcli gq (getQualifier)

Purpose
Gets the CIM qualifier specified by qualifiername.

Operation
```
gq qualifiername -n path
```

Options
For special options and "Common options", see "cimcli Options" on page 107.

Examples
cimcli gq Association

Gets the qualifiers in mof output format

Results

0 Successful execution of the operation

all values other than 0
The execution on the operation returned an error.
cimcli im (invokeMethod)

Purpose
Performs the extrinsic method *methodname* on the specified class or instance.

Operation
```
im classname methodname name = value
```

For "Instance name", see ["cimcli Instance name"] on page 109.

Usage
The parameters are supplied as *name=value* pairs. In the current version, all parameters are treated as strings.

Options
For special options and "Common options", see ["cimcli Options"] on page 107.

Examples
```
cimcli im 'IBMzOS_Test.handle="1"' TriggerIndication NumberOfIndications=3
```

Results

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution of the operation</td>
</tr>
<tr>
<td>all values other than 0</td>
<td>The execution on the operation returned an error.</td>
</tr>
</tbody>
</table>
cimcli mi (modifyInstance)

Purpose
Modifies the specified instance or creates a modified instance of the specified class by building the properties from a combination of the target class and the provided properties.

Operation
```
mi -i modifyInstance classname Instance name
```
```
propertyName = param
```
```
-n path
-pl propertyNameList
```

For "Instance name", see ["cimcli Instance name" on page 109].

Options
For special options and "Common options", see ["cimcli Options" on page 107].

Examples
```
cimcli mi CIM_xxxx name=abc size=zyx
```

Results
0 Successful execution of the operation
all values other than 0
The execution on the operation returned an error.
**cimcli nc (enumerateClassNames)**

**Purpose**
Enumerates sub class names of *classname* or all top level class names of a given namespace.

**Operation**
```
c enumerateClassNames classname
```

**Usage**
Note that on z/OS all class names are returned in lowercase due to a z/OS specific performance optimization. Use the getClass operation to receive the exact case of the class name.

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli nc -di

Enumerates all class names from the root/cimv2 namespace, including subclasses (-di).

**Results**
- **0** Successful execution of the operation
- **all values other than 0** The execution on the operation returned an error.
cimcli ni (enumerateInstanceNames)

**Purpose**
Enumerates all instances of the specified class.

**Operation**

```
ni classname enumerateInstanceNames -n path
```

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli ni CIM_Processor -n root/cimv2

**Results**

- **0**: Successful execution of the operation
- **all values other than 0**: The execution on the operation returned an error.
cimcli ns (enumerateNamespaces)

Purpose
Requests an enumeration of all the namespaces in the target CIM server. This command uses both the CIM_Namespace class and if that fails, the __Namespace class to determine the list of namespaces.

Operation

```
  ns enumerateNamespaces
  Common options
```

Options
For “Common options”, see [cimcli Options](#) on page 107.

Examples

```
cimcli ns
```

Results

```
0  Successful execution of the operation
all values other than 0
    The execution on the operation returned an error.
```
cimcli r (references)

**Purpose**
Enumerates the association classes or association instances linked to the specified CIM class or CIM instance.

**Operation**
```
cimcli r classname references Instance name
```

**Common options**
- `i`
- `ic`
- `iq`
- `n`-`path`
- `r`-`role`
- `rc`-`resultClass`
- `pl`-`propertyNameList`

For "Instance name", see "cimcli Instance name" on page 109.

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli r 'IBMzOS_ComputerSystem.
    CreationClassName="IBMzOS_ComputerSystem",
    Name="sys1"
'
cimcli r IBMzOS_OperatingSystem -rc CIM_OSProcess

**Results**
0 Successful execution of the operation

all values other than 0

The execution on the operation returned an error.

For a given class, the list of linked association classes is returned.

For a given instance name, the list of linked association class instances is returned.
cimcli rn (referenceNames)

Purpose
Enumerates the association class or instance names linked to the specified CIM class or CIM instance.

Operation
```
  rn  -i  -n  -rc  path  -r  -role  classname
```

For "Instance name", see “cimcli Instance name” on page 109.

Options
For special options and "Common options", see “cimcli Options” on page 107.

Examples
```
cimcli rn 'IBMzOS_ComputerSystem.
    CreationClassName="IBMzOS_ComputerSystem",
    Name="sys1"

cimcli rn IBMzOS_OperatingSystem -rc CIM_OSProcess
```

Results
```
0     Successful execution of the operation

all values other than 0
    The execution on the operation returned an error.
```

For a given class, the list of linked association class names is returned.

For a given instance name, the list of linked association instance names is returned.
cimcli sp (setProperty)

Purpose
Sets a single property on a named instance.

Operation

```
sp classname propertyName = value
```

Common options

For "Instance name", see "cimcli Instance name" on page 109.

Usage
If the instance name is specified, the operation is performed directly. If a class name is specified, the enumerateInstanceNames command is performed with the class name and the list of returned instance names is presented to the user to select one to set. cimcli then performs setProperty with the selected instance name.

Options
For special options and "Common options", see "cimcli Options" on page 107.

Examples

cimcli sp 'CIM_Person.Name="Michael"' HomePhone=123456789

Sets the HomePhone property to 123456789.

Results

```
0 Successful execution of the operation
```

all values other than 0

The execution on the operation returned an error.

There is no response at the command prompt when the property has been successfully set.
cimcli ti (testInstance)

**Purpose**
Tests an instance or a class for the equality of the specified properties.

**Operation**
```
  ti
  testInstance  Instance name
  propertyName = _value
```

**Usage**
If the instance name is specified, the operation is performed directly. If a class name is specified, the `enumerateInstanceNames` command is performed with the class name and the list of returned instance names is presented to the user to select one to test. `cimcli` then performs `testInstance` with the selected instance name.

**Options**
For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
cimcli ti TST_Person Name=Mike SSN=333 -n test/TestProvider

**Results**
0  Successful execution of the operation
```
all values other than 0
```
The execution on the operation returned an error.

Returns an error code if the given properties and values do not match.
cimcli xq (execQuery)

**Purpose**
Performs the execQuery CIM operation with the specified *query-expression*. Note that the use of the execQuery operation has been deprecated by the DMTF and it may be removed in a future version of the "Specification for CIM Operations over HTTP".

**Operation**
```
/SM590000/SM590000
xq query-expression
execQuery WQL
```

**Options**
- `query-expression` specifies a WQL query expression. WQL is the only supported query language for this command.

For special options and "Common options", see "cimcli Options" on page 107.

**Examples**
```
cimcli xq "select handle,name from CIM_process
    where handle = "1"" WQL
```

**Results**
- **0** Successful execution of the operation
- **all values other than 0** The execution on the operation returned an error.
**cimcli Options**

**Purpose**
Options are identified on the command line with the ‘-’ or ‘--’ notation. An option that is not used by a particular operation is ignored.

**Common options**

- `-count number` Expected number of objects returned, if the `-sum` option is set.
- `-d` Displays more detailed debug messages.
- `-delay number` Delay in seconds between connection and request. Default is 0.
- `-l location` Allows input of the host name for the CIM server and optionally the port (HostName:port). The default is localhost:5988. The port component is optional. The default is 5988.
- `-n path` Specifies the namespace for the operation. The default is root/cimv2.
- `-o outputformats` Specifies the output format. Valid values are: xml, mof, and table. Default is mof.
- `-p password` Allows the input of a password for the command's server authentication. The default is empty.
- `-r repeat` Sets the number of times to repeat the function. Zero means one time. Repeats the operation without disconnecting. Default is 0.
- `-sort` Sorts the output objects before they are displayed.
- `-sum` Presents only summary information, not the complete output. Generally this option presents counts of objects returned instead of the names or objects themselves.
- `-t time` Measures the time for operation and presentation of the results upon command completion.
- `-timeout sec` Sets the connection timeout in seconds. Default is 20.
- `-trace traceLevel` Sets the common components trace. Sets the trace level. 0 is off. Default is 0. Valid values are 0 to 5.
- `-u username` Allows the input of a user name for authentication. The default is empty.
-v Displays verbose data (including operation parameters).
-x Output objects in xml instead of mof format.

The cimcli command recognizes the following special options:
- **ac assocClass** Passes the assocClass parameter to applicable association operations. Default is to pass no assocClass parameter.
- **ar associationRoleName** Defines an association role for associator operations.
- **di** Specifies the deepInheritance parameter for selected commands. The default is ‘false’. This option has different meanings for different commands and is used only with the enumerate commands. For further information, refer to the CIM Operations over HTTP published by the DMTF.
- **i** Interactively asks the user to select instances. Used with associator and reference operations.
- **ic** Sets the CIM operation parameter classOrigin in the operation request to true. Only useful with option -o xml.
- **iq** Sets includeQualifiers = true.
- **lo** Passes localOnly=true to applicable operations.
- **nlo** When set, sets localOnly = 'false' on operations. Default is 'false'.
  Note that option localOnly has been deprecated by the DMTF for some operations and will completely be removed with the next major version of CIM.
- **niq** Sets includeQualifiers = 'false' on operations. Default is 'false'.
  Note that option includeQualifiers has been deprecated by the DMTF for some operations and will completely be removed with the next major version of CIM.
- **pl propertyNameList** Passes the propertyNameList parameter to applicable operations. Format is p1,p2,p3 (without spaces) or "" for an empty list. The default is to pass no propertyList parameter.
- **r role** Passes the role parameter to applicable association operations. Default is to pass no role parameter.
- **rc resultClass** Passes the resultClass parameter to applicable association operations. Default is to pass no resultClass parameter.
- **rr resultRole** Passes the resultRole parameter to applicable association operations. Default is to pass no resultRole parameter.
**cimcli Instance name**

**Instance name**

Format 1:

```
classname\'keyPropertyName\' = "value1"
```

For the getInstance operation, there is also an alternate way to specify an instance name:

Format 2:

```
classname\keyPropertyName\value2
```

**Usage**

*keyPropertyName*

- to specify an instance, all key properties of the class have to be listed
- Specifying a key property with a "="), but without a value assigns the NULL value to it.

*value2*

- Values separated by a ',' are only valid if you specify an array.

The new syntax listing the key properties separated by spaces now allows to specify array values.

**Examples**

Format 1: 'CIM_Person.CreationClassName="",Name="Mike"'

Format 2: CIM_Person CreationClassName= Name=Mike
cimsub

**Purpose**

The **cimsub** command lets you manage CIM indications on the local CIM server. The command can list, enable, disable and remove indication subscriptions, filters and handlers. However, you cannot modify or create a handler or a filter. The CIM indication must be created or modified by a CIM client program.

- **list** Lists all or selected indication subscriptions, filters, and handlers, and displays the requested information about the instance(s).
- **enable** Enables a specific subscription. Sets a subscription into the enabled state, and the CIM server starts to process it.
- **disable** Disables a specific subscription. Sets a subscription into the disabled state, and it is no longer processed by the CIM server.
- **remove** Removes a specific indication subscription, filter, and/or handler from the CIM server. The information is removed within the CIM server and can only be recreated by a client application. The administrator must take care that a filter or handler is not referenced by any other subscription. If this is the case, but the filter or handler is deleted anyway, this subscription will no longer work.

In order to use the **cimsub** command, the CIM server must be running on the local system and a user needs to have **CONTROL** access to profile CIMSERV in class WBEM.

**Syntax**

**Main diagram:**

```plaintext
  cimsub
     list
     enable
     disable
     remove
        --version
        --help
```

**list:**

```plaintext
  -l
  -v
  -n path
  Filter
  Handler
```

In order to use the **cimsub** command, the CIM server must be running on the local system and a user needs to have **CONTROL** access to profile CIMSERV in class WBEM.
enable:

```
- e  Filter  Handler
```

disable:

```
- d  Filter  Handler
```

remove:

```
- r  Filter  Handler
```

Filter:

```
- f  filtername
```

Handler:

```
- H  hclassname
```

Options

The `cimsub` command recognizes the following options:

- `-l` Lists all or selected
  indication subscriptions (-ls),
  filters (-lf),
  handlers (-lh).

Options -F and -H are superseding the -n namespace option, if -n is set together with either -F or -H.

- `-e` Sets the subscription state to enabled.

- `-d` Sets the subscription state to disabled.

- `-r` Removes a specific
  indication subscription (-rs),
  filter (-rf),
  handler (-rh),
  or all three together (-ra).

Options -F and -H are superseding the -n namespace option, if -n is set together with either -F or -H.
-v Displays verbose information (for example, subscription state, filter query, handler destination) for each listed instance.

-F [fnamespace:]filtername
Specifies the name of the filter instance used for the subscription operation. If the filter namespace [fnamespace:] is not specified, the operation is using the namespace of the subscription.

-H [hnamespace:]hclassname,handlename
Specifies the name of the handler instance used for the subscription operation. If the handler namespace [hnamespace:] is not specified, the operation is using the namespace of the subscription. If the handler class name [hclassname:] is not specified, the operation is using the CIM_ListenListenerDestinationCIMXML handler class name.

Note: Currently the only supported handler is an instance of the CIM_ListenListenerDestinationCIMXML class or subclass.

-n path Specifies the namespace for the operation. For the -l option, if no namespace is specified, instances in all namespaces are listed. For all other operations, if no namespace is specified, the cimsub command operates on instances of the root/PG_InterOp namespace.

Note: It is recommended not to use any other namespace for indications than root/PG_InterOp.

--help Displays the command help information.

--version Displays the CIM server version.

Examples

The following example lists all subscriptions in the namespace root/PG_InterOP in verbose mode:

cimsub -ls -v

Output:

Namespace: root/PG_InterOp
Filter: root/PG_InterOp:IndicationTest_indicationFilter
Handler: root/PG_InterOp:CIM_ListenerDestinationCIMXML.IndicationTest
Query: "SELECT * FROM TestIndication"
Destination: http://test.server.com/
SubscriptionState: Enabled

Disables the subscription specified by -F and -H, and displays the result in verbose mode.

```c

```
MODIFY console command

In addition to the cimconfig command-line utility (see “cimconfig” on page 79), starting with z/OS 1.10 the CIM server configuration can be changed from the z/OS system console using the MODIFY command. The general syntax for using the MODIFY command to pass information to a UNIX System Services Application is described in z/OS MVS System Commands.

Syntax

Following is the specific syntax for using the MODIFY command to pass configuration changes to the CIM server. Between the options, no spaces are allowed:

```
MODIFY jobname, APPL = CONFIG,
property = value, PLANNED
```

Options

Basically the CIM server accepts the same options for the MODIFY command as for the cimconfig utility.

**jobname**
The name of the job that runs the CIM server. When the CIM server is run as a started task, this will usually be CFZCIM.

**APPL=CONFIG**
This is the indicator for the CIM server that a configuration change was requested through the z/OS system console.

**property**
The name of the configuration property to be changed. For a complete list of CIM server configuration properties see Chapter 9, “CIM server configuration,” on page 53. Typically, the only current configuration properties that you can change dynamically are the shutdownTimeout property and the logging and tracing properties. Permanent changes require a CIM server restart. They are indicated using the PLANNED keyword at the end of the MODIFY command string.

**value**
The new value for the configuration property to be changed. For values that contain a comma or for case sensitive property values such as path names the value needs to be enclosed in single quotes ('). To reset a property to its default value, omit the value parameter.

**PLANNED**
Indicates that the configuration change should be made permanently. This means that the change will only become effective after a CIM server restart, and that the change will also persist further restarts until it is changed again. If PLANNED was not specified at the end of the command, the changes will only stay in effect until the next restart of the CIM server.

Examples

```
F CFZCIM,APPL=CONFIG,traceComponents=xmlio
```
F CFZCIM,APPL=CONFIG,traceLevel=4
   Turns on tracing of the CIM server's XML traffic.

F CFZCIM,APPL=CONFIG,enableRemotePrivilegedUserAccess=true,PLANNED
   Permanently enables superusers (UID=0) to issue requests against the CIM
   server from a remote system.
Part 4. Provider reference
Chapter 13. Profiles

A profile defines the CIM model and its behavior that represents a particular domain to be managed. The CIM model comprises CIM classes, associations, indications, properties, methods, and values to describe the domain and its characteristics.

SMI-S profiles

The Storage Management Initiative Specification (SMI-S) was developed by members of the Storage Networking Industry Association (SNIA) and defines an interface for the secure, extensible, and interoperable management of a distributed and heterogeneous storage system. The specification describes the information available to a WBEM client from an SMI-S compliant CIM WBEM server.

The SMI-S specifies standards-based profiles to manage storage networks. It builds on other standards such a CIM. The scope of SMI-S includes storage, storage virtualizers, fibre channel fabrics and IP connectivity, and host storage-specific CIM-based profiles.

The host storage portion of the specification defines profiles for the management of host-based storage devices.

CIM for z/OS supports the host-based storage profiles:

**Host Discovered Resources profile**

The Host Discovered Resources (HDR) profile defines the model for the storage devices presented to z/OS.

**SB Multipath Management profile**

The Host Discovered Resource profile defines the model of the logical relationship of a host driver path to a logical unit. The SB Multipath Management profile defines the asynchronous notification of changes applying to this relation, using CIM life cycle indications.

**Storage HBA profile**

The Storage Host-Bus-Adapter (HBA) profile represents the manageable elements of an HBA and optionally, the storage connected to it, including the HBA Hot Swap Events for HBA creation and deletion, using CIM life cycle indications.

For more information, refer to the SNIA, Storage Management Initiative Specification (SMI-S) website, *Storage Management Technical Specification, Part 6 Host Elements*.

**Host Discovered Resources profile**

The Host Discovered Resources profile allows a client application to discover

- the storage hardware resources (such as host adapters and storage devices, and including the connectivity and correlatable names) attached to a host system,
- the logical storage resources (such as special files that represent storage devices) available through the operating system, and
- the relationship between these hardware and logical resources.
Figure 5 shows a Host Discovered Resources instance diagram with the host portion consisting of a ComputerSystem and an Initiator SBProtocolEndpoint and the storage controller portion consisting of a Target SBProtocolEndpoint and a LogicalDisk.

**Used CIM elements**

<table>
<thead>
<tr>
<th>Element name</th>
<th>implementing z/OS class</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM_ComputerSystem</td>
<td>IBMzOS_ComputerSystem</td>
<td>see page 129</td>
</tr>
<tr>
<td>CIM_LogicalDisk</td>
<td>IBMzOS_LogicalDisk</td>
<td>see page 148</td>
</tr>
<tr>
<td>CIM_StorageExtent</td>
<td>IBMzOS_LogicalDisk</td>
<td>see page 148</td>
</tr>
<tr>
<td>CIM_SystemDevice</td>
<td>IBMzOS_CSFCPortController</td>
<td>see page 251</td>
</tr>
<tr>
<td>CIM_ProtocolEndpoint</td>
<td>IBMzOS_SBProtocolEndpoint</td>
<td>see page 244</td>
</tr>
<tr>
<td>Association CIM_InitiatorTargetLogicalUnitPath</td>
<td>Association IBMzOS_SBI initiator TargetLogicalUnitPath</td>
<td>see page 258</td>
</tr>
</tbody>
</table>

**SB Multipath Management profile**

The SB Multipath Management is a subprofile of the Host Discovered Resource profile. This profile provides the asynchronous notification of the creation, state change and deletion of paths between devices and control units. The asynchronous notification is implemented as CIM life cycle indication (CIM_InstCreation, CIM_InstModification, CIM_InstDeletion) for a CIM_InitiatorTargetLogicalUnitPath.
Storage HBA profile

The storage Host-Bus-Adapter (HBA) profile represents the manageable elements of an HBA and optionally, the storage connected to it. An HBA can be connected to disks contained within a server's internal drive cage or an external drive enclosure or array.

Figure 6 shows an HBA instance diagram with the FC Initiator Port Subprofile consisting of an SBProtocolEndpoint and FCPortStatistics, providing data and implementation for FCPort.

HBA Hot Swap Events

The CIM server on z/OS implements the HBA Hot Swap Events for the Storage HBA profile using CIM life cycle indications. The notifications indicate the dynamic insertion (CIM_InstCreation) and deletion (CIM_InstDeletion) of an HBA represented by a CIM_PortController (representing a FICON channel port).
## Used CIM elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>implementing z/OS class</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM_FCPort</td>
<td>IBMzOS_FCPort</td>
<td>see page 236</td>
</tr>
<tr>
<td>CIM_FCPortStatistics</td>
<td>IBMzOS_FCPortStatistics</td>
<td>see page 239</td>
</tr>
<tr>
<td>CIM_PortController</td>
<td>IBMzOS_PortController</td>
<td>see page 241</td>
</tr>
<tr>
<td>CIM_Product</td>
<td>IBMzOS_Product</td>
<td>see page 243</td>
</tr>
<tr>
<td>CIM_SoftwareIdentity</td>
<td>IBMzOS_SoftwareIdentity</td>
<td>see page 246</td>
</tr>
<tr>
<td>CIM_SystemDevice</td>
<td>IBMzOS_CSFCPortController</td>
<td>see page 251</td>
</tr>
<tr>
<td>CIM_ProtocolEndpoint</td>
<td>IBMzOS_SBProtocolEndpoint</td>
<td>see page 244</td>
</tr>
<tr>
<td>Association CIM_ControlledBy</td>
<td>Association IBMzOS_ControlledBy</td>
<td>see page 249</td>
</tr>
<tr>
<td>Association CIM_DeviceSAPImplementation</td>
<td>Association IBMzOS_SBDeviceSAPImplementation</td>
<td>see page 250</td>
</tr>
<tr>
<td>Association CIM_ElementSoftwareIdentity</td>
<td>Association IBMzOS_ElementSoftwareIdentity</td>
<td>see page 252</td>
</tr>
<tr>
<td>Association CIM_ElementStatisticalData</td>
<td>Association IBMzOS_FCPortStatisticalData</td>
<td>see page 253</td>
</tr>
<tr>
<td>Association CIM_HostedAccessPoint</td>
<td>Association IBMzOS_SBHostedAccessPoint</td>
<td>see page 257</td>
</tr>
<tr>
<td>Association CIM_InstalledSoftwareIdentity</td>
<td>Association IBMzOS_InstalledSoftwareIdentity</td>
<td>see page 254</td>
</tr>
<tr>
<td>Association CIM_ProductElementComponent</td>
<td>Association IBMzOS_ProductElementComponent</td>
<td>see page 255</td>
</tr>
<tr>
<td>Association CIM_InitiatorTargetLogicalUnitPath</td>
<td>Association IBMzOS_SBInitiatorTargetLogicalUnitPath</td>
<td>see page 258</td>
</tr>
</tbody>
</table>
Chapter 14. z/OS Management Instrumentation for CIM

The CIM standard provides the ability to develop management applications that work with systems management data. To work with CIM, developers should have a thorough understanding of the CIM standard defined by the DMTF. For more information about the CIM standard, see Common Information Model (CIM) Standards on the DMTF website.

IBM has developed providers for z/OS that support basic operating system information and some performance metrics. A CIM provider is the link between the CIM server and the system (see Figure 2 on page 5). This interface allows CIM to access and manage the resources. Each CIM provider makes accessible the resources it represents in a standard way.

Notes:

1. IBM only supports the classes and properties listed in the present document or in other z/OS documentation provided by IBM. All other classes or properties which are not documented by IBM, IBM does not support, and bears no responsibility for their use.

2. Not all properties of the supported CIM classes described in this document are implemented by z/OS. Those properties implemented by z/OS are documented in each of the following subchapters. For all CIM properties not implemented by z/OS, the CIM server returns no values.

The following CIM classes and associations are implemented as IBM-supplied providers to provide basic operating system information:

Base classes
(See page 126)
- IBMzOS_ComputerSystem: subclass of CIM_ComputerSystem
- IBMzOS_OperatingSystem: subclass of CIM_OperatingSystem
- IBMzOS_OSProcess: subclass of association CIM_OSProcess
- IBMzOS_Process: subclass of CIM_Process
- IBMzOS_RunningOS: subclass of association CIM_RunningOS
- IBMzOS_UnixProcess: subclass of CIM_UnixProcess
- IBMzOS_LogicalDisk: subclass of CIM_LogicalDisk
- IBMzOS_LogicalDiskDevice: subclass of association CIM_SystemDevice

BaseBoard classes
(See page 138)
- IBM_BaseBoard: subclass of CIM_Card
- IBMzOS_BaseBoard: subclass of IBM_BaseBoard

Processor classes
(See page 142)
- IBMzOS_CSPProcessor: subclass of association CIM_SystemDevice
- IBMzOS_Processor: subclass of CIM_Processor

File System classes
(See page 156)
- IBMzOS_HostedFileSystem: subclass of association CIM_HostedFileSystem
• IBMzOS_NFS: subclass of CIM_NFS
• IBMzOS_UnixLocalFileSystem: subclass of CIM_UnixLocalFileSystem

Network classes
(See page 154)
• IBMzOS_EthernetPort: subclass of CIM_EthernetPort
• IBMzOS_CSNetworkPort: subclass of association CIM_SystemDevice
• IBMzOS_IPProtocolEndpoint: subclass of CIM_IPProtocolEndpoint
• IBMzOS_NetworkPortImplementsIP Endpoint: subclass of association CIM_PortImplementsEndpoint

Job classes
(See page 160)
• IBMzOS_Job: subclass of CIM_Job
• IBMzOS_JES2Job: subclass of IBMzOS_Job
• IBMzOS_JES3Job: subclass of IBMzOS_Job
• IBMzOS_SysoutDataset: subclass of CIM_LogicalFile
• IBMzOS_JES2SysoutDataset: subclass of IBMzOS_SysoutDataset
• IBMzOS_JES3SysoutDataset: subclass of IBMzOS_SysoutDataset
• IBMzOS_Subsystem: subclass of CIM_Service
• IBMzOS_JobsManagementSettings: subclass of CIM_SettingData
• association IBMzOS_SubsystemJES2Jobs (between IBMzOS_Subsystem and IBMzOS_JES2Job)
• association IBMzOS_SubsystemJES3Jobs (between IBMzOS_Subsystem and IBMzOS_JES3Job)
• association IBMzOS_UsesJES3SysoutDatasets (between IBMzOS_JES3Job and IBMzOS_JES3SysoutDataset)
• association IBMzOS_UsesJES2SysoutDatasets (between IBMzOS_JES2Job and IBMzOS_JES2SysoutDataset)

Cluster classes
(See page 191)
• IBMzOS_Sysplex: subclass of IBMzOS_Cluster
• IBMzOS_SysplexNode: subclass of IBMzOS_ClusterNode
• IBMzOS_CouplingFacility: subclass of IBMzOS_ClusterAggregatedResource
• IBMzOS_CFStructure: subclass of IBMzOS_ClusterAggregatedResource
• IBMzOS_CFStructureConnector: subclass of IBMzOS_ClusterResource
• IBMzOS_ClusterResource
• IBMzOS_ClusterGlobalResource
• IBMzOS_ClusterAggregatedResource
• IBMzOS_Cluster
• IBMzOS_ClusterNode
• IBMzOS_CoupleDataset: subclass of CIM_LogicalFile
• IBMzOS_SysplexCoupleDataset: subclass of IBMzOS_CoupleDataset
• IBMzOS_CFRCoupleDataset: subclass of IBMzOS_CoupleDataset
• IBMzOS_CouplingFunction: subclass of IBMzOS_ClusterAggregatedResource
• IBMzOS_CFRCPolicy: subclass of IBMzOS_ClusterAggregatedResource
• association IBMzOS_CollectionOfSysplexNodes
• association IBMzOS_CollectionOfCFs
• association IBMzOS_HostedCFStructure
• association IBMzOS_HostedCFStrConnector
• association IBMzOS_CFStructureDependsOn
• association IBMzOS_UsesCFs
• association IBMzOS_UsesCouplingFunctions
• association IBMzOS_UsesSysplexCoupleDatasets
• association IBMzOS_UsesCFRMCoupleDatasets
• association IBMzOS_UsesCFRMPolicies

Cluster indications
• IBMzOS_SysplexInstCreation
• IBMzOS_SysplexInstModification
• IBMzOS_Sysplex_ReallocateInitiated
• IBMzOS_Sysplex_ReallocateCompleted
• IBMzOS_Sysplex_CFRM_CDS_Initialized
• IBMzOS_SysplexNodeInstCreation
• IBMzOS_SysplexNodeInstDeletion
• IBMzOS_SysplexNodeInstModification
• IBMzOS_CouplingFacilityInstCreation
• IBMzOS_CouplingFacilityInstDeletion
• IBMzOS_CouplingFacilityInstModification
• IBMzOS_CFStructureInstCreation
• IBMzOS_CFStructureInstDeletion
• IBMzOS_CFStructureInstModification
• IBMzOS_CFStrConnectorInstCreation
• IBMzOS_CFStrConnectorInstDeletion
• IBMzOS_CFStrConnectorInstModification
• IBMzOS_CollectionOfSysplexNodesInstCreation
• IBMzOS_CollectionOfSysplexNodesInstDeletion
• IBMzOS_CollectionOfCFsInstCreation
• IBMzOS_CollectionOfCFsInstDeletion
• IBMzOS_HostedCFStructureInstCreation
• IBMzOS_HostedCFStructureInstDeletion
• IBMzOS_HostedCFStrConnectorInstCreation
• IBMzOS_HostedCFStrConnectorInstDeletion
• IBMzOS_UsesCFInstCreation: subclass of CIM_InstCreation
• IBMzOS_UsesCFInstDeletion: subclass of CIM_InstDeletion

Storage management classes
(See page 233)
• CIM_ScorageExtent
• IBMzOS_FCPort
• IBMzOS_FCPortStatistics
• IBMzOS_PortController
• IBMzOS_Product
• IBMzOS_SBProtocolEndpoint
• IBMzOS_SoftwareIdentity
• association IBMzOS_ControlledBy
• association IBMzOS_CSFCPort
• association IBMzOS_CSFCPortController
• association IBMzOS_ElementSoftwareIdentity
• association IBMzOS_FCPortStatisticalData
• association IBMzOS_InstalledSoftwareIdentity
• association IBMzOS_ProductElementComponent
• association IBMzOS_SBDeviceSAPImplementation
• association IBMzOS_SBHostedAccessPoint
• association IBMzOS_SBInitiatorTargetLogicalUnitPath

Storage management indications

For CIM_PortController:
• CIM_InstCreation
• CIM_InstDeletion

For CIM_InitiatorTargetLogicalUnitPath:
• CIM_InstCreation
• CIM_InstDeletion
• CIM_InstModification

WLM classes
(See page 261)
• IBMzOS_WLM
• association IBMzOS_WLMOS (between IBMzOS_WLM and IBMzOS_ComputerSystem)

WLM indications
• IBMzOS_WLMPolicyActivationIndication

CIM classes implemented by RMF

Please note that for using the CIM providers implemented by RMF you need to have RMF installed and additional configuration is required (see “Setting up the CIM server for RMF monitoring” on page 37). For more information, see z/OS RMF Programmer’s Guide and z/OS RMF User’s Guide

• IBMzOS_BaseMetricValue
• IBMzOS_BaseMetricDefinition
• IBMzOS_MetricForME
• IBMzOS_MetricDefForME
• IBMzOS_MetricInstance
• IBMzOS_Channel
• IBMz_CEC
• IBMz_ComputerSystem
• IBMzOS_WLMServiceDefinition
• IBMzOS_WLMServiceClassPeriod

To exploit this functionality, RMF must be installed and running.
Notes:

1. The z/OS Communications Server provides documentation of these CIM classes. For details refer to Considerations for Common Information Model (CIM) providers in z/OS Communications Server: IP Configuration Guide.
2. For all classes, the properties that are common for eServer and the z/OS specific properties are documented in separate tables.
3. Starting with z/OS 1.9, the CIM server exploits the functionality of Common event adapter (CEA). CEA is a z/OS component that provides the ability to deliver z/OS events to C-language clients. A CEA address space is started automatically during initialization of every z/OS system. In order for the address space to start successfully, you must configure CEA to work with z/OS. Failure to do so will cause CEA to run in a minimum function mode. For details refer to z/OS Planning for Installation.
4. An extra security setup is needed for the Job and Cluster classes.

To understand the syntax of the graphics showing class structures, see "Legend for graphics showing class structures" on page 345.

Supported CIM operations

While the z/OS CIM server supports all of the CIM operations from the DMTF’s CIM Operations over HTTP specification, only a specific subset of operations is supported by the OS management CIM providers delivered with this release of z/OS.

The following operations are available for all OS management classes or for association classes.

Available for all OS management classes:

- EnumerateInstanceNames
- EnumerateInstances
- GetInstance

Additionally available for all association classes:

- Associators
- AssociatorNames
- References
- ReferenceNames
Figure 7. CIM Base classes extended by z/OS-specific classes (1)
Figure 7 on page 126 illustrates the relationship between the IBM extension classes, and the CIM Base classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis. The DMTF website provides a detailed description of the CIM base classes. The z/OS-specific classes are described in detail in the following chapters.

The MOF files that define these classes can be found in directory schemas/os_management relative to where the providers for z/OS have been installed. The default is /usr/lpp/wbem/provider.

CIM_ComputerSystem

Purpose
This class represents either virtual or physical computer systems in the sense of a container inside which an operating system may run. This is the central class of the OS Management data model and aggregates all other resource classes.

Inheritance
The z/OS specific subclass is IBMzOS_ComputerSystem (see "IBMzOS_ComputerSystem" on page 129).

Additional subclasses of CIM_ComputerSystem are implemented by RMF, namely IBMz_ComputerSystem (LPARs) and IBMz_CEC. Unless RMF is installed or the RMF CIM providers have been set up appropriately, no instances or errors for those classes will be reported, for example by an enumerateInstances operation against class CIM_ComputerSystem. Errors for the classes supported by RMF are only reported when a CIM operation is invoked directly against one of the specific subclasses like IBMz_ComputerSystem.

For further details on classes IBMz_ComputerSystem and IBMz_CEC, see the z/OS RMF Programmer’s Guide.

CIM_OperatingSystem

Purpose
This class represents a running operating system with its basic properties.

Inheritance
The z/OS specific subclass is IBMzOS_OperatingSystem (see "IBMzOS_OperatingSystem” on page 131).

CIM_OSProcess

Purpose
This class associates an operating system with the set of currently active address spaces and UNIX System Services processes.

Inheritance
The z/OS specific subclass is IBMzOS_OSProcess (see "IBMzOS_OSProcess” on page 133).
CIM_Process

Purpose
This class represents currently active processes on an operating system. For z/OS this is mapped to address spaces and UNIX System Services processes.

Inheritance
The z/OS specific subclasses are:
- IBMzOS_Process (for address spaces) (see "IBMzOS_Process” on page 134)
- IBMzOS_UnixProcess (for UNIX System Services processes) (see "IBMzOS_UnixProcess” on page 137)

CIM_RunningOS

Purpose
This class associates a computer system with the currently running operating system (see Figure 7 on page 126).

Inheritance
The z/OS specific subclass is IBMzOS_RunningOS (see “IBMzOS_RunningOS” on page 136).
IBMzOS_ComputerSystem

Purpose
This class provides basic computer system information such as computer name, and status information. A provider instruments this class so that it can be used by client applications to identify the managed system on which the provider is running (typically a server or an application).

Inheritance
CIM_ManagedElement
  + CIM_ManagedSystemElement
  + CIM_LogicalElement
  + CIM_EnabledLogicalElement
  + CIM_System
  + CIM_ComputerSystem
  + IBMzOS_ComputerSystem

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
  IBMzOS_ComputerSystemProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
  libcmpiOSBase_ComputerSystemProvider.so

Used by the following CIM profiles
- Host Discovered Resources Profile
- IBM OS management

Properties
The following properties are common for eServer:

string Caption
   Always set to IBM z/OS Computer System.

string Description
   Always set to This is an IBMzOS_ComputerSystem.

string ElementName
   Returns IBM:model

string Name [key]
   The fully qualified IP host name.

string CreationClassName [key]
   Always set to IBMzOS_ComputerSystem

string NameFormat
   Describes the format used to build the Name property. Always set to IP.

uint16 Dedicated[]
   Indicates whether this is a special purpose system. Always set to 0 (not dedicated).

string UUID
   The universally unique identifier of the server. For z/OS, no value is supplied for this property, but it is maintained for compatibility with the other IBM eServer platforms.

string HostingSystemName
   A name that identifies the underlying hosting
system in a virtualized environment. Returns Elementname + serialnumber.

**string HostingSystemNameFormat**

The name format used for HostingSystemName. Always returns Other.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>string LPARName</strong></td>
<td>Name of the zSeries® logical partition that makes up the computer system. If not running in LPAR mode, a blank string is returned here.</td>
</tr>
<tr>
<td><strong>string VMGuestID</strong></td>
<td>z/VM® user ID of the virtual machine, of which the current z/OS image is a guest. If z/OS is not running as a guest under z/VM, a blank string is returned here.</td>
</tr>
<tr>
<td><strong>string CPUID</strong></td>
<td>String containing the readable part of the serial number concatenated with the model number.</td>
</tr>
<tr>
<td><strong>string SerialNumber</strong></td>
<td>IBM allocated number used to identify the server on which this computer system is running.</td>
</tr>
<tr>
<td><strong>string MachineType</strong></td>
<td>Processor family of this z/OS server.</td>
</tr>
<tr>
<td><strong>string Model</strong></td>
<td>Model number of the server.</td>
</tr>
<tr>
<td><strong>string Manufacturer</strong></td>
<td>The name of the company that produced the server.</td>
</tr>
<tr>
<td><strong>uint16 LPARid</strong></td>
<td>Logical partition number. This number distinguishes the configuration from all other level-2 configurations provided by the same LPAR hypervisor.</td>
</tr>
<tr>
<td><strong>string Plant</strong></td>
<td>Plant of manufacturer for the CPU.</td>
</tr>
</tbody>
</table>
**IBMzOS_OperatingSystem**

**Purpose**
This class is for use by client applications to obtain basic properties of a running z/OS operating system.

**Inheritance**
CIM_OperatingSystem

← IBMzOS_OperatingSystem

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprov<em>ider</em> command line tool for the administration of CMPI providers is

IBMzOS_OperatingSystemProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiOSBase_OperatingSystemProvider.so

**Properties**
The following properties are common for eServer:

- **string Name [key]**: The name of the z/OS operating system.
- **uint16 OperationalStatus[]**: Overall system status.
- **uint16 OSType**: Always 68 ('z/OS').
- **string Version**: Version, release and modification of the operating system in the format of "VV.RR.MM". For example, for z/OS V1.7.0, this will return "01.07.00".
- **datetime LastBootUpTime**: Time when the operating system was IPLed.
- **datetime LocalDateTime**: Local time of the operating system
- **sint16 CurrentTimeZone**: Time zone for the operating system, offset in minutes from GMT.
- **uint32 NumberOfUsers**: The number of currently logged on TSO and UNIX System Services users.
- **uint32 NumberOfProcesses**: Total number of UNIX processes and active address spaces.
- **uint32 MaxNumberOfProcesses**: The maximum number of processes configured in MaxProcSys.
- **uint64 MaxProcessMemorySize**: The maximum number of KBytes of memory that can be allocated to a process (RLIMIT_AS).
- **uint64 TotalVirtualMemorySize**: Total number of KBytes of virtual memory available to the operating system.
- **uint64 FreeVirtualMemory**: Number of KBytes of virtual memory currently unused and available.
uint64 FreePhysicalMemory  Number of KBytes of physical memory currently unused and available.

uint64 TotalVisibleMemorySize  The total amount of physical memory (in KBytes) available to the operating system.

uint64 SizeStoredInPagingFiles  The total number of KBytes that can be stored in the operating system’s page data sets.

uint64 FreeSpaceInPagingFiles  The total number of KBytes currently free in the operating system’s page data sets.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

string LanguageEdition  eServer specific extension for the language version of the OS. For z/OS always returns ‘en-US’.

string CodeSet  eServer specific extension for the default OS code page. For z/OS this returns the code page for the CIM server process.

uint32 DefaultPageSize  eServer specific extension. The default size of pages used by the virtual memory management in units of bytes. Always 4096 for z/OS.

string SysplexName  The name of the z/OS Sysplex to which this operating system belongs.

string FMID  Function modification identifier of the z/OS operating system.

uint32 LastBootUpDuration  Indicates the time in seconds used to complete the IPL.

string IPLProfile[]  HMC profile from which the operating system was IPLed. IPLProfile contains 4 elements:

  ipaiodfu  IODF unit address
  ipaloads  LOADxx suffix
  ipapromt  Operator prompt flag
  ipanucid  Nucleus ID

string sequentialReleaseNumber  Release number of the operating system as an ever increasing number, e.g. 21.00 for z/OS 1.11.
IBMzOS_OSProcess

**Purpose**
This class provides a link between the operating system and process(es) running in the context of this operating system. Client applications can use this provider to give clients an understanding of the processes (jobs) running on the managed system within the context of its operating system.

**Inheritance**
CIM_OSProcess
IBMzOS_OSProcess

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_OSProcessProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS_OSProcessProvider.so
**IBMzOS_Process**

**Purpose**

This class provides basic process information such as process name, priority, and run-time state. Instances of class IBMzOS_Process are mapped to z/OS address spaces. Client applications can use this class to give clients an understanding of the processes (address spaces) running on the managed system within the context of their operating system.

**Note:** z/OS also provides the notion of a UNIX process through the UNIX System Services. In addition, those processes running under UNIX System Services are supported by the extra IBMzOS_UnixProcess class which is derived from class CIM_UnixProcess. When a client enumerates all instances of class CIM_Process, it gets the complete list of z/OS address spaces, as well as all processes running under UNIX System Services. However, if the client enumerates the instances of class IBMzOS_Process directly, it only gets the list of address spaces since class IBMzOS_UnixProcess is not derived from IBMzOS_Process but only from CIM_UnixProcess. Ideally, IBMzOS_UnixProcess should inherit from IBMzOS_Process, besides inheriting from CIM_UnixProcess, however, multiple inheritance is not the current standard in CIM version 2. For inheritance information of the mentioned classes refer to [Figure 7 on page 126](#).

**Inheritance**

CIM_Process

+ IBMzOS_Process

**Module name**

The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_ProcessProviderModule

**Provider library**

The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is

libcmpiOSBase_ProcessProvider.so

**Properties**

The following properties are common for eServer:

- **string Name**
  The name of the z/OS address space.

- **string Handle [key]**
  The decimal representation of the address space ID(ASID).

- **uint32 Priority**
  The address space’s dispatching priority.

- **datetime CreationDate**
  The time when the address space was created.

- **uint64 KernelModeTime**
  Not supported for z/OS.

- **uint64 UserModeTime**
  Not supported for z/OS.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes:

- **string ProcessOwner**
  The primary z/OS user ID under which an address space was started.
uint16 ProcessType

The type of address space. Possible values are: 0 (Other), 1 (TSO User), 2 (Started Task), 3 (Job), 4 (System Address Space), 5 (Initiator).
IBMzOS_RunningOS

**Purpose**
This class is for use by clients to find associations between a computer system and the operating system that is currently running on the computer system.

**Inheritance**
- CIM_OperatingSystem
  - IBMzOS_OperatingSystem

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

  IBMzOS_RunningOSProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

  libcmpiIBMzOS_RunningOSProvider.so
**IBMzOS_UnixProcess**

**Purpose**
This class provides basic information about z/OS processes running in the UNIX System Services subsystem. It supports all properties from CIM_Process plus a set of properties typical for UNIX processes.

**Inheritance**
Class IBMzOS_UnixProcess is not derived from IBMzOS_Process, and therefore no instances of IBMzOS_UnixProcess are returned when a client enumerates the instances of class IBMzOS_Process, rather than class CIM_Process.

```
  CIM_Process
  + IBMzOS_UnixProcess
```

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is

```
  IBMzOS_UnixProcessProviderModule
```

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

```
  libcmpiOSBase_UnixProcessProvider.so
```

**Properties**
The following properties are common for eServer:

- **string Name**
  The name of the z/OS UNIX process. This is usually the name of the executable that started the process.

- **string Handle [key]**
  The z/OS UNIX process ID.

- **uint32 Priority**
  The process priority.

- **uint16 ExecutionState**
  The process state (ready, blocked, suspended, stopped, and so on).

- **datetime CreationDate**
  The time when the process was started.

- **uint64 KernelModeTime**
  Not supported on z/OS.

- **uint64 UserModeTime**
  Not supported on z/OS.

- **string ParentProcessID**
  The parent process ID.

- **string ParentProcessID**
  The real user ID.

- **uint64 ProcessGroupID**
  The process group ID.

- **uint64 ProcessSessionID**
  The process session ID.

- **string ProcessTTY**
  The TTY currently associated with this process.

- **string ModulePath**
  The executing process's command path.

- **string Parameters[]**
  The operating system parameters provided to the executing process. These are the argv[] values.

Class IBMzOS_UnixProcess has no z/OS specific properties.
OS management BaseBoard classes

Figure 8 illustrates the relationship between the IBM extension classes, and the CIM BaseBoard classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis. The DMTF website provides a detailed description of the CIM BaseBoard classes. The z/OS-specific classes are described in detail in the following chapters.

---

CIM_ComputerSystem
(from System_SystemElements)

NameFormat: string
OtherIdentifyingInfo: string
IdentifyingDescriptions: string
Dedicated: uint16
OtherDedicatedDescriptions: string
ResetCapability: uint16
PowerManagementCapabilities: uint16

SetPowerState()

CIM_PhysicalPackage
(from Physical_Package)

Removable: boolean
Replaceable: boolean
HotSwappable: boolean
Height: real32
Depth: real32
Width: real32
Weight: real32

IsCompatible()

CIM_Card
(from Physical_Package)

HostingBoard: boolean
SlotLayout: string
RequiresDaughterBoard: boolean
SpecialRequirements: boolean
RequirementsDescription: string
OperatingVoltages: sint16

ConnectPower()

IBM_BaseBoard

UUID: string

IBMzOS_CSBaseBoard

MachineType: string
Family: uint16
NumberOfProcessors: uint32
MemorySize: uint32
Architecture: uint16
MaxClockSpeed: uint32
NumberAndTypeOfProcessors: uint32
LPARId: uint16
Plant: string

IBMzOS_CoCSystem

LPARName: string
VMName: string
Caption: string
Manufacturer: string
LPARid: uint16
Plant: string
MachineType: string
Model: string
SerialNumber: string

+Dependent
+Antecedent

IBMzOS_CSBaseBoard

---

Figure 8. OS management BaseBoard Class
IBM_BaseBoard

**Purpose**
This class represents the unique characteristics of the physical hardware as recognized by the z/OS operating system running on that hardware (the inband view). On most platforms these are the characteristics of the main board, and therefore, the name IBM_BaseBoard was chosen for this class. Instances of this class are either identified by a unique ID that was assigned to the main board (property *UUID*) or by the combination of manufacturer, model and serial number. The major purpose of this class is to provide the ability to determine which instances of computer systems are running on the same physical hardware.

**Inheritance**
The z/OS specific subclass is IBMzOS_BaseBoard (see “IBMzOS_BaseBoard” on page 140).

**Properties**
The following properties are common for eServer:

- **string Caption**
  Always returns ‘Base Board’.

- **string Description**
  Always returns ‘A class derived from Card to deliver the systems base board hardware information.’

- **string ElementName**
  Same as property *Tag*.

- **string Tag [key]**
  A combination of manufacturer, model and serial number in the following format: manufacturer:model:serialnumber.

- **string CreationClassName [key]**
  Always returns ‘IBMzOS_BaseBoard’.

- **string SerialNumber**
  IBM allocated number used to identify the CEC.

- **string Model**
  The model number of the CEC, for example ‘314’.

- **string Manufacturer**
  The name of the company that produced the CEC.

- **string PartNumber**
  Not supported for z/OS.

- **boolean HostingBoard**
  Always returns *TRUE*, indicating that this card is a main board.

- **string UUID**
  The unique ID assigned to the main board. For z/OS, no value is supplied for this property, but it is maintained for compatibility with the other IBM eServer platforms.
**IBMzOS_BaseBoard**

**Inheritance**
- IBM_BaseBoard
  - IBMzOS_BaseBoard

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is
- IBMzOS_BaseBoardProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- `libcmpiOSBase_BaseBoardProvider.so`

**Properties**
The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

- **string MachineType**
  Processor type for the class of this z/OS server, for example: 2084

- **uint16 Family**
  The processor family. For z/OS, a value of 204 (z/Architecture® base) is returned.

- **uint32 NumberOfProcessors**
  The number of general purpose processors installed on the system board.

- **uint32 MemorySize**
  The total amount of physical memory (in Kbytes) available to the operating system through which this data was provided. Note that this is not the total amount of installed memory for the zSeries CEC. This is the inband view of z/OS.

- **uint16 Architecture**
  The processor architecture.

- **uint32 NumberAndTypeOfProcessors[]**
  An array of uint32 where the first element is the number of general purpose processors, the second element is the number of zAAPs, the third element is the number of zIIPs, if supported.

- **uint16 LPARid**
  Logical partition number. This number distinguishes the configuration from all other level-2 configurations provided by the same LPAR hypervisor.

- **string Plant**
  Plant of manufacturer for the CPU.
Association CIM_ComputerSystemPackage

**Purpose**
This class associates a ComputerSystem with the physical main board of the system on which it runs.

**Inheritance**
The z/OS specific subclass is IBMzOS_CSBaseBoard (see "Association IBMzOS_CSBaseBoard").

Association IBMzOS_CSBaseBoard

**Purpose**
This class associates a z/OS computer system with the physical zSeries CEC on which it runs (see Figure 8 on page 138). It has no properties.

**Inheritance**
CIM_ComputerSystemPackage

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_CSBaseBoardProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_CSBaseBoardProvider.so
OS management Processor classes

Figure 9 illustrates the relationship between the IBM extension classes, and the CIM Processor classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis. The DMTF website provides a detailed description of the CIM Processor classes. The z/OS-specific classes are described in detail in the following chapters.
CIM_Processor

Purpose
This class represents the physical processors that are available to the operating system.

Inheritance
The z/OS specific subclass is IBMzOS_Processor (see "IBMzOS_Processor" on page 144).

Association CIM_SystemDevice

Purpose
This class associates a ComputerSystem with the instrumented processors.

Inheritance
The z/OS specific subclass is IBMzOS_CSProcessor.
**IBMzOS_Processor**

**Inheritance**

CIM_Processor

+ IBMzOS_Processor

**Module name**

The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_ProcessorProviderModule

**Provider library**

The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libIBMzOS_Processor.so

**Properties**

The following properties are common for eServer:

- **string Caption**
  Always set to ‘zSeries logical processor’.

- **string Description**
  Always set to ‘This class represents instances of processors currently available to the z/OS operating system’.

- **string ElementName**
  Same as DeviceID.

- **string DeviceID [key]**
  Concatenation of the CPUID of the physical processor (PCCACPID) + colon (‘:’) + CPU address. CBA987654321:2 is an example for a valid DeviceID.

  If a CPU is in Reserved or Offline state, the CPUID is FFFFFFFF.

- **uint16 EnabledState**
  2 Online
  3 Reserved
  6 Offline
  9 Offline by WLM

- **string Role**
  CP Central Processor (including zEAP Processors)
  ZIIP zIIP processor
  ZAAP zAAP processor
  UNKNOWN no assigned role

- **uint16 Family**
  200 (=‘S/390® and zSeries Family’).

- **string OtherFamilyDescription**
  ‘S/390 and zSeries Family’ or specific model like ‘z990’.

- **uint32 MaxClockSpeed**
  Not supported for z/OS.

- **uint32 CurrentClockSpeed**
  Not supported for z/OS.
uint16 LoadPercentage

For z/OS provided through RMF metrics provider only.

string Stepping

Not supported for z/OS.

string UniqueID

CPUID of the physical processor (PCCACPID).

uint16 CPUStatus

Not supported for z/OS.

Class IBMzOS_Processor has no z/OS specific properties.

Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unit32 RequestStateChange()</td>
<td>Issues messages for the operator or automation to change the state of the processor.</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16 RequestedState</td>
<td>Must be one of &quot;Enabled&quot; (2) or &quot;Offline&quot; (6).</td>
</tr>
<tr>
<td>CIM_ConcreteJob REFJob</td>
<td>Always returns NULL.</td>
</tr>
<tr>
<td>datetime TimeoutPeriod</td>
<td>Must be either not defined or a CIM NULL value.</td>
</tr>
</tbody>
</table>

Return values

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>4097</td>
</tr>
<tr>
<td>4098</td>
</tr>
</tbody>
</table>
**OS management Logical Disk classes**

Figure 10 on page 147 illustrates the relationship between the IBM extension classes, and the CIM Base classes that they extend. This figure focuses on class IBMzOS_LogicalDisk which was provided in z/OS 1.9 CIM server to support the management of logical disks.

The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis.

The [DMTF website](https://www.dmtf.org) provides a detailed description of the CIM Base classes. The z/OS-specific classes are described in detail in the following chapters.

**Note:** The described metrics are only available for active disks, but not for inactive or offline disks.
**CIM_LogicalDisk**

**Purpose**
This class represents logical disks attached to an operating system.

**Inheritance**
The z/OS specific subclass is IBMzOS_LogicalDisk (see "IBMzOS_LogicalDisk" on page 148).
IBMzOS_LogicalDisk

Purpose
This class provides basic information about disk devices known to the z/OS operating system based on the logical view.

Inheritance
CIM_ManagedElement
- CIM_ManagedSystemElement
- CIM_LogicalElement
- CIM_EnabledLogicalElement
- CIM_LogicalDevice
- CIM_StorageExtent
- CIM_LogicalDisk
- IBMzOS_LogicalDisk

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_LogicalDiskProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS_LogicalDiskProvider.so

Used by the following CIM profiles
- Host Discovered Resources profile

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string Caption</td>
<td>Returns z/OS Storage Volume.</td>
</tr>
<tr>
<td>string Description</td>
<td>Returns Represents a storage volume as seen by z/OS.</td>
</tr>
<tr>
<td>string ElementName</td>
<td>Same as Name (Volume Serial Number)</td>
</tr>
<tr>
<td>string Name</td>
<td>Unique identifier for the extent in the form CC:SS:DDDD, where</td>
</tr>
<tr>
<td></td>
<td>CC is the channel subsystem ID</td>
</tr>
<tr>
<td></td>
<td>SS is the SubchannelSetID</td>
</tr>
<tr>
<td></td>
<td>DDDD is the DeviceNumber</td>
</tr>
<tr>
<td>uint16 NameFormat</td>
<td>Returns 12 OS device name format</td>
</tr>
<tr>
<td>uint16 NameNamespace</td>
<td>Returns 8 OS device namespace</td>
</tr>
<tr>
<td>uint16 EnabledState</td>
<td>Mapped from the UCBONLI and UCBOX values retrieved through UCBSCAN.</td>
</tr>
<tr>
<td></td>
<td>See Table 7 on page 149 for mapping values of EnabledState to system data.</td>
</tr>
<tr>
<td>string CreationClassName</td>
<td>Always returns IBMzOS_LogicalDisk.</td>
</tr>
</tbody>
</table>
string DeviceID

Channel Device ID obtained from UCBCHAN through UCBSCAN

string[] IdentifyingDescriptions

[0]=Device Node Element Description obtained from the NEDID field of the matching IHACDR control block.

string[] OtherIdentifyingInfo

[0]=“Device Node Element Descriptor”

string SystemCreationClassName

Always returns IBMzOS_ComputerSystem.

string SystemName

The systems fully qualified hostname (see IBMzOS_ComputerSystem:colon;Name). Obtained through the OSBase_Common.get_system_name() function.

uint16 OperationalStatus[]

Returns

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>OK</td>
</tr>
<tr>
<td>9</td>
<td>Stopping</td>
</tr>
<tr>
<td>10</td>
<td>Stopped</td>
</tr>
</tbody>
</table>

The property enabledState is set based on the UCB control block information as shown in the following table:

### Table 7. UCB control block information

<table>
<thead>
<tr>
<th>UCBONLI</th>
<th>UCBBOX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Boxed</td>
</tr>
<tr>
<td>Online</td>
<td>Quiesce (9)</td>
</tr>
<tr>
<td>Offline</td>
<td>Disabled (3)</td>
</tr>
<tr>
<td>Pending Offline</td>
<td>Shuting down (4)</td>
</tr>
</tbody>
</table>

**Associations**

**IBMzOS_SBInitiatorTargetLogicalUnitPath**

Source: IBMzOS_LogicalDisk

Target: CIM_ProtocolEndpoint

see page 258

**IBMzOS_LogicalDiskDevice**

Source: IBMzOS_ComputerSystem

Target: IBMzOS_LogicalDisk
OS management File System classes

Figure 11 illustrates the relationship between the IBM extension classes, and the CIM FileSystem classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis. The DMTF website provides a detailed description of the CIM FileSystem classes. The z/OS-specific classes are described in detail in the following chapters.

Figure 11. OS management File System classes
CIM_LocalFileSystem

Purpose
This class represents file systems that are locally attached to a computer system. On z/OS, hierarchical file systems HFS and zFS are supported.

Inheritance
The z/OS specific subclass is IBMzOS_UnixLocalFileSystem (see “IBMzOS_UnixLocalFileSystem” on page 152).

CIM_RemoteFileSystem

Purpose
This class represents file systems that are accessed remotely by a computer system. On z/OS, only NFS is supported.

Inheritance
The z/OS specific subclass is IBMzOS_NFS (see “IBMzOS_NFS” on page 153).

Association CIM_HostedFileSystem

Purpose
The CIM_HostedFileSystem association associates a ComputerSystem with the set of currently mounted UNIX System Services file systems.

Inheritance
The z/OS specific subclass is IBMzOS_HostedFileSystem.
**IBMzOS_UnixLocalFileSystem**

**Inheritance**

CIM_LocalFileSystem

← IBMzOS_UnixLocalFileSystem

**Module name**

The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_UnixLocalFileSystemProviderModule

**Provider library**

The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libIBMzOS_UnixLocalFileSystem.so

**Properties**

The following properties are common for eServer:

- **string Caption**
  
  Always set to 'z/OS hierarchical local file system'.

- **string Description**
  
  Always set to 'This class represents instances of currently mounted local hierarchical file systems'.

- **string ElementName**
  
  Same as Name.

- **string Name [key]**
  
  File system name (z/OS data set name).

- **string Root**
  
  Name of the directory where the file system is mounted.

- **uint64 FileSystemSize**
  
  File system size in bytes.

- **uint64 AvailableSpace**
  
  Space available on the file system in bytes.

- **boolean ReadOnly**
  
  Indicates whether the file system is mounted read only.

- **string FileSystemType**
  
  File system type, for example ‘NFS’.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes:

- **DDName**
  
  DD name that was specified on mount.

- **FSParentDeviceID**
  
  Device ID of the parent file system.

- **FSDeviceID**
  
  Device number which the STAT command will return for all files in this file system.

- **MountParameters**
  
  The parameters that were specified for the mount command.

- **FSOwner**
  
  MVS Owner ID of the file system.

- **FSTypeName**
  
  The file system type name from the PARMLIB statement.
**IBMzOS_NFS**

**Inheritance**

```
CIM_RemoteFileSystem
← IBMzOS_NFS
```

**Module name**

The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is

```
IBMzOS_NFSProviderModule
```

**Provider library**

The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

```
libIBMzOS_NFS.so
```

**Properties**

The following properties are common for eServer:

- **string Caption**
  Always set to ‘z/OS mounted network file system’.

- **string Description**
  Always set to ‘This class represents instances of currently mounted network file systems’.

- **string ElementName**
  Same as Name.

- **string Name [key]**
  File system name (corresponds to the `file system argument of the mount command`).

- **string Root**
  Name of the directory where the file system is mounted.

- **uint64 FileSystemSize**
  File system size in bytes.

- **uint64 AvailableSpace**
  Space available on file system in bytes.

- **boolean ReadOnly**
  Indicates whether the file system is mounted read only.

- **string FileSystemType**
  File system type, for example ‘NFS’.

Class IBMzOS_NFS has no z/OS specific properties.
OS management Network classes

The classes described in this section are implemented by the z/OS Communication Server. For details on these CIM classes, refer to z/OS Communications Server: IP Configuration Guide.

The providers are installed in the /usr/lpp/tcpip/lib hierarchical file system directory and linked to the CIM server's provider directory.

The z/OS CS CIM class definition and provider registration files are installed in the /usr/lpp/tcpip/mof hierarchical file system directory and are already integrated into the CIM server.

Figure 12 on page 155 illustrates the relationship between the IBM extension classes, and the CIM Network classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis. The DMTF website provides a detailed description of the CIM BaseBoard classes. The z/OS-specific classes are described in detail in the following chapters.
Figure 12. OS management Network classes
CIM_EthernetPort

Purpose
This class represents network ports (interfaces) of type Ethernet. For z/OS, all the Ethernet interfaces configured to the TCP/IP stacks on the MVS image are supported.

Inheritance
The z/OS specific subclass is IBMzOS_EthernetPort (see “IBMzOS_EthernetPort” on page 157).

CIM_IPProtocolEndpoint

Purpose
This class represents the installed IP protocols. For z/OS, all IPv4 addresses configured to the TCP/IP stacks on the MVS image are supported.

Inheritance
The z/OS specific subclass is IBMzOS_IPProtocolEndPoint (see “IBMzOS_IPProtocolEndpoint” on page 158).

CIM_PortImplementsEndpoint

Purpose
This class associates a network port with its installed network protocols. Currently, only IP protocols defined for Ethernet ports are returned.

Inheritance
The z/OS specific subclass is IBMzOS_NetworkPortImplementsIPEndpoint.

Association CIM_SystemDevice

Purpose
This class associates a ComputerSystem with the instrumented network ethernet ports.

Inheritance
The z/OS specific subclass is IBMzOS_CSNetworkPort.
**IBMzOS_EthernetPort**

**Inheritance**
- CIM_EthernetPort
  - IBMzOS_EthernetPort

**Provider module**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_EthernetPortProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libcmpiOSBase_EthernetPortProvider.so

**Owning component**
The z/OS component which owns the CMPI provider is Communication Server

**Properties**
The following properties are common for eServer:

- **string Caption**
  - Always set to 'IBMzOS EthernetPort'.

- **string Description**
  - Variable, depending on the type of interface, for example, 'IP Assist Queued Direct I/O Ethernet protocol port'.

- **string ElementName**
  - Same as Name.

- **string Name**
  - The label by which the NetworkPort is known to the operating system ('tcpprocname_intfname').

- **uint16 EnabledState**
  - Indicates whether the protocol endpoint is active or not.

- **string DeviceID [key]**
  - Identifying information to uniquely name the ethernet port. ('tcpprocname_intfname').

- **uint64 Speed**
  - The current bandwidth of the port in bits per second.

- **uint64 MaxSpeed**
  - The maximum bandwidth of the port in bits per second. For z/OS, this is always the same value as Speed.

- **uint16 LinkTechnology**
  - Always 2 (=Ethernet).

- **string OtherLinkTechnology**
  - Not set for z/OS.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

- **TcpipProcName**
  - z/OS TCP/IP stack name.
IBMzOS_IPProtocolEndpoint

Inheritance
CIM_IPProtocolEndpoint
→ IBMzOS_IPProtocolEndpoint

Provider module
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_IPProtocolEndpointProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiOSBase_IPProtocolEndpointProvider.so

Owning component
The z/OS component which owns the CMPI provider is
Communication Server

Properties
The following properties are common for eServer:

string Caption
Always set to 'IBMzOS Protocol Endpoint for IP'.

string Description
Always set to 'A communication point to send and receive data. This class is dedicated to relate IP interfaces to Logical Networks'.

string ElementName
Same as Name.

string Name [key]
The unique name of the protocol endpoint, constructed according to the template in NameFormat.

uint16 EnabledState
Returns whether the protocol endpoint is active or not.

string NameFormat
Describes the format of the name property. For z/OS, this is always set to 'TCP|IPPROCNAME_TYPE_DEVICE_IPADDR(_ETH)'.

string IPv4Address
The IPv4 IP address.

string IPv6Address
Not yet supported for z/OS instrumentation.

string SubnetMask
The IPv4 IP subnet mask.

uint16 IPVersionSupport
Always returns 1 (=IPv4 only).

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.
TcpipProcName

z/OS TCP/IP stack name.
OS management Job classes

The classes described in this section are implemented by z/OS to instrument the z/OS jobs subsystems, JES2 and JES3.

For using these providers you need an extra security setup as described in "Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers" on page 38.

For a list of the Jobs providers' reason codes, see "Appendix C. CEA reason codes" on page 339.
**IBMzOS_JES2Job**

**Purpose**
This class is a subclass of IBMzOS_Job and contains those properties that are unique to a job that has run, or will run, under JES2.

**Inheritance**
IBMzOS_Job

← IBMzOS_JES2Job

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_JES2JobProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_JES2JobProvider.so

**Properties**
The following properties have been implemented for z/OS:

- **string Caption**
  A short description of the class. Returns ‘IBM z/OS JES2 Job’.

- **string Description**
  A description of the class. Returns ‘This is an IBMzOS_JES2Job’.

- **string ElementName**
  Name given to this instance of the class (same as Name)

- **datetime InstallDate**
  Not supported for z/OS.

- **string Name [key]**
  The property is overridden by IBMzOS_JES2Job. It contains a unique identifier for this job.

- **uint16 OperationalStatus[]**
  The current status of the JES2 job.
  1  No subchain exists
  2  Active in CI in FSS address space
  3  Awaiting postscan (batch)
  4  Awaiting postscan (damsel)
  5  Awaiting volume fetch
  6  Awaiting start setup (JES3), Awaiting setup (JES2)
  7  Awaiting/active in MDS system select processing
  8  Awaiting resource allocation
  9  Awaiting unavailable volumes
  10  Awaiting volume mounts
  11  Awaiting/active in MDS system verify processing
  12  Error during MDS processing
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Awaiting selection on main (JES3), Awaiting execution (JES2)</td>
</tr>
<tr>
<td>14</td>
<td>Scheduled on main (JES3), Active executing (JES2)</td>
</tr>
<tr>
<td>17</td>
<td>Awaiting breakdown (JES3), Active in output (JES2)</td>
</tr>
<tr>
<td>18</td>
<td>Awaiting MDS restart processing</td>
</tr>
<tr>
<td>19</td>
<td>Main MDS processing complete</td>
</tr>
<tr>
<td>20</td>
<td>Awaiting output service (JES3), Awaiting hardcopy (JES2)</td>
</tr>
<tr>
<td>21</td>
<td>Awaiting output service writer</td>
</tr>
<tr>
<td>22</td>
<td>Awaiting reserved services</td>
</tr>
<tr>
<td>23</td>
<td>Output service complete</td>
</tr>
<tr>
<td>24</td>
<td>Awaiting selection on main (demand select job)</td>
</tr>
<tr>
<td>25</td>
<td>Ending function rq waiting or I/O completion</td>
</tr>
<tr>
<td>26</td>
<td>Ending function rq not processed</td>
</tr>
<tr>
<td>27</td>
<td>Maximum rq index value</td>
</tr>
<tr>
<td>128</td>
<td>Active in input processing</td>
</tr>
<tr>
<td>129</td>
<td>Waiting conversion</td>
</tr>
<tr>
<td>130</td>
<td>Active in conversion</td>
</tr>
<tr>
<td>131</td>
<td>Active in setup</td>
</tr>
<tr>
<td>132</td>
<td>Active in spin</td>
</tr>
<tr>
<td>133</td>
<td>Awaiting output</td>
</tr>
<tr>
<td>134</td>
<td>Awaiting purge</td>
</tr>
<tr>
<td>135</td>
<td>Active in purge</td>
</tr>
<tr>
<td>136</td>
<td>Active on NJE sysout received</td>
</tr>
<tr>
<td>137</td>
<td>Awaiting NJE transmission</td>
</tr>
<tr>
<td>138</td>
<td>Active on NJE job transmitter</td>
</tr>
</tbody>
</table>

**string StatusDescriptions[]**
Strings describing the various *OperationalStatus* values. Returns NULL.

**string Status**
Not supported for z/OS.

**string JobStatus**
A free form string containing information about the job.

The primary job status is reflected in *OperationalStatus*. *JobStatus* provides additional implementation-specific details.

**datetime TimeSubmitted**
The time that the Job was submitted to execute.

A value of all zeros indicates that the owning element is not capable of reporting a date and time. Therefore, the *ScheduledStartTime* and *StartTime* are reported as intervals relative to the time their values are required.

**datetime ScheduledStartTime**
Not supported for z/OS.

**datetime StartTime**
The time that the Job was actually started.

This may be represented by an actual date and time, or by an interval relative to the time that this property is requested.
datetime ElapsedTime  
The time interval that the Job has been executing  
or the total execution time if the Job is complete.

Note that this property is also present in the  
JobProcessingStatistics class. This is necessary to  
capture the processing information for recurring  
Jobs, since only the ‘last’ run time can be stored in  
this single-valued property.

uint32 JobRunTimes  
Number of times that the Job should be run.

A value of 1 indicates that the Job is NOT  
recurring, while any non-zero value indicates a  
limit to the number of time that the Job will recur.

Zero indicates that there is no limit to the number  
of times that the Job can be processed, but that it is  
terminated either AFTER the UntilTime, or by  
manual intervention.

By default, a job is processed once.

This property is not modifiable.

uint8 RunMonth  
Not supported for z/OS.

sint8 RunDay  
Not supported for z/OS.

sint8 RunDayOfWeek  
Not supported for z/OS.

datetime RunStartInterval  
The time interval after midnight when the Job  
should be processed.

For example, 00000000020000.000000:000 indicates  
that the Job should be run on or after two o’clock,  
local time of UTC time (distinguished using the  
LocalOrUtcTime property).

This property is not modifiable.

uint16 LocalOrUtcTime  
This property indicates whether the time  
represented in the RunStartInterval and UntilTime  
properties represent local or UTC times.

Time values are synchronized worldwide by using  
the enumeration value 2, "UTC Time". Permitted  
values are:
1 Local time
2 UTC time

This property is not modifiable.

datetime UntilTime  
The time after which the Job is invalid or should  
be stopped.

This may be represented by an actual date and  
time, or by an interval relative to the time that this  
property is requested.
A value of all nines indicates that the Job can run indefinitely.

This property is not modifiable.

**string Notify**
User to be notified upon the Job completion or failure.

This property can be modified using the `RequestPropertyChange()` method.

**string Owner**
The User that submitted the Job or the Service/method name/etc. that caused the job to be created.

**uint32 Priority**
Indicates the urgency or importance of execution of the Job.

The lower the number, the higher the priority.

Note that this property is also present in the `JobProcessingStatistics` class. This is necessary to capture the setting information that would influence a Job's results.

This property can be modified using the `RequestPropertyChange()` method.

**uint16 PercentComplete**
Not supported for z/OS.

**boolean DeleteOnCompletion**
Indicates whether or not the Job should be automatically deleted upon completion.

Note that the 'completion' of a recurring Job is defined by its `JobRunTimes` or `UntilTime` properties, OR when the Job is terminated by manual intervention.

If this property is set to `false` and the Job completes, then the extrinsic method `DeleteInstance` MUST be used to delete the Job versus updating this property.

This property is not modifiable.

**uint16 ErrorCode**
Not supported for z/OS.

**string ErrorDescription**
Not supported for z/OS.

**uint16 RecoveryAction**
Not supported for z/OS.

**string OtherRecoveryAction**
Not supported for z/OS.

**string AbendCode**
Job completed with abend code.

**string AccountNumber**
Account number from job card.

**boolean ARMRegistered**
Job is ARM registered indicator.

**string AvailableSchedEnvSystem []**
System names on which the scheduling environment required by job is available. Only valid if job requires a scheduling environment and that environment is available on at least one system.
string AvailableSeclabelSystems []
    System names on which the seclabel associated with the job is available. Only valid if seclabel by system is active in the security product and the seclabel is available on at least on system.

boolean AwaitingARMRestart
    Job awaiting ARM restart indicator.

string Building
    NJE building.
    This property is "Expensive".

uint32 CardCount
    Card (output) count.

string Class
    Job class.
    This property can be modified using the RequestPropertyChange() method.

uint32 CompletionCode
    Completion code (set for conditions marked with + in job completion indicator).

uint8 CompletionType
    Specific completion type:
    0   No completion info
    1   Job ended normally
    2   Job ended by CC
    3   JCL error
    4   Canceled
    5   Abended
    6   Converter abended
    7   Security error
    8   Job failed in EOM

uint16 CopyCount
    Job copy count.
    This property is "Expensive".

string CSName
    The scoping Computer System.

string DefaultPrintDest
    Default print destination.
    This property can be modified using the RequestPropertyChange() method.

string DefaultPunchDest
    Default punch destination.
    This property can be modified using the RequestPropertyChange() method.

string Department
    NJE department.
    This property is "Expensive".

string Device
    Name of device job is active on.

uint32 EstimatedTimeToExecution
    Estimated time to execution in seconds.
    This field is only available if the job is awaiting execution, job is scheduled to run to a WLM managed class, job is not held (duplicate job name, operator hold, etc.), member it has affinity to is available, and the scheduling environment is available.
**datetime ExecutionEndTime**  Execution end time and date.
This property is "Expensive".

**string ExecutionMember**  Execution JES2 member name.
This property is "Expensive".

**string ExecutionNode**  Execution node.
This property can be modified using the RequestPropertyChange() method.

**datetime ExecutionStartTime**  Execution start time and date.
This property is "Expensive".

**string ExecutionSystem**  Execution MVS system name.
This property is "Expensive".

**uint8 HoldIndicator**  Job hold indicator:
1  Not held
2  Held
3  Held for duplicate job name

**uint32 InputCount**  Job input count.
This property is "Expensive".

**string InputDevice**  Input device name.
This property is "Expensive".

**datetime InputStartTime**  Input start time and date.
This property is "Expensive".

**string InputSystem**  Input system or member.
This property is "Expensive".

**boolean JesLogSpinnable**  Jeslog spinnable indicator.

**boolean JobClassModeWLM**  Job class mode for job. If true, mode is WLM, otherwise mode is JES.

**string JobID**  Job identifier.

**boolean JobIsActive**  Indicate job is executing.

**string JobName**  Job name.

**uint8 JobType**  Job type:
1  Started task (STC)
2  Time sharing user (TSU)
3  Batch job (JOB)
4  APPC indicator

**uint32 LineCount**  Line count.
This property is "Expensive".

**string MemberName**  JES2 member on which the job is active.

**string MessageClass**  Message class from job card.

**string NotifyNode**  Notify node.
This property is "Expensive".

**string OriginalJobID**  Original job identifier.
string OriginNode
Original node (node of submittal).

string OSName
The scoping Operating System's name.

uint32 PageCount
Job page count.
This property is "Expensive".

uint8 Phase
Phase job is in:
1. No subchain exists
2. Active in CI in FSS address space
3. Awaiting postscan (batch)
4. Awaiting postscan (damsel)
5. Awaiting volume fetch
6. Awaiting start setup (JES3), Awaiting setup (JES2)
7. Awaiting/active in MDS system select processing
8. Awaiting resource allocation
9. Awaiting unavailable volumes
10. Awaiting volume mounts
11. Awaiting/active in MDS system verify processing
12. Error during MDS processing
13. Awaiting selection on main (JES3),
   Awaiting execution (JES2)
14. Scheduled on main (JES3), Active executing
    (JES2)
17. Awaiting breakdown (JES3), Active in output (JES2)
18. Awaiting MDS restart processing
19. Main MDS processing complete
20. Awaiting output service (JES3), Awaiting hardcopy (JES2)
21. Awaiting output service writer
22. Awaiting reserved services
23. Output service complete
24. Awaiting selection on main (demand select job)
25. Ending function rq waiting or I/O completion
26. Ending function rq not processed
27. Maximum rq index value
128. Active in input processing
129. Awaiting conversion
130. Active in conversion
131. Active in setup
132. Active in spin
133. Awaiting output
134. Awaiting purge
135. Active in purge
136. Active on NJE sysout received
137. Awaiting NJE transmission
138. Active on NJE job transmitter

string ProgrammerName
Programmer name from job card.

string RoomNumber
Job card room number.
string Seclabel
Seclabel from job.

boolean Spin
Indicator of whether jobs in the job class can be spun.

string Subsystem
Subsystem name.

string SystemName
MVS system name on which the job is active.

uint32 WLMActiveJobCount
Number of active jobs in this WLM service class.

uint32 WLMJobsOnQueueCount
Number of jobs on WLM service class queue.

uint32 WLMPosition
Position of this job on WLM service class queue.

uint32 WLMServiceClass
WLM service class.

string WLMServiceClass
WLM service class.

This property can be modified using the RequestPropertyChange() method.

string PercentSpoolUtilization
Percent of spool used by the following CIM profiles the job.

boolean ConverterWait
Job can be converted only by CNVT PCEs that can wait for OS

boolean Independent
Job is set to independent mode.

uint32 JobKey
Job key

boolean JobNotRunReasonJobBusyOnDevice
Job not running because job busy on device

boolean JobNotRunReasonJobClassHeld
Job not running because job class held

boolean JobNotRunReasonJobClassLimitReached
Job not running because job class limit reached

boolean JobNotRunReasonNoSystem
Job not running because no system with right combination of resources

boolean JobNotRunReasonSchedulingEnvironment
Job not running due to unavailable scheduling environment

boolean JobNotRunReasonSeclabelAffinity
Job not running because of seclabel affinity

boolean JobNotRunReasonSpoolNotAvailable
Job not running because spools not available

boolean JobNotRunReasonSystemAffinity
Job not running due to system affinity

boolean Protected
Job is protected

uint32 SpoolDataToken
Spool data token
string SystemAffinity [] System affinity for job
boolean SystemDataSet Job represents a system data set
uint32 TrackGroupCount Number of track groups of spool space used by this job

**Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 Hold()</td>
<td>Holds a job.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 Release()</td>
<td>Releases a job.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 ReleaseOutput()</td>
<td>Releases output for a job.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>
### sint32 RequestPropertyChange()
Changes a property and returns response messages from the generated command.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] string PropertyName</td>
<td>The property to be changed.</td>
</tr>
<tr>
<td>[IN] string PropertyValue</td>
<td>The new value for the property.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

### sint32 Restart()
Restarts a job.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] boolean Hold</td>
<td>Indicates if the job should be held prior to its execution.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

### sint32 Cancel()
Cancels a job.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] boolean PurgeOutput</td>
<td>Indicates if any output associated with the job is to be cancelled.</td>
</tr>
<tr>
<td>[IN] boolean TakeDump</td>
<td>Indicates if a dump should be taken when the job is canceled.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>
IBMzOS_JES3Job

Purpose
This class is a subclass of IBMzOS_Job and contains those properties that are unique to a job that has run, or will run, under JES3.

Inheritance
IBMzOS_Job
← IBMzOS_JES3Job

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_JES3JobProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS_JES3JobProvider.so

Properties
The following properties have been implemented for z/OS:

string Caption
A short description of the class

string Description
A description of the class

string ElementName
Name of given to this instance of the class

datetime InstallDate
Not supported for z/OS.

string Name
The property is overridden by IBMzOS_JES3Job. It contains a unique identifier for this Job.

uint16 OperationalStatus [ ]
The current status of the JES3 Job:
1  No subchain exists
2  Active in CI in FSS address space
3  Waiting postscan (batch)
4  Waiting postscan (damsel)
5  Waiting volume fetch
6  Waiting start setup (JES3), Waiting setup (JES2)
7  Waiting/active in MDS system select processing
8  Waiting resource allocation
9  Waiting unavailable volumes
10  Waiting volume mounts
11  Waiting/active in MDS system verify processing
12  Error during MDS processing
13  Waiting selection on main (JES3),
    Waiting execution (JES2)
14  Scheduled on main (JES3), Active executing (JES2)
17  Waiting breakdown (JES3), Active in output (JES2)
18  Waiting MDS restart processing
Main MDS processing complete
Awaiting output service (JES3), Awaiting hardcopy (JES2)
Awaiting output service writer
Awaiting reserved services
Output service complete
Awaiting selection on main (demand select job)
Ending function rq waiting or I/O completion
Ending function rq not processed
Maximum rq index value
Active in input processing
Awaiting conversion
Active in conversion
Active in setup
Active in spin
Awaiting output
Awaiting purge
Active in purge
Active on NJE sysout received
Awaiting NJE transmission
Active on NJE job transmitter

string StatusDescriptions [ ] Strings describing the various Operational Status values.

string Status Not supported for z/OS.

string JobStatus A free form string representing the Job's status.
The primary status is reflected in the inherited OperationStatus property.
JobStatus provides additional implementation-specific details.

datetime TimeSubmitted The time that the Job was submitted to execute.
A value of all zeros indicates that the owning element is not capable of reporting a date and time. Therefore, the ScheduledStartTime and StartTime are reported as intervals relative to the time their values are required.

datetime ScheduledStartTime Not supported for z/OS.
datetime StartTime The time that the Job was actually started.
This may be represented by an actual date and time, or by an interval relative to the time that this property is requested.
Note that this property is also present in the JobProcessingStatistics class. This is necessary to capture the processing information for recurring Jobs, since only the 'last' run time can be stored in this single-valued property.

datetime ElapsedTime The time interval that the Job has been executing or the total execution time if the Job is complete.
Note that this property is also present in the JobProcessingStatistics class. This is necessary to capture the processing information for recurring Jobs, since only the ‘last’ run time can be stored in this single-valued property.

**uint32 JobRunTimes**

Number of times that the Job should be run.

A value of 1 indicates that the Job is NOT recurring, while any non-zero value indicates a limit to the number of times that the Job will recur.

Zero indicates that there is no limit to the number of times that the Job can be processed, but that it is terminated either AFTER the UntilTime, or by manual intervention.

By default, a Job is processed once. This property is not modifiable.

**uint8 RunMonth**

Not supported for z/OS.

**sint8 RunDay**

Not supported for z/OS.

**sint8 RunDayOfWeek**

Not supported for z/OS.

**datetime RunStartInterval**

The time interval after midnight when the Job should be processed.

For example, 00000000020000.000000:000 indicates that the Job should be run on or after two o’clock, local time of UTC time (distinguished using the LocalOrUtcTime property).

This property is not modifiable.

**uint16 LocalOrUtcTime**

This property indicates whether the time represented in the RunStartInterval and UntilTime properties represent local or UTC times.

Time values are synchronized worldwide by using the enumeration value 2, “UTC Time”. Permitted values are:

1  Local time
2  UTC time

This property is not modifiable.

**datetime UntilTime**

The time after which the Job is invalid or should be stopped. This may be represented by an actual date and time, or by an interval relative to the time that this property is requested. A value of all nines indicates that the Job can run indefinitely.

This property is not modifiable.

**string Notify**

User to be notified upon the Job completion or failure.

This property can be modified using the RequestPropertyChange() method.
string Owner

The User that submitted the Job or the Service/method name/etc. that caused the job to be created.

uint32 Priority

Indicates the urgency or importance of execution of the Job. The lower the number, the higher the priority. Note that this property is also present in the JobProcessingStatistics class. This is necessary to capture the setting information that would influence a Job's results.

This property can be modified using the RequestPropertyChange() method.

uint16 PercentComplete

Not supported for z/OS.

boolean DeleteOnCompletion

Indicates whether or not the Job should be automatically deleted upon completion.

Note that the 'completion' of a recurring Job is defined by its JobRunTimes or UntilTime properties, OR when the Job is terminated by manual intervention.

If this property is set to false and the Job completes, then the extrinsic method DeleteInstance MUST be used to delete the Job versus updating this property.

This property is not modifiable.

uint16 ErrorCode

Not supported for z/OS.

string ErrorDescription

Not supported for z/OS.

uint16 RecoveryAction

Not supported for z/OS.

string OtherRecoveryAction

Not supported for z/OS.

string AbendCode

Job completed with abend code.

string AccountNumber

Account number from job card.

boolean ARMRegistered

Job is ARM registered indicator.

string AvailableSchedEnvSystems []

System names on which the scheduling environment required by job is available. Only valid if job requires a scheduling environment and that environment is available on at least one system.

string AvailableSeclabelSystems []

System names on which the seclabel associated with the job is available. Only valid if seclabel by system is active in the security product and the seclabel is available on at least on system.

boolean AwaitingARMRestart

Job awaiting ARM restart indicator.

string Building

NJE building.

This property is "Expensive".
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32 CardCount</td>
<td>Card (output) count.</td>
</tr>
<tr>
<td>string Class</td>
<td>Job class. This property can be modified using the <code>RequestPropertyChange()</code> method.</td>
</tr>
<tr>
<td>uint32 CompletionCode</td>
<td>Completion code (set for conditions marked with + in job completion indicator.</td>
</tr>
<tr>
<td>uint8 CompletionType</td>
<td>Specific completion type: 0 No completion info 1 Job ended normally 2 Job ended by CC 3 JCL error 4 Canceled 5 Abended 6 Converter abended 7 Security error 8 Job failed in EOM</td>
</tr>
<tr>
<td>uint16 CopyCount</td>
<td>Job copy count. This property is &quot;Expensive&quot;.</td>
</tr>
<tr>
<td>string CSName</td>
<td>The scoping Computer System.</td>
</tr>
<tr>
<td>string DefaultPrintDest</td>
<td>Default print destination. This property can be modified using the <code>RequestPropertyChange()</code> method.</td>
</tr>
<tr>
<td>string DefaultPunchDest</td>
<td>Default punch destination. This property can be modified using the <code>RequestPropertyChange()</code> method.</td>
</tr>
<tr>
<td>string Department</td>
<td>NJE department. This property is &quot;Expensive&quot;.</td>
</tr>
<tr>
<td>string Device</td>
<td>Name of device job is active on.</td>
</tr>
<tr>
<td>uint32 EstimatedTimeToExecution</td>
<td>Estimated time to execution in seconds. This field is only available if the job is awaiting execution, job is scheduled to run to a WLM managed class, job is not held (duplicate job name, operator hold, etc.), member it has affinity to is available, and the scheduling environment is available.</td>
</tr>
<tr>
<td>datetime ExecutionEndTime</td>
<td>Execution end time and date. This property is &quot;Expensive&quot;.</td>
</tr>
<tr>
<td>string ExecutionMember</td>
<td>Execution JES2 member name. This property is &quot;Expensive&quot;.</td>
</tr>
<tr>
<td>string ExecutionNode</td>
<td>Execution node. This property can be modified using the <code>RequestPropertyChange()</code> method.</td>
</tr>
<tr>
<td>datetime ExecutionStart_time</td>
<td>Execution start time and date. This property is &quot;Expensive&quot;.</td>
</tr>
</tbody>
</table>
string ExecutionSystem  Execution MVS system name.
   This property is "Expensive".

uint8 HoldIndicator  Job hold indicator:
   1 Not held
   2 Held
   3 Held for duplicate job name

uint32 InputCount  Job input count.
   This property is "Expensive".

string InputDevice  Input device name.
   This property is "Expensive".

datetime InputStartTime  Input start time and date.
   This property is "Expensive".

string InputSystem  Input system or member.

boolean JesLogSpinnable  Jeslog spinnable indicator.

boolean JobClassModeWLM  Job class mode for job. If true, mode is WLM, otherwise mode is JES.

string JobID  Job identifier.

boolean JobIsActive  Indicate job is executing.

string JobName  Job name.

uint8 JobType  Job type:
   1 Started task (STC)
   2 Time sharing user (TSU)
   3 Batch job (JOB)
   4 APPC indicator

uint32 LineCount  Line count.
   This property is "Expensive".

string MemberName  JES2 member on which the job is active.

string MessageClass  Message class from job card.

string NotifyUserid  Notify user ID.

string OriginalJobID  Original job identifier.

string OriginNode  Original node (node of submittal).

string OSName  The scoping Operating System's name.

uint32 PageCount  Job page count.
   This property is "Expensive".

uint8 Phase  Phase, the job is in. For the values and their meanings, see property OperationalStatus.

string ProgrammerName  Programmer name from job card.

string RoomNumber  Job card room number.

string Seclabel  Seclabel from job.

boolean Spin  Indicator of whether jobs in the job class can be spun.
string Subsystem
Subsystem name.

string SystemName
MVS system name on which the job is active.

uint32 WLMActiveJobCount
Number of active jobs in this WLM service class.

uint32 WLMJobsOnQueueCount
Number of jobs on WLM service class queue.

uint32 WLMPosition
Position of this job on WLM service class queue.

uint32 WLMSchedulingEnvironment
WLM scheduling environment.
This property can be modified using the RequestPropertyChange() method.

string WLMServiceClass
WLM service class.
This property can be modified using the RequestPropertyChange() method.

string PercentSpoolUtilization
Percent of spool used by the job.

uint8 JobNotRunReasonCodes [ ]
List or reasons by system for why job is waiting to run.

string JobNotRunSystems [ ]
List of system names corresponding to JobNotRunReasonCodes.

Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 Hold()[OUT]</td>
<td>Holds a job.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Description</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
<tr>
<td>sint32 Release()</td>
<td>Releases a job.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Description</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>sint32 ReleaseOutput()</td>
<td>Releases output for a job.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 RequestPropertyChange()</td>
<td>Changes a property and returns response messages from the generated command.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] string PropertyName</td>
<td>The property to be changed.</td>
</tr>
<tr>
<td>[IN] string PropertyValue</td>
<td>The new value for the property.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint32 Restart()</td>
<td>Restarts a job.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] boolean Hold</td>
<td>Indicates if the job should be held prior to its execution.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>sint32 Cancel()</td>
<td>Cancels a job.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[IN] boolean PurgeOutput</td>
<td>Indicates if any output associated with the job is to be cancelled.</td>
</tr>
<tr>
<td>[IN] boolean TakeDump</td>
<td>Indicates if a dump should be taken when the job is canceled.</td>
</tr>
<tr>
<td>[IN] datetime TimeoutPeriod</td>
<td>Specifies the maximum amount of time that the client expects the transition to the new state to take.</td>
</tr>
<tr>
<td>[OUT] string ResponseText[]</td>
<td>Command response messages.</td>
</tr>
</tbody>
</table>
IBMzOS_JES2SysoutDataset

Purpose
This class is a subclass of IBMzOS_SysoutDataset and contains those properties that are unique to a job that has run under JES2.

Inheritance
IBMzOS_SysoutDataset
← IBMzOS_JES2SysoutDataset

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_JES2SysoutDatasetProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS_JES2SysoutDatasetProvider.so

Properties
string Caption
A short description of the class

string Description
A description of the class

string ElementName
Name of given to this instance of the class

datetime InstallDate
Not supported for z/OS.

string Name [key]
JES2 Sysout Dataset name

uint16 OperationalStatus [ ]
The current status of the JES2SysoutDataset:
  0 = Unknown
  2 = OK
  6 = Error
  9 = Stopping

string StatusDescriptions [ ]
Not supported for z/OS.

string Status
Not supported for z/OS.

string CSCreationClassName [key]
The scoping ComputerSystem's CreationClassName.

string CSName [key]
The scoping ComputerSystem's Name.

string FSCreationClassName [key]
The scoping FileSystem's CreationClassName.

string FSName [key]
The scoping FileSystem's Name.

string CreationClassName [key]
Indicates the name of the class or the subclass used in the creation of an instance. When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.
uint64 FileSize
datetime CreationDate
datetime LastModified
datetime LastAccessed
boolean Readable
boolean Writeable
boolean Executable
string CompressionMethod
string EncryptionMethod
uint64 InUseCount
string ActiveMember
string ActiveSysname
boolean Burst
uint64 ByteCount
string Class
datetime CreateTime
string DataSetName
uint32 DataSetNumber
string DDName
string Destination
string DeviceName
string FCB
boolean HeldByOperator
boolean HeldBySystem
boolean HeldByUser
string Identifier
boolean IPAddrDest
string JobID
string Jobname
uint16 MaxLogicalRecordLength
Maximum logical record length
This property is "Expensive".

string ModifyModname
Modify=(modname)

string ModifyTrc
Modify=(,trc)

boolean NotSelectable
Not selectable

string OutDisp
Output disposition

string Owner
Sysout owner

uint32 PageCount
Page count

uint8 Priority
Sysout priority

string ProcessMode
Processing mode

string ProcName
Procmname for the step creating this data set

uint32 RecordCount
Record count

string RecordFormat
Record format
This property is "Expensive".

string Seclabel
Seclabel for sysput

uint32 SegmentID
Segment ID (zero if data set is not segmented)

boolean Spin
Spin data set

string StepName
Stepname for the step creating this data set
This property is "Expensive".

string Subsystem
Subsystem name

string SystemHoldReason
Reason for system hold

string TPJobName
APPC transaction program jobname that created this data set

string TranslateTable [ ]
Printer translate table

string UCS
UCS

string WriterName
External writer name

string JobToken
Job token

string OutputGroupElement
Sysout group name

datetime OutputGroupElementCreateTime
JOE creation time

uint16 OGID1
JOE ID1

string Forms
specifies the forms on which the data set is to be printed

string Flash
specifies the form overlay
**IBMzOS_JES3SysoutDataset**

**Purpose**
This class is a subclass of IBMzOS_SysoutDataset and contains those properties that are unique to a job that has run under JES3.

**Inheritance**
IBMzOS_SysoutDataset

  ← IBMzOS_JES3SysoutDataset

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

  IBMzOS_JES3SysoutDatasetProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

  libcmpiIBMzOS_JES3SysoutDatasetProvider.so

**Properties**
The properties of IBMzOS_JES3SysoutDataset are the same as for IBMzOS_JES2SysoutDataset (see [“IBMzOS_JES2SysoutDataset” on page 180](#) with some exceptions:

IBMzOS_JES3SysoutDataset does not provide the following properties of IBMzOS_JES2SysoutDataset:

  • OutputGroupElement
  • OutputGroupElementCreateTime
  • OGID1

The following properties are only part of IBMzOS_JES3SysoutDataset:

  boolean HeldForTSO          Sysout is held for TSO
  boolean HeldForExternalWriter  Sysout is held for external writer
IBMzOS_Job

Purpose
This class represents a z/OS job. Jobs are associated with a subsystem, such as JES2, JES3, or MSTR. Some properties may require significant overhead, including I/O, to obtain their data. These properties are identified with the qualifier of "Expensive". To reduce system overhead, the provider will only return the values for these expensive properties if they are explicitly requested by name.

Inheritance
Subclasses are IBMzOS_JES2Job (see “IBMzOS_JES2Job” on page 161) and IBMzOS_JES3Job (see “IBMzOS_JES3Job” on page 171).
IBMzOS_JobsManagementSettings

Purpose
The IBMzOS_JobsManagementSettings class provides a mechanism by which users can influence the behavior of the IBMzOS_JES2SysoutDataset, IBMzOS_JES3SysoutDataset, IBMzOS_JES2Jobs, and IBMzOS_JES3Jobs providers.

Properties

string Caption          A short description of the class
string Description      A description of the class
string ElementName      Name given to this instance of the class
string InstanceID [Key] Within the scope of the instantiating NameSpace, InstanceID opaquely and uniquely identifies an instance of this class. In order to ensure uniqueness within the NameSpace, the value of InstanceID SHOULD be constructed using the following algorithm:
<OrgID>:<LocalID>

where <OrgID> and <LocalID> are separated by a colon ':', and where <OrgID> MUST include a copyrighted, trademarked or otherwise unique name that is owned by the business entity creating/defining the InstanceID, or is a recognized global authority (This is similar to the <Schema Name>_<Class Name> structure of Schema class names.) In addition, to ensure uniqueness <OrgID> MUST NOT contain a colon (':'). When using this algorithm, the first colon in InstanceID MUST be between <OrgID> and <LocalID>.

<LocalID> is chosen by the business entity and SHOULD not be re-used to identify different underlying (real-world) elements. If the above 'preferred' algorithm is not used, the defining entity MUST assure that the resultant InstanceID is not re-used across any InstanceIDs produced by this or other providers for this instance's NameSpace.

For DMTF defined instances, the 'preferred' algorithm MUST be used with the <OrgID> set to 'CIM'.

uint32 MaxInstances     The maximum number of instances that can be returned.
uint32 MaxProperties    The maximum number of properties that can be returned
IBMzOS_Subsystem

Purpose
This class represents a z/OS Subsystem.

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_SubsystemProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libcmpiIBMzOS_SubsystemProvider.so

Properties

string Caption A short description of the class
string Description A description of the class
string ElementName Name given to this instance of the class
datetime InstallDate Not supported for z/OS.
string Name [key] Subsystem name
uint16 OperationalStatus [ ] The current status of the JobSubSystem:

   0  Unknown
   2  OK
   6  Error
   9  Stopping

string StatusDescriptions [ ] Strings describing the various Operational Status values.
string Status Not supported for z/OS.
uint16 EnabledState Indicates the Enabled or Disabled state.
string OtherEnabledState String describing the Enabled State value.
uint16 RequestedState The last requested State.
uint16 EnabledDefault Indicates the default value for Enabled State.
datetime TimeOfLastStateChange Not supported for z/OS.
string SystemCreationClassName [key] The scoping System's CreationClassName.
string SystemName [key] The scoping System's Name.
string CreationClassName [key] Indicates the name of the class or the subclass used in the creation of an instance. When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.
string PrimaryOwnerName Not supported for z/OS.
string PrimaryOwnerContact Not supported for z/OS.
string StartMode

StartMode is a string value indicating whether the Service is automatically started by a System, Operating System, etc. or only started upon request.

This property is deprecated. Use the EnabledDefault property inherited from EnabledLogicalElement instead.

boolean Started

True if subsystem is active.

boolean Dynamic

True if subsystem is dynamic.

boolean DynamicCommands

True if subsystem responds to SETSSI command.

boolean Primary

Indicator for primary subsystem

uint8 Type

Subsystem type code:
1 Unknown
2 JES2
3 JES3
IBMzOS_SysoutDataset

Purpose
This class represents a z/OS sysout dataset. Some properties may require significant overhead, including I/O, to obtain their data. These properties are identified with the qualifier of "Expensive". To reduce system overhead, the provider will only return the values for these expensive properties if they are explicitly requested by name.

Inheritance
Subclasses are
- IBMzOS_JES2SysoutDataset (see "IBMzOS_JES2SysoutDataset" on page 180) and
- IBMzOS_JES3SysoutDataset (see "IBMzOS_JES3SysoutDataset" on page 183).
Association IBMzOS_SubsystemJES2Jobs

Purpose
This class associates an IBMzOS_Subsystem with an IBMzOS_JES2Job.

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
IBMzOS_SubsystemJES2JobsProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
libcmpiIBMzOS_SubsystemJES2JobsProvider.so

Association IBMzOS_SubsystemJES3Jobs

Purpose
This class associates an IBMzOS_Subsystem with an IBMzOS_JES3Job.

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
IBMzOS_SubsystemJES3JobsProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
libcmpiIBMzOS_SubsystemJES3JobsProvider.so

Association IBMzOS_UsesJES2SysoutDatasets

Purpose
This class associates an IBMzOS_JES2Job with an IBMzOS_JES2SysoutDataset.

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
IBMzOS_UsesJES2SysoutDatasetsProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
libcmpiIBMzOS_UsesJES2SysoutDatasetsProvider.so

Association IBMzOS_UsesJES3SysoutDatasets

Purpose
This class associates an IBMzOS_JES3Job with an IBMzOS_JES3SysoutDataset.
**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_UsesJES3SysoutDatasetsProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_UsesJES3SysoutDatasetsProvider.so
OS management Cluster classes

The classes described in this section are implemented by z/OS to instrument the z/OS "Systems Complex" (Sysplex) clustering facility.

For using these providers you need an extra security setup as described in “Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers” on page 38.
IBMzOS_CFRMCoupleDataset

**Purpose**
This class represents Coupling Facility Resource Manager (CFRM) couple datasets. A CFRM couple dataset contains CFRM policies, one of which can be active (started), defining how z/OS manages coupling facility resources.

A CFRM couple dataset can be the active primary, or optionally, the active alternate couple dataset supporting the CFRM coupling function. Minimally, a CFRM couple dataset must be in use as the active primary CFRM couple dataset for CFRM coupling function to be active.

**Inheritance**
IBMzOS_CoupleDataset
–> IBMzOS_CFRMCoupleDataset

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS_CFRMCoupleDatasetProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS_CFRMCoupleDatasetProvider.so

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string Name</td>
<td>The name of the couple dataset represented by an instance of this class.</td>
</tr>
<tr>
<td>uint32 NumberOfStructures</td>
<td>The number of coupling facility (CF) structures that the CFRM couple dataset is formatted to support. It is the maximum number of structures that can be defined for use in a policy contained in this couple dataset.</td>
</tr>
<tr>
<td>uint32 NumberOfConnectors</td>
<td>Identifies the number of connectors per structure that the couple dataset is formatted to support. Connectors are programs running under z/OS that establish a connection to a CF structure. It is the maximum number of concurrent connectors that can be supported for each structure defined in the couple dataset.</td>
</tr>
<tr>
<td>uint32 NumberOfCFs</td>
<td>The number of coupling facilities the couple dataset is formatted to support. It is the maximum number of CFs that can be defined for use in a CFRM policy contained in this couple dataset.</td>
</tr>
<tr>
<td>uint32 NumberOfPolicies</td>
<td>The number of administrative (inactive) policies that the couple dataset is formatted to support.</td>
</tr>
<tr>
<td>boolean SystemManagedDuplexing</td>
<td>Indicates whether or not the couple dataset is</td>
</tr>
</tbody>
</table>
formatted to support the use of the system-managed duplexing rebuild process.

System-managed duplexing rebuild is a process managed by z/OS that allows a structure to be maintained as a duplexed pair. The process is controlled by CFRM policy definitions as well as by the program owning the structure. The process can be initiated via operator command (SETXCF), programming interface (IXLREBLD), or can be z/OS-initiated. Note that user-managed duplexing rebuild is controlled and initiated in the same manner as system-managed duplexing rebuild, but is managed by the program owning the structure and applies only to cache structures.

### boolean SystemManagedRebuild

Indicates whether or not the couple dataset is formatted to support the use of the system-managed structure rebuild process.

System-managed structure rebuild is a process managed by z/OS that allows a structure to be rebuilt by z/OS. The process is controlled by CFRM policy definitions as well as by the program owning the structure. The process can be initiated via operator command (SETXCF), programming interface (IXLREBLD), or can be z/OS-initiated. Note that user-managed structure rebuild is controlled and initiated in the same manner as system-managed rebuild, but is managed by the program owning the structure and applies only to cache structures.

### boolean MessageBased

Indicates whether or not the couple dataset is formatted to support the use of message-based CFRM event notification and confirmation capabilities.
IBMzOS_CFRMPolicy

Purpose
This class represents administrative (inactive) Coupling Facility Resource Manager (CFRM) policies. CFRM policies are used to control Coupling Facility (CF) and CF structure resources available to a z/OS Sysplex (Systems Complex). There can be only one active CFRM policy and some number of administrative (inactive) policies.

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_CFRMPolicyProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_CFRMPolicyProvider.so

Properties
string Caption
A short description of the class.

string Description
A description of the class.

string ElementName
Name given to this instance of the class.

datetime InstallDate
A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.

string Name [key]
Name of CFRM Policy

uint16 OperationalStatus [ ]
The current status of the SysplexCoupleDataset:

0 = Unknown
2 = OK
6 = Error
9 = Stopping

string StatusDescriptions [ ]
Strings describing the various Operational Status values.

string Status
A string indicating the current status

string EnabledState
Indicates the Enabled or Disabled state.

string OtherEnabledState
String describing the Enabled State value.

uint16 RequestedState
The last requested State.

uint16 EnabledDefault
Indicates the default value for Enabled State.

datetime TimeOfLastStateChange
The date and time Enabled State was last changed.

string PolicyText
This property contains the CFRM policy statements that define the Coupling Facilities (CFs) and CF structures that are eligible to be used by programs operating in the Sysplex when this policy is activated (started) via the StartPolicy() method.
The CFRM policy, as defined by its *PolicyText*, governs many aspects of the use of CFs and CF structures by the Sysplex. For example, it governs CF structure placement, fixing, recovery and availability considerations.

**Methods**

- **StartPolicy0**
  - Starts a policy.

- **StopPolicy0**
  - Stops a policy.
IBMzOS_CFStructure

**Purpose**
This class represents a zSeries Coupling Facility Structure.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_CFStructureProviderModule

**Provider library**
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is

libCmpiIBMzOS_CFStructureProvider.so

**Properties**
- **string Caption**
  A short description of the class.
- **string Description**
  A description of the class.
- **string ElementName**
  Name given to this instance of the class.
- **datetime InstallDate**
  A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
- **string Name**
  The Name of the structure as defined in the CFRM policy.
- **uint16 OperationalStatus [ ]**
  The current status of the CF Structure:
  - 0 Unknown
  - 2 OK
  - 6 Error
  - 9 Stopping
- **string StatusDescriptions [ ]**
  Strings describing the various Operational Status values.
- **string Status**
  A string indicating the current status
- **uint16 EnabledState**
  Indicates the Enabled or Disabled state.
- **string OtherEnabledState**
  String describing the Enabled State value
- **uint16 RequestedState**
  The last requested State
- **uint16 EnabledDefault**
  Indicates the default value for Enabled State
- **datetime TimeOfLastStateChange**
  The date and time Enabled State was last changed.
- **uint64 IdentityToken [key]**
  The generated identity value for sysplex cluster.
  (Part 1 of 2)
- **string IdentityName [key]**
  The generated identity value for sysplex cluster.
  (Part 2 of 2)
- **uint32 State**
  CF structure operational state:
  - 1 Okay
  - 2 Unknown
  - 3 Error
  - 4 Stopping
uint32 SubState
CF structure substate:
1 Normal (no exceptional conditions).
2 Temporarily degraded (alter in progress, structure dump serialization held).
3 Permanently degraded (allocated smaller than desired size, pending CFRM policy change).
4 Recovering (Valid only when the value of 'State' is 1 (Okay) or 4 (Stopping)).

uint8 Type
Structure type based on exploiter allocation requirements:
0x03 List
0xFF Lock
0x04 Cache
0xFE Serialized List

boolean AllowAlter
Indicator of whether this structure can be dynamically altered, based on current conditions. All active connectors to the structure specified ALLOWAUTO = YES on the IXLCONN connect request.

boolean AllowAuto
All active connectors to the structure specified ALLOWREBLD = YES on the IXLCONN connect request.

boolean AllowRebuild
All active connectors to the structure specified ALLOWDUPREBLD = YES on the IXLCONN connect request.

boolean AllowDupRebuild
All active connectors to the structure specified ALLOWALTER = YES on the IXLCONN connect request.

boolean IsDuplexed
Indicator of whether this structure actually is duplexed at this time. Only when this property indicates that the structure is duplexed, will the following properties be valid:
- MaximumStructureSize2
- InitialStructureSize2
- MinimumStructureSize2
- OverFullThreshold2
- StructureVersion2
- CFName2
- CurrentStructureSize2

boolean PendPolicyChange
Indicates that there is a change pending in structure policy.

boolean Disposition
Defines whether the structure is persistent when there are no longer any defined connections (active or failed):
FALSE Keep
TRUE Delete

string CFName1
The name of the Coupling Facility in which this structure instance has been allocated.
It is possible to have two structure instances due to rebuild-in-progress or duplexing.

It is possible to have no structure instances when the structure is not currently allocated.

When Duplexed this is the 'Old' instance of the structure.

string CFName2 The name of the Coupling Facility in which the 'New' structure instance has been allocated. Null if not allocated.

This property is only valid when Duplexed.

string StructureVersion1 Structure version number for the currently allocated instance of the structure.

It is possible to have two structure instances due to rebuild-in-progress or duplexing. It is possible to have no structure instances when the structure is not currently allocated.

When Duplexed this is the 'Old' instance of the structure.

string StructureVersion2 Structure version number for the 'New' instance of the structure, when the structure is in the process of rebuilding or has been duplexed.

This property is only valid when Duplexed.

uint32 MaximumStructureSize1 The maximum size to which this instance of the structure can be expanded, in units of 4KB.

When Duplexed this is the 'Old' instance of the structure.

uint32 MaximumStructureSize2 The maximum size to which the 'New' instance of the structure can be expanded, in units of 4KB.

This property is only valid when Duplexed.

uint32 InitialStructureSize1 The requested initial structure allocation size, in units of 4KB, for this instance of the structure.

When Duplexed this is the 'Old' instance of the structure.

uint32 InitialStructureSize2 The requested initial structure allocation size, in units of 4KB, for the 'New' instance of the structure.

This property is only valid when Duplexed.

uint32 MinimumStructureSize1 The minimum size at which this instance of the structure can be allocated or contracted to, in units of 4KB.

When Duplexed this is the 'Old' instance of the structure.
uint32 MinimumStructureSize2
The minimum size at which the 'New' instance of the structure can be allocated or contracted to, in units of 4KB.
This property is only valid when Duplexed.

uint32 CurrentStructureSize1
The allocated structure size, in units of 4 KB, for this instance of the structure. Not provided if the structure is not allocated.
When Duplexed this is the 'Old' instance of the structure.

uint32 CurrentStructureSize2
The allocated structure size, in units of 4 KB, for the 'New' instance of the structure. Not provided if the structure is not allocated.
This property is only valid when Duplexed.

uint32 SysMgdProcessLevel1
System Managed Process Level required by the instance of the structure to participate in a system-managed process.
When Duplexed this is the 'Old' instance of the structure.

uint32 SysMgdProcessLevel2
System Managed Process Level required by the 'New' instance of the structure to participate in a system-managed process.
This property is only valid when Duplexed.

uint32 ElementCount1
When Duplexed this is the 'Old' instance of the structure.

uint32 ElementCount2
This property is only valid when Duplexed.

uint32 EntryCount1
When Duplexed this is the 'Old' instance of the structure.

uint32 EntryCount2
This property is only valid when Duplexed.

uint32 EMCCount1
Event Monitor Controls count for List Structures. Invalid for Cache structures and Lock structures.
When Duplexed this is the 'Old' instance of the structure.
uint32 EMCCount2
Event Monitor Controls count for 'New' List Structures. Invalid for Cache structures and Lock structures.
This property is only valid when Duplexed.

uint32 LockCount1
When Duplexed this is the 'Old' instance of the structure.

uint32 LockCount2
This property is only valid when Duplexed.

string LogicalVersion1
Logical Version number for the instance of the structure.
When Duplexed this is the 'Old' instance of the structure.

string LogicalVersion2
Logical Version number for the 'New' instance of the structure.
This property is only valid when Duplexed.

string PreferenceList1 [ ]
Structure Preference List for the instance of the structure. It is an array of up to 8 Coupling Facility names.
When Duplexed this is the 'Old' instance of the structure.

string PreferenceList2 [ ]
Structure Preference List for the instance of the structure. This is an array of up to 8 coupling facility names.
This property is only valid when Duplexed.

string ExclusionList1 [ ]
The Structure Exclusion List for the instance of the structure. This is an array of up to 8 coupling facility names.
When Duplexed this is the 'Old' instance of the structure.

string ExclusionList2 [ ]
Structure Exclusion List for the 'New' instance of the structure. This is an array of up to 8 coupling facility names.
This property is only valid when Duplexed.

uint32 AccessTimeMax1
This instance of the structure was allocated with access time for IXLCONN ACCESSTIME(MAXIMUM).
When Duplexed this is the 'Old' instance of the structure.

uint16 AccessTimeMax2
The 'New' instance of the structure was allocated with access time for IXLCONN ACCESSTIME(MAXIMUM).
This property is only valid when Duplexed.

**uint16 MaximumConnections1**

The maximum number of connections allowed when the structure was allocated in the coupling facility.

When Duplexed this is the 'Old' version of the structure.

**uint16 MaximumConnections2**

The maximum number of connections allowed when the 'New' instance of the structure was allocated in the coupling facility.

This property is only valid when Duplexed.

**uint8 FullThreshold1**

Percentage value for the structure full monitoring threshold for the structure, as defined in CFRM policy. This threshold is set on-platform and is not currently settable through the resource model.

When Duplexed this is the 'Old' version of the structure.

**uint8 FullThreshold2**

Percentage value for the structure full monitoring threshold for the 'New' version of the structure, as defined in CFRM policy. This threshold is set on-platform and is not currently settable through the resource model.

This property is only valid when Duplexed.

**uint8 RebuildPercent1**

REBUILDPERCENT for the instance of the structure as specified in CFRM active policy. Not valid indicates not specified.

When Duplexed this is the 'Old' version of the structure.

**uint8 RebuildPercent2**

REBUILDPERCENT for the 'New' instance of the structure as specified in CFRM active policy. Not valid indicates not specified.

This property is only valid when Duplexed.

**uint8 DuplexPolicy1**

The effective DUPLEX option for the structure as specified in the CFRM active policy or defaulted.

When Duplexed this is the 'Old' version of the structure.

**uint8 DuplexPolicy2**

The effective DUPLEX option for the 'New' structure as specified in the CFRM active policy or defaulted.

This property is only valid when Duplexed.

**boolean OverFullThreshold1**

Indicator of whether or not the instance of the structure is currently in violation of its structure full monitoring threshold.

When Duplexed this is the 'Old' instance of the structure.
boolean OverFullThreshold2  Indicator of whether or not the 'New' instance of the structure is currently in violation of its structure full monitoring threshold.
This property is only valid when Duplexed.

boolean AllowAutoAlter1  ALLOWAUTOALT(YES) was specified in the CFRM active policy for the structure.
When Duplexed this is the 'Old' instance of the structure.

boolean AllowAutoAlter2  ALLOWAUTOALT(YES) was specified in the CFRM active policy for the 'New' structure.
This property is only valid when Duplexed.

boolean EnforceOrder1  ENFORCEORDER(YES) was specified in the CFRM active policy for the structure.
When Duplexed this is the 'Old' instance of the structure.

boolean EnforceOrder2  ENFORCEORDER(YES) was specified in the CFRM active policy for the 'New' structure.
This property is only valid when Duplexed.

boolean AllowReallocate1  ALLOWREALLOCATE(YES) was specified in the CFRM active policy for the structure.
When Duplexed this is the 'Old' instance of the structure.

boolean AllowReallocate2  ALLOWREALLOCATE(YES) was specified in the CFRM active policy for the 'New' structure.
This property is only valid when Duplexed.

boolean AccessTimeNoLimit1  The instance of the structure was allocated with IXLCONN ACCESSTIME(NOLIMIT)
When Duplexed this is the 'Old' instance of the structure.

boolean AccessTimeNoLimit2  The 'New' instance of the structure was allocated with IXLCONN ACCESSTIME(NOLIMIT).
This property is only valid when Duplexed.

When Duplexed this is the 'Old' instance of the structure.

uint32 MaxElementCount2  The maximum Element Count for the 'New' structure. List set element count for List structures. Data area element count for Cache Structures. Invalid for Lock Structures. This property is only valid when Duplexed.

uint32 MaxEntryCount1  The maximum Entry Count for the structure. List
set entry count for List and Lock Structures.
Directory entry count for cache structures.
When Duplexed this is the 'Old' instance of the structure

**uint32 MaxEntryCount2**
The maximum Entry Count for the 'New' structure.
List set entry count for List and Lock Structures.
Directory entry count for cache structures.
This property is only valid when Duplexed.

**uint32 MaxEMCCount1**
The maximum Event Monitor Controls count for List Structures. Invalid for Cache structures and Lock structures. When Duplexed this is the 'Old' instance of the structure.

**uint32 MaxEMCCount2**
The maximum Event Monitor Controls count for 'New' List Structures. Invalid for Cache structures and Lock structures. This property is only valid when Duplexed.

**Methods**

**uint32 StartRebuild()**
Asynchronously rebuilds the structure into the same or a different CF than the one in which it is currently located.

Only works if supported by exploiters. The Location parameter specifies the location where the new structure can be built.

The LessConnAction parameter indicates whether the rebuild should be allowed to continue, in spite of a degradation in connectivity to the new structure.

A rebuild operation should only be requested for structures that are identified as rebuild capable. The rebuild will be performed asynchronously. The return and reason codes will indicate whether the operation was initiated successfully. A property change event will be generated asynchronously when the rebuild has completed.

Coupling Facility Structure operations should only be invoked from a single system in the sysplex.

**uint32 StopRebuild()**
Stops a Rebuild operation.

A property change event will be generated when the operation has completed.

Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.

**uint32 StartDuplex()**
Asynchronously establishes duplexing for the specified structure.

Only works if supported by exploiters. The request to start duplexing will be performed asynchronously. The return and reason codes will indicate whether the operation was initiated.
Successfully. A property change event will be generated asynchronously when the duplexing has completed.

Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.

**uint32 StopDuplex()**

Stops duplexing.

The required Keep parameter indicates which structure is to persist after duplexing has been stopped. The request to stop duplexing will be performed asynchronously. The return and reason codes will indicate whether the operation was initiated successfully. A property change event will be generated asynchronously when operation has completed.

Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.

**uint32 Force()**

Asynchronously forces the deallocation of a persistent structure.

Force of a structure does not work if there are any active connectors to the structure, and may or may not work if there are failed connectors to the structure. The return and reason codes will indicate whether the operation was initiated successfully. CFStructure property change event or life cycle event will be generated asynchronously when the Force operation has completed.

**uint32 ForceAll()**

Asynchronously forces the deletion of all failed-persistent connections for this structure.

The return and reason codes will indicate whether the operation was initiated successfully. Connector life cycle events or relationship-related events will be generated asynchronously when the failed persistent connectors are deleted.

**Associations**

**IBMzOS_CFStrDependsOn**

Source: IBMzOS_CFStructure

Target: IBMzOS_CFStructureConnector

see page 228

**Indications**

**IBMzOS_CFStructureInstCreation**

A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStructure class has been created.

**IBMzOS_CFStructureInstDeletion**

A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStructure class has been deleted.
IBMzOS_CFSStructureInstModification
A 'life cycle' indication that indicates that an
instance of the IBMzOS_CFSStructure class has been
modified.
IBMzOS_CFStructureConnector

Purpose
This class represents a zSeries Coupling Facility Structure Connector.

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
IBMzOS_CFStructureConnectorProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
libcmpiIBMzOS_CFStructureConnectorProvider.so

Properties
string Caption
A short description of the class.

string Description
A description of the class.

string ElementName
Name given to this instance of the class.

datetime InstallDate
A datetime value indicating when the object was
installed. A lack of a value does not indicate that
the object is not installed.

string Name
The Connector name.

uint16 OperationalStatus [ ]
The current status of the CF connector:
0  Unknown
2  OK
6  Error
9  Stopping

string StatusDescriptions [ ]
Strings describing the various Operational Status
values.

string Status
A string indicating the current status

uint16 EnabledState
Indicates the Enabled or Disabled state.

string OtherEnabledState
String describing the Enabled State value

uint16 RequestedState
The last requested State

uint16 EnabledDefault
Indicates the default value for Enabled State

datetime TimeOfLastStateChange
The date and time Enabled State was last changed.

uint64 IdentityToken [key]
The generated identity value for sysplex cluster.
(Part 1 of 2)

string IdentityName [key]
The generated identity value for sysplex cluster.
(Part 2 of 2)

string ConnectorStructureName
The CFStructure name for the connection.

string ConnectorSystemName
OperatingSystem name for the system where the
connector is running.
string ConnectorProcessName  Process name for the process in which the connector is running (for z/OS this is a jobname).

string ConnectorProcessID   Unique process identification for the process in which the connector is running (for z/OS this is a token).

uint32 State                Operational state of the CF connector:
                             0   Okay
                             2   Unknown
                             6   Error
                             9   Stopping

string ConnectorLevel       Connector-specified level information, or 0 if not provided by the connector.

boolean FailureIsolation    Indicator of whether or not the structure as currently allocated satisfies this connector's requirements for failure-isolation.

boolean Disposition         Indicator of the connector disposition. Defines whether the connection is persistent if the connection abnormally terminates.
                             FALSE  Delete
                             TRUE   Keep

boolean NonVolatileRequest  Indicator of whether the connector requested non-volatility.

string ConnectorIdentifier  Connector Identifier.

string ConnectorVersion     Connector version number.

string ConnectorData        Connector data.

uint8 ConnectorInfoLevel    Connector Level of information.

uint8 ConnectorCFLevelRequired  Connector CF Level required.

boolean AllowRebuild        Indicates that the connector was connected with ALLOWREBUILD = YES

boolean AllowDupRebuild     Indicates that the connector was connected with ALLOWDUPBUILD = YES

boolean AllowAuto           Indicates that the connector was connected with ALLOWAUTO = YES

boolean AllowAlter          Indicates that the connector was connected with ALLOWALTER = YES

boolean Suspend             Indicates that the connector was connected with ALLOWALTER = YES, SUSPEND = YES

boolean AllowRatio          Indicates that the connector was connected with ALLOWALTER = YES, RATIO = YES

uint8 MinEntry              Indicates the value the connector specified for MINENTRY

uint8 MinElement            Indicates the value the connector specified for MINELEMENT

uint8 MinEMC                Indicates the value the connector specified for MINEMC
Methods

uint32 Force()  Asynchronously forces deletion of a failed connector to a structure, following a failure.

For some structures this is not permitted unless the structure itself is also forced (deallocated). This operation can only be performed against a structure connector in the ERROR state. The return and reason codes will indicate whether the operation was initiated successfully. Structure connector property change events or life cycle events will be generated asynchronously when the force operation has completed.

Indications

IBMzOS_CFStrConnectorInstCreation
A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been created.

IBMzOS_CFStrConnectorInstDeletion
A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been deleted.

IBMzOS_CFStrConnectorInstModification
A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been modified.
IBMzOS_CoupleDataset

Purpose
This class represents the methods and properties common to all specific types of z/OS couple datasets. Examples of z/OS couple datasets include z/OS System Complex (Sysplex) and Coupling Facility Resource Manager (CFRM) couple datasets.

Inheritance
Subclasses are
- IBMzOS_SysplexCoupleDataset (see “IBMzOS_SysplexCoupleDataset” on page 224) and
- IBMzOS_CFRMCoupleDataset (see “IBMzOS_CFRMCoupleDataset” on page 192).

Properties
- string Caption
  A short description of the class.
- string Description
  A description of the class.
- string ElementName
  Name given to this instance of the class.
- datetime InstallDate
  Not supported for z/OS.
- string Name [key]
  Name of Couple Dataset
- uint16 OperationalStatus [ ]
  The current status of the SysplexCoupleDataset:
  0 = Unknown
  2 = OK
  6 = Error
  9 = Stopping
- string StatusDescriptions [ ]
  Strings describing the various Operational Status values.
- string Status
  Not supported for z/OS.
- string CSCreationClassName [key]
  The scoping ComputerSystem's CreationClassName.
- string CSName [key]
  The scoping ComputerSystem's Name.
- string FSCreationClassName [key]
  The scoping FileSystem's CreationClassName.
- string FSName [key]
  The scoping FileSystem's name.
- string CreationClassName [key]
  CreationClassName indicates the name of the class or the subclass used in the creation of an instance.
  When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.
- uint64 FileSize
  Not supported for z/OS.
- datetime CreationDate
  Not supported for z/OS.
- datetime LastModified
  Not supported for z/OS.
- datetime LastAccessed
  Not supported for z/OS.
boolean Executable
Indicates that the File is executable.

string CompressionMethod
Not supported for z/OS.

string EncryptionMethod
Not supported for z/OS.

uint64 InUseCount
Not supported for z/OS.

string SysplexName
This is the name of the z/OS Sysplex to which the couple dataset represented by an instance of this class belongs.

Couple datasets are formatted for use in a particular Sysplex and cannot be used by a Sysplex other than the one for which they have been formatted.

string Volser
This is the volume serial of the logical volume on which the couple dataset is defined.

string DeviceNumber
This is the z/OS device number of the logical volume on which the couple dataset is defined. The device number is local to the z/OS system from which this instance was obtained.

A logical volume may have different device numbers on different z/OS systems in the Sysplex, even though it is the same logical volume being shared by the different z/OS systems.

string NarrativeInfo
This property contains information used by the couple dataset owner to provide additional descriptive information about the couple dataset and its usage. This information includes formatting characteristics and any special functions or attributes that the couple dataset supports.

string Type
This property identifies the type of couple dataset the instance represents. Some examples of couple dataset types include CFRM and SYSPLEX. There are other types of couple datasets, although not all of them are externalized through CIM.

boolean IsPrimary
This property identifies whether the couple dataset represented by an instance is currently in use as the primary couple dataset for its type.

A value of True indicates that this instance represents the couple dataset that is currently in use as the primary couple dataset of its type.

boolean IsAlternate
This property identifies whether the couple dataset represented by an instance is currently in use as the alternate couple dataset for its type.

A value of True indicates that this instance represents the couple dataset that is currently in use as the alternate couple dataset of its type.

uint32 MaximumNumberOfSystems
This property identifies the number of z/OS
systems in the Sysplex that the couple dataset represented by this instance was formatted to support.

**datetime FormatTime**

This property identifies the local time that the couple dataset was formatted.

*Note:* This property is in the local time of the operating system host servicing the request.

**boolean IsSynchronized**

This property applies only to instances representing couple datasets that are currently in use as the alternate couple dataset for their type.

A value of True indicates that the couple dataset has been fully synchronized with the primary couple dataset of its type.

A value of False indicates that the couple dataset is still in the process of synchronizing with the primary couple dataset of its type.

An alternate couple dataset must be fully synchronized with the primary couple dataset of its type in order to provide failover capability in the event of an error affecting the primary couple dataset.

**boolean ErrorState**

This property identifies whether the couple dataset is in an error state. When True, the couple dataset has experienced a permanent error and is in the process of being removed from active use.

**uint32 NumberOfStructures**

This is the number of coupling facility (CF) structures that the CFRM couple dataset is formatted to support. It is the maximum number of structures that can be defined for use in a policy contained in this couple dataset.

**uint32 NumberOfConnectors**

Connectors are programs running under z/OS that establish a connection to a CF structure. This property identifies the number of connectors per structure that the couple dataset is formatted to support. It is the maximum number of concurrent connectors that can be supported for each structure defined in the couple dataset.

**uint32 NumberOfCFs**

This is the number of coupling facilities the couple dataset is formatted to support. It is the maximum number of CFs that can be defined for use in a CFRM policy contained in this couple dataset.

**uint32 NumberOfPolicies**

This is the number of administrative (inactive) policies that the couple dataset is formatted to support.

### Methods

**uint32 SwitchPrimary()**

This method switches the couple dataset represented by this instance as follows:
If the instance represents a current in-use alternate couple dataset, it is switched to become the current primary couple dataset. If the alternate couple dataset is not fully synchronized or is in an error state, the method returns an error.

If the instance represents a current in-use primary couple dataset, then it is switched out and the current in-use alternate couple dataset is switched to become the primary. If there is no current in-use alternate couple dataset or the in-use alternate couple dataset is not fully synchronized or in an error state, the method returns an error.

This method functions like the z/OS operator command:

```
SETXCF COUPLE,TYPE=___,PSWITCH
```
IBMzOS_CouplingFacility

**Purpose**
This class represents a zSeries Coupling Facility, which is the system that manages a Sysplex (System Complex).

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_CouplingFacilityProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmplIBMzOS_CouplingFacilityProvider.so

**Properties**
- **string Caption**
  A short description of the class. Returns 'IBM z/OS Coupling Facility'.
- **string Description**
  A description of the class. Returns This is an IBM z/OS Coupling Facility.
- **string ElementName**
  Name given to this instance of the class (same as Name)
- **datetime InstallDate**
  A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
- **string Name**
  Coupling Facility Logical Name as assigned by CFM policy definitions.
  
  Note: CF Name is not considered an immutable property of a Coupling Facility, since the name can be changed via a CFM policy update. The physical CF information is the immutable identification information.
- **uint16 OperationalStatus [ ]**
  The current status of the CF (summarized from more granular CF state information):
  
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>OK</td>
</tr>
<tr>
<td>6</td>
<td>Error</td>
</tr>
<tr>
<td>9</td>
<td>Stopping</td>
</tr>
</tbody>
</table>
- **string StatusDescriptions [ ]**
  Strings describing the various Operational Status values.
- **string Status**
  A string indicating the current status
- **uint16 EnabledState**
  Indicates the Enabled or Disabled state.
- **string OtherEnabledState**
  String describing the Enabled State value
- **uint16 RequestedState**
  The last requested State
- **uint16 EnabledDefault**
  Indicates the default value for Enabled State
- **datetime TimeOfLastStateChange**
  The date and time Enabled State was last changed.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint64 IdentityToken [key]</td>
<td>The generated identity value for sysplex cluster.</td>
</tr>
<tr>
<td>string IdentityName [key]</td>
<td>The generated identity value for sysplex cluster.</td>
</tr>
<tr>
<td>string MachineType</td>
<td>Machine type of the server hosting the CF</td>
</tr>
<tr>
<td>string Manufacturer</td>
<td>Name of the manufacturer of the server hosting the CF</td>
</tr>
<tr>
<td>string ManufacturerPlant</td>
<td>The plant number where the machine was manufactured</td>
</tr>
<tr>
<td>string SerialNumber</td>
<td>A manufacturer assigned number to identify the server hosting the CF</td>
</tr>
<tr>
<td>uint8 LPARid</td>
<td>Platform-assigned ID of a logical partition in which the CF is running.</td>
</tr>
<tr>
<td>uint32 CFLevel</td>
<td>Facility operational (functionality) level</td>
</tr>
<tr>
<td>uint32 State</td>
<td>CF Operational State (summarized from more granular CF state information):</td>
</tr>
<tr>
<td></td>
<td>1 Okay</td>
</tr>
<tr>
<td></td>
<td>2 Unknown</td>
</tr>
<tr>
<td></td>
<td>6 Error</td>
</tr>
<tr>
<td></td>
<td>9 Stopping</td>
</tr>
<tr>
<td>uint16 NumberOfProcessors</td>
<td>Total number of CF processors</td>
</tr>
<tr>
<td>uint16 CPUUtilization</td>
<td>Percent CF processor utilization</td>
</tr>
<tr>
<td>uint32 FreeSpace</td>
<td>Currently unused storage available in the CF (in number of 4KB blocks)</td>
</tr>
<tr>
<td>uint32 TotalSpace</td>
<td>Total storage available in the CF (in number of 4KB blocks)</td>
</tr>
<tr>
<td>uint32 FreeDumpSpace</td>
<td>Currently unused allocated dump storage available in the CF (in number of</td>
</tr>
<tr>
<td></td>
<td>4KB blocks)</td>
</tr>
<tr>
<td>uint32 TotalDumpSpace</td>
<td>Total allocated dump storage available in the CF (in number of 4KB blocks)</td>
</tr>
<tr>
<td>uint32 StorageIncrementSize</td>
<td>Storage increment. The number of 4K blocks in a single storage increment</td>
</tr>
<tr>
<td>boolean Standalone</td>
<td>Coupling Facility Standalone indicator:</td>
</tr>
<tr>
<td></td>
<td>TRUE Not Standalone</td>
</tr>
<tr>
<td></td>
<td>FALSE Standalone</td>
</tr>
<tr>
<td>boolean Volatile</td>
<td>Indicator of whether this CF is volatile or nonvolatile (based on battery</td>
</tr>
<tr>
<td></td>
<td>backup or standby power source)</td>
</tr>
<tr>
<td>boolean CPUType</td>
<td>Indicates whether all of the CF processors are shared, or whether at least</td>
</tr>
<tr>
<td></td>
<td>one is dedicated:</td>
</tr>
<tr>
<td></td>
<td>TRUE All shared</td>
</tr>
<tr>
<td></td>
<td>FALSE Some are dedicated</td>
</tr>
<tr>
<td>boolean MaintenanceMode</td>
<td>Indicates whether the CF is currently in Maintenance mode:</td>
</tr>
<tr>
<td></td>
<td>TRUE Not in Maintenance mode</td>
</tr>
</tbody>
</table>
CF is in Maintenance mode

<table>
<thead>
<tr>
<th>boolean RecoveryMgrSite</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
</tr>
<tr>
<td>Recovery Manager is not active or the CF does not reside at the recovery site</td>
</tr>
<tr>
<td>FALSE</td>
</tr>
<tr>
<td>Recover Manager is active and the CF resides at the recovery site</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string SiteName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the SITE specified in the CFRM policy</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string CPCID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Facility's Central Processor Complex (CPC) ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string CFCCReleaseLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td>The release level of the CFCC code</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>string CFCCServiceLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td>The service level of the CFCC code</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>datetime CFCCCodeBuildDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>The date and time that the CFCC code was built</td>
</tr>
</tbody>
</table>

**Methods**

**uint32 StartCFMaintenanceMode()**

Sets the maintenance mode of the specified coupling facility to ON.

When a CF is in maintenance mode, the CF is not eligible for CF structure allocation purposes and all structure allocation processes will modify their CF selection processing accordingly.

**uint32 StopCFMaintenanceMode()**

Sets the maintenance mode of the specified coupling facility to OFF.

When a CF is no longer in maintenance mode, the CF is eligible for CF structure allocation purposes.

**Associations**

**IBMzOS_HostedCFStructure**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_CFStructure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_CouplingFacility</td>
</tr>
</tbody>
</table>

**IBMzOS_UsesCFs**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_SysplexNode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_CouplingFacility</td>
</tr>
</tbody>
</table>

**Indications**

**IBMzOS_CouplingFacilityInstCreation**

A 'life cycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been created.

**IBMzOS_CouplingFacilityInstDeletion**

A 'life cycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been deleted.
IBMzOS_CouplingFacilityInstModification

A 'life cycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been modified.
IBMzOS_CouplingFunction

Purpose
This class represents an abstraction of z/OS clustering capabilities. The clustering
capabilities are referred to as coupling functions, each serving a unique purpose in
a z/OS Systems Complex (Sysplex). Coupling functions are capabilities that are
facilitated through the use of:

- Couple datasets, which serve as repositories.
- Coupling facilities, which are used by z/OS systems to cache data structures,
  serialization structures and provide signaling capabilities to z/OS systems
  participating in a Sysplex.
- Cross-System Coupling Facility (XCF) software, which is a component of z/OS
  that provides functions to support cooperation between authorized programs
  running within a Sysplex.

Coupling functions include such capabilities as basic Sysplex support and
Coupling Facility Resource Manager (CFRM) support. There are other such
coupling functions supported by z/OS, though not all of them may be externalized
through CIM providers.

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is

IBMzOS_CouplingFunctionProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is

libcmpiIBMzOS_CouplingFunctionProvider.so

Properties
string Caption
A short description of the class.

string Description
A description of the class.

string ElementName
Name given to this instance of the class.

datetime InstallDate
A datetime value indicating when the object was
installed. A lack of a value does not indicate that
the object is not installed.

string Name [key]
Name of the coupling function

uint16 OperationalStatus [ ]
The current status of the SysplexCoupleDataset:

0 = Unknown
2 = OK
6 = Error
9 = Stopping

string StatusDescriptions [ ]
Strings describing the various Operational Status
values.

string Status
A string indicating the current status

uint16 EnabledState
Indicates the Enabled or Disabled state.

string OtherEnabledState
String describing the Enabled State value.
uint16 RequestedState
  The last requested State.

uint16 EnabledDefault
  Indicates the default value for Enabled State.

datetime TimeOfLastStateChange
  The date and time Enabled State was last changed.

uint32 Redundancy
  This identifies the level of couple dataset redundancy currently active for the coupling function.
  0  No couple datasets in use. The coupling function is not active.
  1  Primary couple dataset in use.
  2  Primary and alternate couple dataset are in use.

string ActivePolicyName
  Specifies the name of the active policy for the coupling function. Instances of coupling functions such as SYSPLEX, which have no policy, will have a null string value.

datetime TimeActivePolicyStarted
  The local date and time that the active policy was started.

Note: This property is in the local time of the operating system host servicing the request.

boolean isActive
  Identifies whether the coupling function is active.
  Coupling functions with no primary CDS are considered inactive.
  Coupling functions that support policies will be identified as active if they have a primary couple dataset in use, even if there is not active policy.

Methods

uint32 StartPolicy()
  This method activates (starts) the specified policy.
  The policy specified by the name parameter must be an administrative policy defined in the primary couple dataset currently in use by the coupling function.

uint32 StopPolicy()
  This method inactivates the currently active policy.
  For Coupling Facilities (CFs) or structures that are actively being used, not all aspects of the policy may become inactive immediately. These changes will become pending until the resources in question are no longer being used by programs operating in the Sysplex.

uint32 DeletePolicy()
  This method deletes the specified administrative policy.
  The policy specified by the name parameter must be an administrative policy defined in the primary couple dataset currently in use by the coupling function.

uint32 SwitchPrimary()
  This method makes the current in-use alternate
couple dataset the current primary couple dataset for the type represented by the coupling function instance.

The current in-use primary couple dataset at the time this method is invoked, upon successful completion of the method, will no longer be recognized by XCF and the coupling function instance will be operating solely with a primary couple dataset.

This method is similar to the z/OS operator command:

```
SETXCF COUPLE,TYPE=__,PSWITCH
```

**uint32 MakeAlternate()**

This method makes the specified couple dataset the current in use alternate couple dataset for the type represented by the coupling function instance.

The type of the specified couple dataset must be compatible with the coupling function instance for which the method was invoked.

The specified couple dataset must be a newly formatted couple dataset, formatted specifically for use in the Sysplex in which the coupling function instance exists. The method will fail if the specified couple dataset is currently or was previously active in the Sysplex.

The specified couple dataset may be one created using the Duplicate method or one created manually via the XCF couple dataset format utility (IXCL1DSU).

**uint32 Duplicate()**

This method duplicates the characteristics of the currently active primary couple dataset, for the type represented by the coupling function instance, to a new couple dataset. The name of the new couple dataset and the volume serial of the logical volume on which it will be allocated must be specified by the method invoker. The type of the couple dataset is determined by the coupling function instance.

**Associations**

**IBMzOS_UsesCouplingFunctions**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_Sysplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_CouplingFunction</td>
</tr>
<tr>
<td>see</td>
<td>page 231</td>
</tr>
</tbody>
</table>

**IBMzOS_UsesSysplexCoupleDatasets**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_CouplingFunction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_SysplexCoupleDataset</td>
</tr>
<tr>
<td>see</td>
<td>page 231</td>
</tr>
</tbody>
</table>

**IBMzOS_UsesCFRMCoupleDatasets**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_CouplingFunction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_CFRMCoupleDataset</td>
</tr>
<tr>
<td>see</td>
<td>page 231</td>
</tr>
</tbody>
</table>
IBMzOS_SFMAtributes

**Purpose**
An array of embedded instances of this class is used as input parameter to method SetSFMAtributes() (see “Methods” on page 222).

**Properties**

- **uint64 IdentityToken**
  Is the ‘IdentityToken’ of the SysplexNode whose SFM attributes are to be modified. The IdentityToken is a 64 bit unsigned integer that must be converted to a 20 character field, padded on the left with the character zero ('0'). An IdentityToken and IdentityName of '0' indicates that default values should be set for all SysplexNodes.

- **string IdentityName**
  Is the 'IdentityName' of the SysplexNode whose SFM attributes are to be modified. An IdentityToken and IdentityName of '0' indicates that default values should be set for all SysplexNodes.

- **boolean SetSystemWeight**
  Indicates that the SFM_Weight property should be updated.

- **boolean SetSystemSFMAction**
  Indicates that the SFM_Action (and possibly the SFM_Interval) property should be updated.

- **boolean SetMemStallTime**
  Indicates that the SFM stalled member action for the system should be updated.

- **boolean ResetMemStallTime**
  Indicates that the SFM stalled member action for the system should be cleared.

- **uint32 System_Weight**
  Is the new SFM weight value. The SFM weight is a 32 bit unsigned integer that must be converted to a 10 character field, padded on the left with the character zero ('0').

- **uint32 SFM_Action**
  Is the new SFM action value. Valid character values are:
  1  Prompt operator
  2  Isolate
  3  System reset
  4  Deactivate

- **uint32 SFM_Interval**
  Is the time in seconds corresponding to the SFM action. It is valid only when the action is being set to isolate (2), SystemReset (3), or Deactivate (4). The time is a 32 bit unsigned integer that must be converted to a 10 character field, padded on the left with the character zero ('0').

- **uint32 MemStallTime**
  Is the time in seconds that must pass before SFM takes action against a stalled member causing signal sympathy sickness.
IBMzOS_Sysplex

**Purpose**
This class represents a zSeries Sysplex (System Complex).

**Inheritance**
A subclasses is IBMzOS_SysplexNode (see “IBMzOS_SysplexNode” on page 225).

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_SysplexProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_SysplexProvider.so

**Properties**
- **string Caption**
  A short description of the class.
- **string Description**
  A description of the class.
- **string ElementName**
  Name given to this instance of the class.
- **datetime InstallDate**
  A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
- **string Name**
  Sysplex name
- **uint16 OperationalStatus [ ]**
  The current status of the Sysplex, based on the states of the systems it is comprised of:
  0  Unknown
  2  OK
  6  Error
  9  Stopping
- **string StatusDescriptions [ ]**
  Strings describing the various Operational Status values.
- **string Status**
  A string indicating the current status
- **uint16 EnabledState**
  Indicates the Enabled or Disabled state.
- **string OtherEnabledState**
  String describing the Enabled State value
- **uint16 RequestedState**
  The last requested State
- **uint16 EnabledDefault**
  Indicates the default value for Enabled State
- **datetime TimeOfLastStateChange**
  The date and time Enabled State was last changed.
- **uint64 IdentityToken [key]**
  The generated identity value for sysplex cluster. (Part 1 of 2)
- **string IdentityName [key]**
  The generated identity value for sysplex cluster. (Part 2 of 2)
- **uint32 Type**
  The type of sysplex cluster:
  1  Local
uint32 State

State of the Sysplex, based on the states of the systems it is comprised of:
1 Okay
2 Unknown
3 Error
4 Stopping

z/OS, will only report a state of ‘Okay’ (1)

boolean SysplexConnectionFail

Corresponds to the CONNFAIL attribute in the SFM policy. Indicates whether or not action taken when connectivity failure occurs in the sysplex.

Methods

uint32 SetSFMAttributes()

Updates the SFM policy to set the SFM weights for each system specified in the input, SystemArray, and will set the Sysplex Connect Fail property value for the sysplex.

Successful execution of this method will indicate that all the entries in the SystemArray were processed. If any of the system entries could not be processed the method will return an error.

An array of embedded instances of class IBMzOS_SFMAttributes is used as input parameter to this method (see “IBMzOS_SFMAttributes” on page 220).

uint32 SetSysplexConnFail()

Sets the ConnectionFail property value.

uint32 ResetSysplexConnFail()

Resets the ConnectionFail property value.

uint32 StartReallocate()

Analyzes all structures in the Sysplex and performs corrective actions on structures that are operating outside current CFRM policy parameters.

Sysplex Process Completion Indication will be generated when asynchronous processing has completed.

uint32 StopReallocate()

Stops the reallocation of CF structures.

Sysplex Process Completion Indication will be generated when asynchronous processing has completed.

uint32 ForceReallocate()

Forces an in process reallocation to be stopped.

Sysplex Process Completion Indication will be generated when asynchronous processing has completed.

Associations

IBMzOS_CollectionOfCFs

Source IBMzOS_Sysplex
Target IBMzOS_CouplingFacility
IBMzOS_CollectionOfSysplexNodes
Source IBMzOS_Sysplex
Target IBMzOS_SysplexNode
see page 229

IBMzOS_UsesCouplingFunctions
Source IBMzOS_Sysplex
Target IBMzOS_CouplingFunction
see page 231

Indications

IBMzOS_SysplexInstCreation
A 'life cycle' indication that indicates that an instance of the IBMzOS_Sysplex class has been created. The Sysplex supports services that may report on cluster manageable resources. This event occurs when each system has IPLed into the Sysplex with a Cluster capable Sysplex Couple Dataset. This event occurs on each system when a Cluster capable dataset has been brought into use.

IBMzOS_SysplexInstModification
A 'life cycle' indication that indicates that an instance of the IBMzOS_Sysplex class has been modified. The SysplexConnectionFail property has changed.

IBMzOS_Sysplex_CFROM_CDS_Initialized
A 'process' indication that indicates that the process of reallocating the CF Structures has completed. CFRM Resources (Coupling Facility, CF Structure and CF Structure Connectors) has been defined to the Sysplex. The z/OS Cluster MR Services should be issued to obtain the CFRM resource instances in use by the Sysplex.

IBMzOS_Sysplex_ReallocateInitiated
A 'process' indication that indicates that the Start Reallocate CF Structures process has been initiated. The reallocate command may have been initiated by an operator command or through a CIM StartReallocate() method.

IBMzOS_Sysplex_ReallocateCompleted
A 'process' indication that indicates that the Start, Stop, or Force Reallocate CF Structures command has completed processing. The reallocate command may have been initiated by an operator command or through a CIM StartReallocate(), StopReallocate(), or ForceReallocate() methods.
IBMzOS_SysplexCoupleDataset

Purpose
This class represents the z/OS Systems Complex (Sysplex) couple datasets. A Sysplex couple dataset contains Sysplex-wide data about systems, groups, and members that use Cross-System Coupling Facility (XCF) services. All z/OS systems in a Sysplex must have connectivity to the Sysplex couple dataset.

A Sysplex couple dataset can be the primary, or optionally, the active alternate couple dataset supporting the Sysplex coupling function. Minimally, a Sysplex couple dataset must be in use as the active primary Sysplex couple dataset for the Sysplex function to be active.

Inheritance
IBMzOS_CoupleDataset
    + IBMzOS_SysplexCoupleDataset

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_SysplexCoupleDatasetProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_SysplexCoupleDatasetProvider.so

Properties

string Name
The name of the couple dataset represented by an instance of this class.

uint32 NumberOfGroups
The number of XCF groups that the couple dataset is formatted to support. It is the maximum number of concurrently active XCF groups that can be active in the Sysplex while this couple dataset is in use as the primary Sysplex couple dataset.

uint32 NumberOfMembers
The number of XCF members per group that this couple dataset is formatted to support. Each XCF group in the Sysplex may have up to this number of concurrently active programs (XCF members) participating in the group.

uint32 GRSLevel
Indicates whether or not this couple dataset supports the use of Global Resource Serialization (GRS) STAR for Sysplex-scope resource serialization. GRS STAR provides improved performance and reliability over the use of GRS RING.
IBMzOS_SysplexNode

**Purpose**
This class represents a node in a zSeries Sysplex (System Complex). There is one node in a Sysplex for every z/OS system that comprises the Sysplex.

**Inheritance**
IBMzOS_Sysplex
← IBMzOS_SysplexNode

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_SysplexNodeProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_SysplexNodeProvider.so

**Properties**

string Caption
A short description of the class.

string Description
A description of the class.

string ElementName
Name given to this instance of the class.

datetime InstallDate
A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.

string Name
SysplexNode name which is the same as the Operating System's System Name

uint16 OperationalStatus [ ]
The current status of the SysplexNode:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>OK</td>
</tr>
<tr>
<td>6</td>
<td>Error</td>
</tr>
<tr>
<td>9</td>
<td>Stopping</td>
</tr>
</tbody>
</table>

string StatusDescriptions [ ]
Strings describing the various Operational Status values.

string Status
A string indicating the current status

uint16 EnabledState
Indicates the Enabled or Disabled state.

string OtherEnabledState
String describing the Enabled State value

uint16 RequestedState
The last requested State

uint16 EnabledDefault
Indicates the default value for Enabled State

datetime TimeOfLastStateChange
The date and time Enabled State was last changed.

uint64 IdentityToken [key]
The generated identity value for sysplex cluster. (Part 1 of 2)

string IdentityName [key]
The generated identity value for sysplex cluster. (Part 2 of 2)
uint32 State

State of node:
1  Okay
2  Unknown
3  Error
4  Stopping

uint32 SubState

SubState of node:
1  Normal
2  StatusUpdateMissing
3  InActive
4  IPLing

Valid when State = Error. Not valid for all other system states.

uint32 SystemSFMWeight

Corresponds to System Weight attribute on SFM policy. Relative system weight used by clique algorithm following Sysplex connectivity failure.

uint32 SystemFDIInterval

Corresponds to Failure Detection Interval attribute of SFM policy. Time interval during which missing status updates are tolerated. When failure interval is exceeded the SystemPartitionPolicy determines response.

uint32 SystemSFMAction

Corresponds to Action attribute on SFM policy. One of four actions are settable in the SFM policy:
1  Prompt Operator
2  Isolate (isolate system using the CF fencing controls)
3  System Reset Partition
4  Deactivate Partition (deactivate the partition using the HMC controls)

uint32 SystemSFMInterval

When the System SFM Action is Automatic, System Reset, or Deactivate, this property will contain the time value in seconds corresponding to the SFM action.

uint32 SystemMemStallTime

For MEMSTALLTIME(stalltime), SFM will take action to resolve a sympathy sickness problem attributed to a stalled XCF group member if the problem persists for stalltime seconds.

uint32 SystemOpNotify

The length of time after a system is status update missing before SFM takes action. For PROMPT, the interval used is the XCF OPNOTIFY value.

Methods

uint32 SetSystemFDIInterval()

Sets the SFM failure detection interval (FDI) for the system.

Associations

IBMzOS_HostedCFStrConnector

Source IBMzOS_SysplexNode
Target IBMzOS_CFStructureConnector

see page 230

IBMzOS_UsesCFs
Indications

IBMzOS_SysplexNodeInstCreation
A 'life cycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been created.

IBMzOS_SysplexNodeInstDeletion
A 'life cycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been deleted.

IBMzOS_SysplexNodeInstModification
A 'life cycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been modified.
Association IBMzOS_CFStrDependsOn

Purpose
This class associates an IBMzOS_CFStructure with an IBMzOS_CFStructureConnector.

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_CFStrDependsOnProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libcmpiIBMzOS_CFStrDependsOnProvider.so

Indications
IBMzOS_CFStrDependsOnInstCreation
A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStrDependsOn association class has been created.

IBMzOS_CFStrDependsOnInstDeletion
A 'life cycle' indication that indicates that an instance of the IBMzOS_CFStrDependsOn association class has been deleted.

Association IBMzOS_CollectionOfCFs

Purpose
This class associates an IBMzOS_Sysplex with an IBMzOS_CouplingFacility.

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_CollectionOfCFsProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libcmpiIBMzOS_CollectionOfCFsProvider.so

Indications
IBMzOS_CollectionOfCFsInstCreation
A 'life cycle' indication that indicates that an instance of the IBMzOS_CollectionOfCFs association class has been created.

IBMzOS_CollectionOfCFsInstDeletion
A 'life cycle' indication that indicates that an instance of the IBMzOS_CollectionOfCFs association class has been deleted.
Association IBMzOS_CollectionOfSysplexNodes

**Purpose**
This class associates an IBMzOS_Sysplex with an IBMzOS_SysplexNode.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_CollectionOfSysplexNodesProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_CollectionOfSysplexNodesProvider.so

**Indications**

**IBMzOS_CollectionOfSysplexNodesInstCreation**
A 'life cycle' indication that indicates that an instance of the IBMzOS_CollectionOfSysplexNodes association class has been created.

**IBMzOS_CollectionOfSysplexNodesInstDeletion**
A 'life cycle' indication that indicates that an instance of the IBMzOS_CollectionOfSysplexNodes association class has been deleted.

Association IBMzOS_HostedCFStructure

**Purpose**
This class associates an IBMzOS_CFStructure with an IBMzOS_CouplingFacility.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_HostedCFStructureProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_HostedCFStructureProvider.so

**Indications**

**IBMzOS_HostedCFStructureInstCreation**
A 'life cycle' indication that indicates that an instance of the IBMzOS_HostedCFStructure association class has been created.

**IBMzOS_HostedCFStructureInstDeletion**
A 'life cycle' indication that indicates that an instance of the IBMzOS_HostedCFStructure association class has been deleted.
**Association IBMzOS_HostedCFStrConnector**

**Purpose**
This class associates an IBMzOS_SysplexNode with an IBMzOS_CFStructureConnector.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_HostedCFStrConnectorProviderModule

**Provider library**
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_HostedCFStrConnectorProvider.so

**Indications**
- **IBMzOS_HostedCFStrConnectorInstCreation**
  A ‘life cycle’ indication that indicates that an instance of the IBMzOS_HostedCFStrConnector association class has been created.

- **IBMzOS_HostedCFStrConnectorInstDeletion**
  A ‘life cycle’ indication that indicates that an instance of the IBMzOS_HostedCFStrConnector association class has been deleted.

**Association IBMzOS_UsesCFs**

**Purpose**
This class associates an IBMzOS_SysplexNode with an IBMzOS_CouplingFacility.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_UsesCFsProviderModule

**Provider library**
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_UsesCFsProvider.so

**Indications**
- **IBMzOS_UsesCFsInstCreation**
  A ‘life cycle’ indication that indicates that an instance of the IBMzOS_UsesCFs association class has been created.

- **IBMzOS_UsesCFsInstDeletion**
  A ‘life cycle’ indication that indicates that an instance of the IBMzOS_UsesCFs association class has been deleted.
Association IBMzOS_UsesCFRMCoupleDatasets

**Purpose**
This class associates an instance of IBMzOS_CouplingFunction with instances of IBMzOS_CFRMCoupleDataset classes.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_UsesCFRMCoupleDatasetsProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_UsesCFRMCoupleDatasetsProvider.so

Association IBMzOS_UsesCFRMPolicies

**Purpose**
This class associates an instance of the IBMzOS_CFRMCoupleDataset class with instances of the IBMzOS_CFRMPolicy classes.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_UsesCFRMPoliciesProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_UsesCFRMPoliciesProvider.so

Association IBMzOS_UsesCouplingFunctions

**Purpose**
This class associates an instance of the IBMzOS_Sysplex class with instances of the IBMzOS_CouplingFunction classes.

Association IBMzOS_UsesSysplexCoupleDatasets

**Purpose**
This class associates an instance of the IBMzOS_CouplingFunction class with instances of the IBMzOS_SysplexCoupleDataset classes.

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_UsesSysplexCoupleDatasetsProviderModule
**Provider library**

The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is

```
libcmpiIBMzOS_UsesSysplexCoupleDatasetsProvider.so
```
Storage management classes

CIM_FCPort

Purpose
This class represents capabilities and management of a Fiber Channel Port device.

Inheritance
The z/OS specific subclass is IBMzOS_FCPort (see IBMzOS_FCPort on page 236).

CIM_FCPortStatistics

Inheritance
The z/OS specific subclass is IBMzOS_FCPortStatistics (see IBMzOS_FCPortStatistics on page 239).

CIM_PortController

Inheritance
The z/OS specific subclass is IBMzOS_PortController (see IBMzOS_PortController on page 241).

CIM_Product

Inheritance
The z/OS specific subclass is IBMzOS_Product (see IBMzOS_Product on page 243).

CIM_ProtocolEndpoint

Inheritance
The z/OS specific subclass is IBMzOS_SBProtocolEndpoint (see IBMzOS_SBProtocolEndpoint on page 244).

CIM_SoftwareIdentity

Inheritance
The z/OS specific subclass is IBMzOS_SoftwareIdentity (see IBMzOS_SoftwareIdentity on page 246).

CIM_StorageExtent

Inheritance
CIM_StorageExtent is supported as a superclass of IBMzOS_LogicalDisk (see IBMzOS_LogicalDisk on page 148) and won’t have a separate implementation.

CIM_ManagedElement
  + CIM_ManagedSystemElement
  + CIM_LogicalElement
  + CIM_EnabledLogicalElement
  + CIM_LogicalDevice
  + CIM_StorageExtent
**Used by the following CIM profiles**

- Host Discovered Resources Profile

**Association CIM_ControlledBy**

**Purpose**
The CIM_ControlledBy relationship indicates which devices such as IBMzOS_FCPort are controlled by a CIM_Controller such as IBMzOS_PortController on z/OS.

**Inheritance**
The z/OS specific subclass is IBMzOS_ControlledBy (see “Association IBMzOS_ControlledBy” on page 249).

**Association CIM_DeviceSAPImplementation**

**Inheritance**
The z/OS specific subclass is IBMzOS_SBDeviceSAPImplementation (see “Association IBMzOS_SBDeviceSAPImplementation” on page 256).

**Association CIM_ElementSoftwareIdentity**

**Inheritance**
The z/OS specific subclass is IBMzOS_ElementSoftwareIdentity (see “Association IBMzOS_ElementSoftwareIdentity” on page 252).

**Association CIM_ElementStatisticalData**

**Inheritance**
The z/OS specific subclass is IBMzOS_FCPortStatisticalData (see “Association IBMzOS_FCPortStatisticalData” on page 253).

**Association CIM_HostedAccessPoint**

**Inheritance**
The z/OS specific subclass is IBMzOS_SBHostedAccessPoint (see “Association IBMzOS_SBHostedAccessPoint” on page 257).

**Association CIM_InitiatorTargetLogicalUnitPath**

**Inheritance**
The z/OS specific subclass is IBMzOS_SBIInitiatorTargetLogicalUnitPath (see “Association IBMzOS_SBIInitiatorTargetLogicalUnitPath” on page 258).

**Association CIM_InstalledSoftwareIdentity**

**Inheritance**
The z/OS specific subclass is IBMzOS_InstalledSoftwareIdentity (see “Association IBMzOS_InstalledSoftwareIdentity” on page 254).
Association CIM_ProductElementComponent

Inheritance
The z/OS specific subclass is IBMzOS_ProductElementComponent (see "Association IBMzOS_ProductElementComponent" on page 255).

Association CIM_SystemDevice

Inheritance
The z/OS specific subclasses are
- IBMzOS_CSFCPort (see "Association IBMzOS_CSFCPort" on page 250) and
- IBMzOS_CSFCPortController (see "Association IBMzOS_CSFCPortController" on page 251).
IBMzOS.FCPort

Purpose
The IBMzOS.FCPort class defines the capabilities and management of a Fiber Channel Port device on z/OS.

Inheritance
CIM_ManagedElement
+ CIM_ManagedSystemElement
+ CIM_LogicalElement
+ CIM_EnabledLogicalElement
+ CIM_LogicalDevice
+ CIM_LogicalPort
+ CIM_NetworkPort
+ CIM_FCPort
+ IBMzOS.FCPort

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
IBMzOS.FCPortProviderModule

Provider library
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is
libcmpiIBMzOS.FCPortProvider.so

Used by the following CIM profiles
• Storage HBA profile

Properties
string Caption Returns IBM z/OS FCPort.
string Description Returns This is a z/OS FCPort.
string ElementName Returns LPARName:CSSID:CHPID, where
  LPARName is the name of the logical partition - empty
  if z/OS does not run in an LPAR
  CSSID is the channel subsystem ID
  CHPID is the channel path ID
string Name Returns LPARName:CSSID:CHPID, where
  LPARName is the name of the logical partition - empty
  if z/OS does not run in an LPAR
  CSSID is the channel subsystem ID
  CHPID is the channel path ID
uint16 OperationalStatus [] Returns the current status of the FCPort:
  2 OK
uint16 EnabledState
Returns
2 enabled

uint16 RequestedState
Returns
2 enabled

uint16 EnabledDefault
Indicates the administrator's default or startup configuration for the enabled state of an element. Always returns
2 enabled

string SystemCreationClassName
Indicates the system's CreationClassName.
Returns IBMzOS_ComputerSystem.

string SystemName
Displays the fully qualified host name of the system.

string CreationClassName
Indicates the name of the class or the subclass used in the creation of an instance.
Returns IBMzOS_FCPort.

string DeviceID
Displays a unique address or other identifying name for the LogicalDevice.
Returns
WWPN:LPARID
or
NodeDescriptor:LPARID
when the WWPN cannot be retrieved.

where
WWPN is the World Wide Port Number
LPARID is the logical partition ID
NodeDescriptor is the node descriptor

uint16 PortNumber
Returns the logical port number (CHPID).

uint64 Speed
Returns the bandwidth of the port in bits per second - 0 if z/OS does not run in an LPAR

uint64 MaxSpeed
Returns the maximum bandwidth of the port in bits per second - 0 if z/OS does not run in an LPAR

uint16 UsageRestriction
Returns
4 not restricted

uint16 PortType
Specifies the specific mode currently enabled for the port.
Returns
10 N-Port

uint16 LinkTechnology
Specifies the type of link.
Returns
string PermanentAddress
  Defines the network address of the port.
  Returns
  WWPN if a network address is available
  NULL else

uint64 SupportedMaximumTransmissionUnit
  Specifies the maximum transmission unit (MTU) that can be supported.
  Returns 8192.

uint64 ActiveMaximumTransmissionUnit
  Specifies the active or negotiated maximum transmission unit (MTU) that can be supported.
  Returns 8192.

uint16 SupportedCOS []
  Indicates the Fibre Channel Class of Service that is supported.
  Returns 3.

uint16 ActiveCOS []
  Indicates the Fibre Channel Class of Service that is active. Returns 3.

uint16 SupportedFC4Types []
  Indicates the supported Fibre Channel FC-4 protocol.
  Returns
  27 FC-SB-x channel

uint16 ActiveFC4Types []
  Indicates the currently running Fibre Channel FC-4 protocol.
  Returns
  27 FC-SB-x channel

Associations

IBMzOS_FCPortStatisticalData
  ManagedElement
  IBMzOS_FCPort
  Stats IBMzOS_FCPortStatistics
  see page 253

IBMzOS_ControlledBy
  Source IBMzOS_PortController
  Target IBMzOS_FCPort
  see page 249

IBMzOS_SBDeviceSAPImplementation
  Source IBMzOS_FCPort
  Target IBMzOS_SBProtocolEndpoint
  see page 256

IBMzOS_CSFCPort
  Source IBMzOS_ComputerSystem
  Target IBMzOS_FCPort
  see page 250
IBMzOS_FCPortStatistics

**Purpose**
The IBMzOS_FCPort class defines the statistics for the FCPort on z/OS.

**Inheritance**
CIM_ManagedElement
  ↳ CIM_StatisticalData
  ↳ CIM_NetworkPortStatistics
  ↳ CIM_FCPortStatistics
  ↳ IBMzOS_FCPortStatistics

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

IBMzOS_FCPortStatisticsProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

libcmpiIBMzOS_FCPortStatisticsProvider.so

**Used by the following CIM profiles**
• Storage HBA profile

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string Caption</td>
<td>Returns IBM z/OS FCPortStatistics.</td>
</tr>
<tr>
<td>string Description</td>
<td>Returns This is a z/OS FCPortStatistics.</td>
</tr>
<tr>
<td>string InstanceID</td>
<td>Returns <strong>IBM:FCPortStat:WWPN:LPARID</strong> or <strong>IBM:FCPortStat:NodeDescriptor:LPARID</strong> when the WWPN cannot be retrieved.</td>
</tr>
<tr>
<td>string ElementName</td>
<td>Returns <strong>FCPortStat:LPARName:CSSID:CHPID</strong>, where <strong>LPARName</strong> is the name of the logical partition - empty if z/OS does not run in an LPAR, <strong>CSSID</strong> is the channel subsystem ID, and <strong>CHPID</strong> is the channel path ID</td>
</tr>
<tr>
<td>uint64 BytesTransmitted</td>
<td>Returns the total number of bytes that are transmitted, including framing characters - 0 if z/OS does not run in an LPAR</td>
</tr>
<tr>
<td>uint64 BytesReceived</td>
<td>Returns the total number of bytes that are received, including framing characters - 0 if z/OS does not run in an LPAR</td>
</tr>
<tr>
<td>uint64 PacketsTransmitted</td>
<td>Returns the total number of packets that are transmitted - 0 if z/OS does not run in an LPAR</td>
</tr>
</tbody>
</table>
uint64 PacketsReceived

Returns the total number of packets that are received - 0 if z/OS does not run in an LPAR

Associations
IBMzOS_FCPortStatisticalData
  ManagedElement
  IBMzOS_FCPort
  Stats
  IBMzOS_FCPortStatistics
  see page 253
IBMzOS_PortController

**Purpose**
The IBMzOS_PortController class represents a logical device corresponding to a hardware network port controller on z/OS. Port controllers provide various features depending on their types and versions. Since it is not possible from inband z/OS instrumentation to distinguish between Ports and PortControllers, the PortController provider returns one instance for each FCPort, using the same key information.

**Inheritance**
- CIM_ManagedElement
  - CIM_ManagedSystemElement
  - CIM_LogicalElement
  - CIM_EnabledLogicalElement
  - CIM_LogicalDevice
  - CIM_Controller
  - CIM_PortController
  - IBMzOS_PortController

**Module name**
The module names of the CMPI providers that are registered for a CIM class which are used by the cimprovider command line tool for the administration of CMPI providers are
- IBMzOS_PortControllerProviderModule
- IBMzOS_PortControllerIndicationProviderModule

**Provider library**
The physical names of a CMPI provider's shared object library stored in the hierarchical file system are
- libcmpiIBMzOS_PortControllerProvider.so
- libcmpiIBMzOS_PortControllerIndicationProvider.so

**Used by the following CIM profiles**
- Storage HBA profile

**Properties**
- **string Caption** Returns IBM z/OS PortController.
- **string Description** Returns This is a z/OS PortController.
- **uint16 OperationalStatus []** Returns
  - 2 OK
- **uint16 EnabledState** Returns
  - 2 enabled
- **uint16 RequestedState** Returns
  - 2 enabled
- **uint16 EnabledDefault** Indicates the administrator's default or startup configuration for the enabled state of an element.
  Returns
  - 2 enabled
string SystemCreationClassName
Returns IBMzOS_ComputerSystem.

string SystemName
Displays the fully qualified host name of the system.

string CreationClassName
Returns IBMzOS_PortController.

string DeviceID
Returns a unique name for the logical device:

\[
\begin{align*}
\text{WWPN} & \rightarrow \text{LPARID} \\
& \quad \text{when the WWPN can be retrieved} \\
\text{NodeDescriptor} & \rightarrow \text{LPARID} \\
& \quad \text{else}
\end{align*}
\]

where

\[
\begin{align*}
\text{WWPN} & \quad \text{is the World Wide Port Number} \\
\text{LPARID} & \quad \text{is the logical partition ID} \\
\text{NodeDescriptor} & \quad \text{is the node descriptor}
\end{align*}
\]

uint16 ControllerType
Returns

4 FC

Associations

IBMzOS_ControlledBy

Source IBMzOS_PortController
Target IBMzOS_FCPort
see page 249

IBMzOS_ElementSoftwareIdentity

Source IBMzOS_SoftwareIdentity
Target IBMzOS_PortController
see page 252

Indications

CIM_InstCreation
A life cycle indication that indicates that an instance of the IBMzOS_PortController class has been created.

\[
\text{CIM\_IndicationFilter query string:} \\
\text{"SELECT * FROM CIM\_InstCreation} \\
\text{WHERE SourceInstance ISA CIM\_PortController"}
\]

CIM_InstDeletion
A life cycle indication that indicates that an instance of the IBMzOS_PortController class has been deleted.

\[
\text{CIM\_IndicationFilter query string:} \\
\text{"SELECT * FROM CIM\_InstDeletion} \\
\text{WHERE SourceInstance ISA CIM\_PortController"}
\]

For more information on how to subscribe to an indication, see "CIM subscription mechanism" on page 289. Specify your queries using the CIM\_IndicationFilter query string (see also "CIM\_IndicationFilter" on page 290).
IBMzOS_Product

Purpose
The IBMzOS_Product is a concrete class that aggregates PhysicalElements, software (SoftwareIdentity and SoftwareFeatures), services or other products on z/OS.

For z/OS 1.12, an instance of IBMzOS_Product is created for each FCPort returned by the IBMzOS_FCPort provider.

Inheritance
CIM_ManagedElement
  ↪ CIM_Product
  ↪ IBMzOS_Product

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
  IBMzOS_ProductProviderModule

Provider library
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is
  libcmpiIBMzOS_ProductProvider.so

Used by the following CIM profiles
• Storage HBA profile

Properties
string Caption  Returns IBM z/OS Product.
string Description  Returns Represents a z/OS FCPortController Product.
string Name  Returns the DeviceID from IBMzOS_FCPort.
string ElementName  Returns the DeviceID from IBMzOS_FCPort.
string IdentifyingNumber  Returns the DeviceID from IBMzOS_FCPort.
string Vendor  Returns IBM.
string Version  Returns unknown.
IBMzOS_SBProtocolEndpoint

Purpose
The IBMzOS_SBProtocolEndpoint class is used to represent two different entities, Initiator and Target. The Initiator entity describes the protocol endpoint on the computer system side, the target entity describes the protocol endpoint on the disk controller side of a logical disk attached to a computer system.

Protocol endpoints are identified via World Wide Port Numbers (WWPN), which are used as the primary key for the instances of the class IBMzOS_SBProtocolEndpoint, reflected in the name property. For the retrieval of WWPN the IOS services IOSCDR and IOSCHPD have been extended for z/OS 1.13 to facilitate the retrieval of WWPN for the Initiator (IOSCHPD) and Target (IOSCDR) protocol endpoints. Therefore, the retrieval of WWPN through IOSCDR is only possible under the following conditions:
1. The used hardware is at least an IBM System z10™.
2. The requestor or CIM client has UPDATE access to the IOSCDR profile.

Inheritance
CIM_ManagedElement
  + CIM_ManagedSystemElement
    + CIM_LogicalElement
      + CIM_EnabledLogicalElement
        + CIM_ServiceAccessPoint
          + CIM_ProtocolEndpoint
            + IBMzOS_SBProtocolEndpoint

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_SBProtocolEndpointProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libcmpiIBMzOS_SBProtocolEndpointProvider.so

Used by the following CIM profiles
- Host Discovered Resources Profile
- Storage HBA profile

Properties
string Caption
  Returns IBM z/OS SBProtocolEndpoint.
string Description
  Returns This is a z/OS SBProtocolEndpoint.
string Name
  The Initiator returns the WWPN of the computer system side.
  The Target returns the WWPN of the storage controller side.
uint16 OperationalStatus []
  Returns
  2  OK
uint16 EnabledState
  Returns
  2   Enabled

uint16 RequestedState
  Returns
  2   Enabled

uint16 EnabledDefault
  Returns
  2   Enabled

string SystemCreationClassName
  Returns IBMzOS_ComputerSystem

string SystemName
  Displays the name of the host system.

string CreationClassName
  Returns IBMzOS_SBProtocolEndpoint

uint16 ProtocolIFType
  Returns
  56 Fibre Channel

string OtherTypeDescription
  Returns SB.

uint16 ConnectionType
  Returns
  2   Fibre Channel

uint16 Role
  Returns
  2   Initiator
  
or
  3   Target

Associations

IBMzOS_SBHostedAccessPoint
  Source IBMzOS_ComputerSystem
  Target IBMzOS_SBProtocolEndpoint (Initiator Instance)
  see page 257

IBMzOS_SBDeviceSAPImplementation
  Source IBMzOS_FCPort
  Target IBMzOS_SBProtocolEndpoint
  see page 256

IBMzOS_SBInitiatorTargetLogicalUnitPath
  Initiator IBMzOS_SBProtocolEndpoint (Initiator Instance)
  Target IBMzOS_SBProtocolEndpoint (Target instance)
  LogicalUnit IBMzOS_LogicalDisk
  see page 258
IBMzOS_SoftwareIdentity

Purpose
The IBMzOS_SoftwareIdentity class provides descriptive information about a software component for asset tracking or installation dependency management.

The idea behind SoftwareIdentity as defined in the SMI-S Storage HBA profile does not match the concepts of z/OS. Therefore this class has only been implemented for formal compliance with the SMI-S Storage HBA profile.

For z/OS 1.12, therefore only one instance of IBMzOS_SoftwareIdentity is created and associated to all PortControllers.

Inheritance
CIM_ManagedElement
  + CIM_ManagedSystemElement
  + CIM_LogicalElement
  + CIM_SoftwareIdentity
  + IBMzOS_SoftwareIdentity

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is

  IBMzOS_SoftwareIdentityProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

  libcmpiIBMzOS_SoftwareIdentityProvider.so

Used by the following CIM profiles
• Storage HBA profile

Properties
string Caption
  Returns IBM z/OS SoftwareIdentity.

string Description
  Returns The Software driving the IBMzOS_PortController.

uint16 OperationalStatus []
  Returns
  2   OK

string InstanceID
  Uniquely identifies an instance of this class.
  Returns IBMzOS:CSSID:LPARID, where
  CSSID  is the channel subsystem ID
  LPARID is the logical partition ID

string ElementName
  Returns IBMzOS:CSSID:LPARID, where
  CSSID  is the channel subsystem ID
  LPARID is the logical partition ID

string VersionString
  Returns the z/OS Version and Release number in the form Major.Minor.Revision, where
  Major  is the z/OS version
  Minor  is the release
Revision is the revision number

string Manufacturer
Returns IBM.

uint16 Classifications []
Returns
2 Driver
and
8 Operating System

string TargetOperatingSystems []
Returns z/OS.
### Associations

<table>
<thead>
<tr>
<th>IBMzOS_ElementSoftwareIdentity</th>
<th>Source</th>
<th>IBMzOS_SoftwareIdentity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_ProtController</td>
<td></td>
</tr>
<tr>
<td>see</td>
<td>see page 252</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IBMzOS_InstalledSoftwareIdentity</th>
<th>Source</th>
<th>IBMzOS_ComputerSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_SoftwareIdentity</td>
<td></td>
</tr>
<tr>
<td>see</td>
<td>see page 254</td>
<td></td>
</tr>
</tbody>
</table>
**Association IBMzOS_ControlledBy**

**Inheritance**
- CIM_Dependency
- CIM_DeviceConnection
- CIM_ControlledBy
- IBMzOS_ControlledBy

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is

- IBMzOS_ControlledByProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

- libcmpiIBMzOS_ControlledByProvider.so

**Used by the following CIM profiles**
- Storage HBA profile

**Properties**
- **Ref Antecedent**
  References an IBMzOS_PortController
- **Ref Dependent**
  References an IBMzOS_FCPort
- **Uint16 AccessState**
  Returns
  - 1 Active
- **String DeviceNumber**
  Returns the device number of the IBMzOS_FCPort.
- **Uint16 AccessMode**
  Returns
  - 2 ReadWrite
Association IBMzOS_CSFCPort

Inheritance
CIM_Component
  ⊆ CIM_SystemComponent
  ⊆ CIM_SystemDevice
  ⊆ IBMzOS_CSFCPort

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
  IBMzOS_CSFCPortProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
  libcmpiIBMzOS_CSFCPortProvider.so

Properties
Ref GroupComponent References an IBMzOS_ComputerSystem
Ref PartComponent References an IBMzOS_FCPort
Association IBMzOS_CSFCPortController

**Inheritance**
- CIM_Component
  - CIM_SystemDevice
  - IBMzOS_CSFCPortController

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_CSFCPortControllerProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_CSFCPortControllerProvider.so

**Properties**
- **Ref GroupComponent** References an IBMzOS_ComputerSystem
- **Ref PartComponent** References an IBMzOS_PortController
Association IBMzOS_ElementSoftwareIdentity

**Purpose**
The IBMzOS_ElementSoftwareIdentity class allows a Managed Element such as an IBMzOS_PortController to report its software related asset information (such as firmware, drivers, or configuration software) on z/OS.

**Inheritance**
- CIM_Dependency
  - CIM_ElementSoftwareIdentity
  - IBMzOS_ElementSoftwareIdentity

**Used by the following CIM profiles**
- Storage HBA profile

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_ElementSoftwareIdentityProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_ElementSoftwareIdentityProvider.so

**Properties**
- **Ref Antecedent** References an IBMzOS_SoftwareIdentity
- **Ref Dependent** References an IBMzOS_PortController
Association IBMzOS_FCPortStatisticalData

**Purpose**
This class associates an IBMzOS_FCPort with IBMzOS_FCPortStatistics.

**Inheritance**
- CIM_ElementStatisticalData
  - IBMzOS_FCPortStatisticalData

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is
- IBMzOS_FCPortStatisticalDataProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_FCPortStatisticalDataProvider.so

**Properties**
- **Ref ManagedElement**: References an IBMzOS_FCPort
- **Ref Stats**: References IBMzOS_FCPortStatistics
Association IBMzOS_InstalledSoftwareIdentity

Purpose
The IBMzOS_InstalledSoftwareIdentity association identifies the Software installed on a system. On z/OS this class has only been implemented for formal compliance with the SMI-S Storage HBA profile and is of limited use.

Inheritance
- CIM_InstalledSoftwareIdentity
  - IBMzOS_InstalledSoftwareIdentity

Used by the following CIM profiles
- Storage HBA profile

Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_InstalledSoftwareIdentityProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_InstalledSoftwareIdentityProvider.so

Properties
- Ref System References an IBMzOS_ComputerSystem
- Ref InstalledSoftware References an IBMzOS_SoftwareIdentity
Association IBMzOS_ProductElementComponent

Inheritance

- CIM_Component
- CIM_ProductElementComponent
- IBMzOS_ProductElementComponent

Used by the following CIM profiles

- Storage HBA profile

Module name

The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is

- IBMzOS_ProductElementComponentProviderModule

Provider library

The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is

- libcmpiIBMzOS_ProductElementComponentProvider.so

Properties

Ref GroupComponent References an IBMzOS_Product
Ref PartComponent References an IBMzOS_PortController
Association IBMzOS_SBDeviceSAPImplementation

**Purpose**
The IBMzOS_SBDeviceSAPImplementation class describes an association between a ServiceAccessPoint (SAP) and how it is implemented.

**Inheritance**
CIM.Dependency
- CIM_DeviceSAPImplementation
- IBMzOS_SBDeviceSAPImplementation

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the `cimprovider` command line tool for the administration of CMPI providers is

- IBMzOS_SBDeviceSAPImplementationProviderModule

**Provider library**
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is

- libcmpiIBMzOS_SBDeviceSAPImplementationProvider.so

**Used by the following CIM profiles**
- Storage HBA profile

**Properties**
- Ref Antecedent: References an IBMzOS_FCPort
- Ref Dependent: References an IBMzOS_SBProtocolEndpoint
Association IBMzOS_SBHostedAccessPoint

**Purpose**
The IBMzOS_SBHostedAccessPoint class is an association between a Service Access Point and the System on which it is provided.

**Inheritance**
- CIM_Dependency
  - CIM_HostedDependency
  - CIM_HostedAccessPoint
  - IBMzOS_SBHostedAccessPoint

**Module name**
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is
- IBMzOS_SBHostedAccessPointProviderModule

**Provider library**
The physical name of a CMPI provider’s shared object library as it is stored in the hierarchical file system is
- libcmpiIBMzOS_SBHostedAccessPointProvider.so

**Used by the following CIM profiles**
- Host Discovered Resources Profile
- Storage HBA profile

**Properties**
- **Ref Antecedent** References an IBMzOS_ComputerSystem
- **Ref Dependent** References an IBMzOS_SBProtocolEndpoint (Initiator instance)
Association IBMzOS_SBInitiatorTargetLogicalUnitPath

Purpose
The IBMzOS_SBInitiatorTargetLogicalUnitPath class is a three way association
between an z/OS disk device, identified by the LogicalUnit reference, the channel,
identified by the Initiator reference and the control unit, identified by the Target
reference. Each permutation of initiator and target ProtocolEndpoints and logical
units is considered as a separate path.

Retrieving the data for IBMzOS_SBInitiatorTargetLogicalUnitPath is only possible
under the following conditions:
1. The used hardware is at least an IBM System z10.
2. The requestor or CIM client user ID has UPDATE access to the IOSCDR profile.

Inheritance
CIM_InitiatorTargetLogicalUnitPath
+ IBMzOS_SBInitiatorTargetLogicalUnitPath

Module name
The module name of the CMPI provider that is registered for a CIM class which is
used by the cimprovider command line tool for the administration of CMPI
providers is
IBMzOS_SBInitiatorTargetLogicalUnitPathProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the
hierarchical file system is
libcmpiIBMzOS_SBInitiatorTargetLogicalUnitPathProvider.so

Used by the following CIM profiles
- Host Discovered Resources Profile
- Storage HBA profile

Properties
Ref Initiator References an IBMzOS_SBProtocolEndpoint (Initiator instance)
Ref Target References an IBMzOS_SBProtocolEndpoint (Target instance)
Ref LogicalUnit References an IBMzOS_LogicalDisk

uint32 State Returns the state of the path:
2 active
4 disabled
8 removed (boxed)
9 transitioning

Indications
CIM_InstCreation
A life cycle indication that indicates that an instance of the
IBMzOS_SBInitiatorTargetLogicalUnitPath class has been created.

CIM_IndicationFilter query string:
"SELECT * FROM CIM_InstCreation
WHERE SourceInstance ISA
CIM_InitiatorTargetLogicalUnitPath"
CIM_InstModification

A life cycle indication that indicates a path state change of an instance of the IBMzOS_SBInitiatorTargetLogicalUnitPath class.

CIM_IndicationFilter query string:

```
*SELECT * FROM CIM_InstModification
WHERE SourceInstance ISA CIM_InitiatorTargetLogicalUnitPath AND
SourceInstance.CIM_InitiatorTargetLogicalUnitPath::State
<>
PreviousInstance.CIM_InitiatorTargetLogicalUnitPath::State
```

CIM_InstDeletion

A life cycle indication that indicates that an instance of the IBMzOS_SBInitiatorTargetLogicalUnitPath class has been deleted.

CIM_IndicationFilter query string:

```
*SELECT * FROM CIM_InstDeletion
WHERE SourceInstance ISA CIM_InitiatorTargetLogicalUnitPath
```

For more information on how to subscribe to an indication, see "CIM subscription mechanism" on page 289. Specify your queries using the CIM_IndicationFilter query string (see also "CIM_IndicationFilter" on page 290).
Chapter 15. WLM classes

Figure 13 shows the relationship between the IBM extension classes, the IBM extension classes for WLM, and the CIM classes that they extend. The DMTF website provides a detailed description of the CIM classes. The z/OS-specific classes are described in detail in the following chapters.

Figure 13. WLM classes

Figure 14 on page 262 shows a process indication that indicates that a service policy has been activated in the sysplex. This event occurs on each system in the sysplex.
IBMzOS_WLM

Purpose
This class represents the z/OS Workload Manager.

Before you can access this class, be sure that you have prepared the security steps as described in "Setting up the CIM server for WLM management" on page 40.

- Grant the requestor's user ID READ access to the RACF facility class MVSADMIN.WLM.POLICY
- If your environment requires program control, be sure that library BLSUXTID in SYS1.MIGLIB is program controlled.

Example:
RDEFINE PROGRAM BLSUXTID
RALT PROGRAM BLSUXTID ADDMEM('SYS1.MIGLIB'/******'/NOPADCHK) +
UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH

Inheritance
CIM_ManagedElement
  + CIM_ManagedSystemElement
  + CIM_LogicalElement
Module name
The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_WLMProviderModule

Provider library
The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libwlmprovider.so

Owning component
The z/OS component which owns the CMPI provider is WLM

Properties
string Caption A short description of the class
string Description A description of the class
string ElementName Name given to this instance of the class
datetime InstallDate Not supported
uint16 OperationalStatus[] The current status of WLM:
[2] [OK]
string StatusDescriptions[] Not supported
string Status Not supported
uint16 HealthState The health status of WLM:
5 OK
uint16 EnabledState Indicates the Enabled or Disabled state:
2 Enabled
string OtherEnabledState Not supported
uint16 RequestedState The last requested state:
2 Enabled
uint16 EnabledDefault Indicates the default value for Enabled State:
2 Enabled
datetime TimeOfLastStateChange Not supported
string SystemCreationClassName [key] The scoping system’s CreationClassName
string SystemName [key] The name of the scoping system
string CreationClassName [key] Indicates the name of the class used in the creation of an instance
string Name [key]  Name of z/OS Workload Management service
string PrimaryOwnerName  Not supported
string PrimaryOwnerContact  Not supported
boolean Started  Indicates if z/OS WLM runs
string ActiveServicePolicy  Name of WLM service policy activated for the sysplex
string PolicyDescription  Description of the WLM service policy activated for the sysplex
datetime PolicyActivationTimestamp  The time the WLM service policy has been activated
string PolicyActivationUser  Userid that activated the WLM service policy
string PolicyActivationSystem  System from which the WLM service policy activation was triggered
string RelatedServiceDefinition  Name of the service definition the WLM service policy was activated from
datetime ServiceDefinitionInstallationTimestamp  Time the service definition was installed
string ServiceDefinitionInstallationUser  User that installed the service definition
string ServiceDefinitionInstallationSystem  System from which the service definition installation was triggered
uint8 ServiceDefinitionFunctionalityLevel  Functionality level of the service definition
string EmbeddedEWLMPolicy  Name of the EWLM policy embedded in the active WLM service policy
datetime EWLMDMPolicyActivationTimestamp  Time the EWLM Domain Manager has triggered the activation of the EWLM policy that is activated on this system
datetime EWLMPolicyActivationTimestamp  Time the EWLM Managed Server has activated the EWLM policy that is activated on this system
datetime EWLMManagementActivationTimestamp  Time when management towards EWLM goals has been activated on this system
boolean PolicyActivationInProgress  Indicates whether a WLM policy activation is currently in progress
boolean AbnormalSystemConfiguration  Indicates an abnormal system configuration
string PolicyActivatingSystem
If a WLM policy activation is currently in progress, the name of the system where the policy activation was triggered

uint8 WLMVersion
WLM version

uint16 CDSFormat
WLM Couple Dataset format

string SysplexMembersSystemName[]
Name of systems in sysplex

uint8 SysplexMembersWLMMode[]
Workload management mode of systems in sysplex:
0 Undefined
1 Compatibility Mode
2 Goal Mode
3 EWLM Mode

uint8 SysplexMembersWLMStatus[]
Workload management status of systems in sysplex:
0 Undefined
1 Initializing
2 Active
3 Active, Not Running with Active Policy
4 Quiesce in Progress
5 Cleanup Initiated by System
6 WLM Inactive, Cleanup Complete
7 Unknown
8 System Inactive, Cleanup Pending
9 System Inactive, Cleanup Complete
10 Unknown

uint8 SysplexMembersGPAMStatus[]
Guest platform management provider (GPMP) status of systems in sysplex:
0 PgmError
1 Inactive
2 Started
3 Active
4 Connected
5 Shutdown1
6 Shutdown2
7 Shutdown3
8 Failed
9 Stopped
10 SevFailed
11 Early-IPL
12 Disabled
13 Unavailable
14 Unknown

string SysplexMembersActivePolicy[]
Name of WLM service policy active on systems in sysplex
datetime SysplexMembersPolicyActivationTimestamp[]
  Time the WLM service policy was activated on systems in sysplex

string SysplexMembersCleaningSystem[]
  If WLM state is ‘Cleanup Initiated by System’, the name of the system performing the cleanup

string CouplingFacilityStructureNames[]
  Name of the WLM coupling facility structures

uint8 CouplingFacilityStructureStatus[]
  Status of the WLM coupling facility structures:
  0  Disconnected
  1  Connected

Methods

uint32 RequestStateChange()  Not supported
uint32 StartService()        Not supported
uint32 StopService()         Not supported
uint32 ActivateServicePolicy()
  Activate a service policy contained in the WLM service definition installed in the WLM couple dataset. UPDATE access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method. Successful execution of this method is indicated by an IBMzOS_WLMPolicyActivationIndication indication.

uint32 InstallServiceDefinition()
  Install the passed service definition to the WLM couple dataset. UPDATE access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method.

uint32 ExtractServiceDefinition()
  Extract the service definition from the WLM couple dataset. READ access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method.

uint32 UploadServiceDefinition()
  Save service definition in XML format in a sequential dataset.

uint32 DownloadServiceDefinition()
  Download a service definition that is stored in XML format in a sequential dataset.

Indications

IBMzOS_WLMPolicyActivationIndication
  A ‘process’ indication that indicates that a service policy has been activated in the sysplex. This event occurs on each system in the sysplex.
**Associations**

**IBMzOS_WLMOS**

<table>
<thead>
<tr>
<th>Source</th>
<th>IBMzOS_WLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>IBMzOS_ComputerSystem</td>
</tr>
</tbody>
</table>

**see**

page 267

---

**Association IBMzOS_WLMOS**

**Purpose**

This class associates an IBMzOS_WLM with an IBMzOS_ComputerSystem.

**Inheritance**

CIM_Dependency

+ CIM_HostedDependency

+ CIM_HostedService

+ IBMzOS_WLMOS

**Module name**

The module name of the CMPI provider that is registered for a CIM class which is used by the cimprovider command line tool for the administration of CMPI providers is IBMzOS_WLMOSProviderModule

**Provider library**

The physical name of a CMPI provider's shared object library as it is stored in the hierarchical file system is libiwmOSProvider.so

**Owning component**

The z/OS component which owns the CMPI provider is WLM
Part 5. Developer's guide
Chapter 16. CMPI provider development for z/OS

The system-specific management data for the CIM Schema and system-specific Schema extension classes are provided through management instrumentation. While some management instrumentation is already provided by z/OS CIM (see Chapter 14, “z/OS Management Instrumentation for CIM,” on page 121), it is also possible to develop additional management instrumentation for other z/OS resources which are not accessible through the existing z/OS management instrumentation.

You can implement management instrumentation by developing a provider. A provider is a dynamic load library that implements a given interface and contains the program code used by the CIM server to interact with the system resource described by a certain CIM class, for example CIM_Processor. Providers are registered with the CIM server for a defined CIM class, allowing the CIM server to route all client requests directed against this class to the provider for interacting with the resource. A provider logically acts as an extension of the CIM server for interfacing directly with the managed resources.

Providers are the de facto standard concept for developing management instrumentation, though this purpose of providers is not explicitly mentioned by the various CIM and WBEM standards available from the DMTF. The Common Manageability Programming Interface (CMPI) technical standard was defined by The Open Group to allow for developing providers independently from a specific CIM server implementation.

Figure 15 shows the CMPI provider interfaces:

![Figure 15. CMPI provider interfaces](image)

CMPI is a C-based programming interface for providers designed for binary compatibility. All management instrumentation included with the z/OS CIM server was developed following the CMPI standard. CMPI is the only supported provider programming interface for the z/OS CIM server. Documentation about the CMPI Technical Standard is available from The Open Group and is not repeated in any documentation available for z/OS. Developers of management instrumentation for
z/OS need to be familiar with the CMPI and CIM/WBEM standards. The information contained here explains the specific aspects that need to be considered for developing CMPI providers for z/OS:

__1. Obtain the required header files__
To be able to develop a CMPI provider for z/OS, a set of C header files is required that define the CMPI interface. Due to legal implications with the OpenSource nature of these files, they are not provided together with z/OS CIM, but must be obtained from their original location at The Open Group instead.

Due to the CMPI interface design, you need not link a CMPI provider to any library of the z/OS CIM server. Only the header files are needed for developing a CMPI provider library.

See “Obtaining the required header files” for more information.

__2. Follow general aspects of developing a provider__
(see “Following general aspects of developing a provider” on page 273)

__3. Expose a provider initialization and function signatures__
(see “Preparing provider initialization and function signatures” on page 274)

__4. Consider security aspects__
(see “Planning provider security” on page 275)

__5. Convert EBCDIC provider data into UTF-8__
(see “Converting data to ASCII, EBCDIC andUTF-8” on page 276)

__6. Follow the guidelines for installing third-party providers__
(see “Provider installation” on page 276)

__7. Register the provider with the CIM server__
(see “Registering a provider with the CIM server” on page 277)

__8. Optionally use the out-of-process support for providers__
(see “Using the out-of-process support for providers” on page 285)

---

**Obtaining the required header files**

Before you can start to develop a provider dynamic load library, you must obtain the following C header files from the OpenPegasus project through the internet:

- `cmpidt.h`: Data type definitions
- `cmpift.h`: Function signature definitions in the form of function tables
- `cmpimacs.h`: CMPI convenience macros (optional)

These files are available in the OpenPegasus CVS Repository. Users familiar with CVS can check out these files using a CVS client on any platform by following the instructions on the [OpenPegasus website](https://openpegasus.org) in the "CVS Overview" section. The required files are located in directory `pegasus/src/Pegasus/Provider/CMPI`. To get the correct version of the files, they need to be checked out with at least the `RELEASE_2.8.1` tag.

If you are not familiar with using CVS, obtain the files through a web browser starting at the [OpenPegasus website](https://openpegasus.org). You can navigate from the "Web CVS" section to the required CMPI files by clicking on the following directory names (see also Figure 16 on page 273):

`pegasus/src/Pegasus/Provider/CMPI`
Once you have successfully navigated to the CMPI directory, the required header files are at the end of the list of displayed files. To get the correct version of the files, select the tag RELEASE_2_8_1 or above from the list.

To download the files, first click on the version number displayed in the column after each file name and then select download on the next screen where the content of the file is displayed. Once you have successfully downloaded the files, transfer them to the z/OS system on which the provider dynamic load library will be developed, ideally to a ZFS directory. Please note that when transferring files from the workstation to a z/OS system, they should be converted from ASCII to EBCDIC encoding.

There are also a couple of samples for CMPI providers available on the OpenPegasus CVS Repository. They can be obtained the same way as the header files by navigating to the pegasus/src/Providers/sample/CMPI directory.

### Following general aspects of developing a provider

Before you can start to develop a CMPI provider, you first need to have the CIM class model containing descriptions for the resource to be instrumented in the form of a CIM class. Follow the WBEM standards and in particular be consistent with the CIM Schema supported by the CIM server when you develop the CIM class. Usually, a CIM class for which a provider is written, is derived from one of the classes in the CIM Schema provided by the DMTF, and named with a vendor-specific class name prefix. For example, the prefix "IBMzOS_" is used for all classes provided by IBM for the z/OS operating system. This naming scheme also helps to prevent conflicts with the resources that have already been instrumented for CIM by IBM or other vendors.

**Note:** In general it is not recommended to create new providers for resources that have already been instrumented by IBM.
Preparing provider initialization and function signatures

The nature of a CMPI provider does not require static linking to any of the CIM server's libraries. Instead, for each provider function group a single initialization routine (factory) entry point must be exposed following a defined naming scheme, so that the CIM server can call this entry point by name once it has dynamically loaded a provider dynamic load library. The CIM server will attempt to determine the function groups supported by a provider and the respective entry points by verifying the existence of the according provider factory entry points.

The signature for the factory functions looks like this:

```c
CMPI<mi-type>MI * <mi-name>_Create_<mi-type>MI(CMPIBroker*,
                                          CMPIContext*,
                                          CMPIService*);
```

where `<mi-type>` refers to the function group of the provider, and `<mi-name>` refers to the actual provider name as specified during provider registration.

**Important:**
The actual signature of this function has an additional '_' after '_Create', which is not described as such in the initial version of the CMPI Technical Standard, but is changed in a corrigendum to match the existing implementations of the CMPI interface.

The factory function must return a pointer to a valid `CMPI<mi-type>MI` structure, where the major component of this structure is the table holding the function pointers, and thus enabling access to the individual provider group functions for the CIM server. An example of such a function pointer is the pointer to the `enumerateInstances` function in the `CMPIInstanceMI` structure.

The function groups for CMPI providers are `Instance, Association, Property, Method or Indication`, where type `Property` is not supported by the z/OS CIM server.

In file `cmpimacs.h`, a set of C preprocessor macros is defined that you may use for the provider initialization code and through which the required code for the `<mi-name>_Create_<mi-type>`MI function is generated in a convenient way. These macros are called `CM<mi-type>MIStub` and they are used in many of the examples referenced in "Samples" on page 285.

For further details refer to "MI Factories" in CMPI Technical Standard Document provided by The Open Group.

For each of the CMPI provider function groups, a set of C functions must be implemented as described in "MI Function Signatures" of the CMPI Technical Standard Document.

**Instance provider functions**

Instance providers are the most common kind of management instrumentation. They implement the basic access to the resources described in a CIM class. With an instance provider it is possible to create, enumerate, modify, delete, query or simply retrieve system resources:

- `cleanup(...)`
- `enumInstanceNames(...)`
Method provider functions
Method providers are needed to implement the methods defined for a CIM class.
- cleanup(...)
- invokeMethod(...)

Association provider functions
Association providers are needed to implement the relationships between system resources as defined by the association classes.
- cleanup(...)
- Associators(...)
- AssociatorNames(...)
- References(...)
- ReferenceNames(...)

Indication provider functions
Event or indication providers must be implemented for event subscription and notification:
- cleanup(...)
- AuthorizeFilter(...)
- MustPoll(...)
- ActivateFilter(...)
- DeActivateFilter(...)
- EnableIndications(...)
- DisableIndications(...)

Please note that the function MustPoll is not supported for z/OS.

Planning provider security
When developing a CMPI provider for z/OS, consider the security context in which the provider runs. Besides the levels of security provided by the z/OS CIM server for authentication and authorization, a provider is processed in the context of a user ID:

Requestor's user ID
By default, a provider is processed in the context of the requestor's user ID for all invocations that are caused by an external CIM operation. This means that the provider runs under the identity of the requestor's user ID, and resource access authorization occurs against this user ID. See the usage notes for the pthread_security_np call in "Callable services descriptions" in z/OS UNIX System Services Programming: Assembler Callable Services Reference for additional information.
Designated user ID

Alternatively, you can provide a designated user ID that runs the provider.

Specify the designated user ID during provider registration using the UserContext and DesignatedUserContext properties of the PG_ProviderModule class.

When a provider is registered with a designated user ID, the CIM server processes all requests under the designated user ID, regardless which client user ID has issued the request.

The user ID of the requestor is still available for the provider and should be used for further authorization checking in order to prevent unauthorized access to a resource. You have to specify similar security definitions for the designated user ID as for regular client users, as described in “Switching identity (surrogate)” on page 29.

Converting data to ASCII, EBCDIC and UTF-8

Character encoding in the CIM over HTTP protocol is done using UTF-8 character encoding. For that reason CIM clients expect valid UTF-8 returned by the CIM server. The z/OS CIM server executes in the Enhanced ASCII mode. This means that all string data within the CIM server’s address space is represented in ASCII rather than EBCDIC encoding. For a provider this means that all string data exchanged with the CIM server is expected to be in ASCII (codepage ISO/IEC 8859-1), encoded in UTF-8 format. Since the native data of z/OS resources is usually represented in EBCDIC, the provider code needs to convert this data before it can return it to the CIM server through the CMPI interface, or when it receives data from the CIM server through the CMPI interface.

UTF-8 is a multi-byte character encoding for UNICODE which can represent much more characters than EBCDIC. While no issue on returning data from a provider through the CIM server to a client, the range of input characters from a client can be larger than a provider can represent in EBCDIC. All valid (7-bit) ASCII characters are also valid UTF-8. Please note that a transformation of the character encoding from EBCDIC to ASCII can generate invalid ASCII, that is ASCII-code above the 7-bit margin.

Therefore it is recommended to compile the provider's C code using the ASCII option of the z/OS XL C/C++ compiler. Using the ASCII option also requires the XPLINK compile and link option.

See Appendix B in the z/OS XL C/C++ Run-Time Library Reference for additional information about the Enhanced ASCII support. Also see the z/OS XL C/C++ Guide and the z/OS XL C/C++ Programming Guide for details about the ASCII compiler option.

Provider installation

To enable the CIM server to find and load provider modules and related modules, a provider has to be stored in the hierarchical file system and the CIM server run-time environment has to be tailored. A CMPI provider for z/OS consists of provider modules, dependent modules, the CIM Schema extensions (MOF), and the CMPI provider registration information (MOF).
Installing providers and dependent load modules

When you develop a CMPI provider, you ship a provider module, a dynamic load library (DLL) module, and, if applicable, its dependent libraries.

__ We recommend to store the provider DLL and its dependent libraries in a separate hierarchical file system directory, such as /usr/lpp/myProd/provider.

__ On systems where program control is enabled, flag the provider DLL and its dependent libraries as program controlled using the extattr UNIX System Services command:

`extattr +p <providerfile>`

We recommend to flag all modules as program controlled by default.

More information:

"Defining modules to program control" in z/OS UNIX System Services Planning

Customizing the CIM server environment for third-party providers

__ To enable the CIM server to locate the provider module, extend the CIM server’s search list for provider directories by setting the `providerDir` configuration property, such as

`providerDir=/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lpp/myProd/provider`

More information: Chapter 9, “CIM server configuration,” on page 53

__ To locate the provider dependent libraries, extend the library search path (LIBPATH) for the CIM server.

- The default library search path for the CIM server is defined in the file `/etc/wbem/cimserver.env` for the started task CFZCIM. Add your installation directory to the LIBPATH, for example:

  `LIBPATH=/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib:
  /usr/lpp/myProd/provider`

- If you run the CIM server and tools from the UNIX System Services shell, extend the LIBPATH of the shell.

Registering a provider with the CIM server

When the provider dynamic load library has been made physically accessible to the CIM server, it needs to be registered via a special MOF file using the `cimmof` command. A provider registration MOF file contains instances of the CIM classes from the provider registration schema, namely of classes PG_ProviderModule, PG_Provider and PG_ProviderCapabilities as shown in Figure 17 on page 278
The instances of these classes contain all the information that the CIM server needs to know about a provider, for example its physical packaging structure, supported CIM classes and namespaces, as well as the set of supported provider operations.

Once the provider registration MOF file has been created with the instances of classes PG_Provider, PG_ProviderModule and PG_ProviderCapabilities, the content of this MOF file can be loaded into the CIM server's root/PG_InterOp namespace using the cimmof command.

The cimmof command stores this information in the CIM server run-time repository.

Example:

cimmof -n root/PG_InterOp TestProviderRegistration.mof

The CIM server automatically migrates the repository from one z/OS version to the next. This means, that once the additional provider MOFs have been installed, there is no need to install them again after a z/OS release upgrade.
If the run-time repository including your definitions has been deleted and the CIM server master repository has to be restored, your CIM Schema extensions and provider registration are lost and you have to register them again.

Therefore these MOF files should be part of your delivery and stored in your hierarchical file system directory, such as for example:

```
/usr/lpp/myProd/schemas
    stores the schema descriptions and registration information

MYPROD_ClassName.mof
    contains the CIM Schema description

MYPROD_ClassNameRegistration.mof
    contains the provider registration
```

More information:

“cimof” on page 77
“PG_Provider” on page 280
“PG_ProviderModule” on page 281
“PG_ProviderCapabilities” on page 283
PG_Provider

Purpose
This class is the logical representation of a CIM provider. Its only properties are the name of the provider, the name of the provider module in which the code of the provider physically resides and the name of a SAF security profile to be checked before a client is granted access to the provider.

Properties

string ProviderModuleName
The name of the provider module containing the code for this provider. This name needs to match the value of the Name property of the corresponding instance of class PG_ProviderModule.

string Name
The name of the provider. This name is used to identify a specific provider within a provider module (dynamic load library) and specifies the prefix of a provider’s _Create_<mi-type>M1() initialization function.

string SecurityAccessProfile
This property defines the name of a z/OS security server’s profile in the CIM server’s WBEM class that will be checked for a requestor’s access before a request is routed to this provider. Depending on the type of the CIM operation, a different level of access to the security profile is required as listed in Table 2 on page 28.

This is not a required property and can be omitted from the provider registration MOF.

Examples
Example of an instance of class PG_Provider in MOF syntax:

instance of PG_Provider
{
    ProviderModuleName = "TestClassProviderModule";
    Name = "TestClassProvider";
};
**PG_ProviderModule**

**Purpose**
This class represents the physical packaging of one or more providers in a dynamic load library or shared library.

**Properties**

- **string Name**
  The logical name of the provider module.

- **string Vendor**
  The name of the provider module vendor, for example, IBM.

- **string Version**
  The provider module version.

- **string InterfaceType**
  The interface type implemented by the provider. Must be CMPI for z/OS.

- **string InterfaceVersion**
  The interface version number implemented by the provider. Must be 2.0.0 for CMPI on z/OS.

- **string Location**
  The name of the dynamic load library or shared library in the hierarchical file system without a path name. The name specified for Location is automatically prefixed with lib and extended with .so by the CIM server: lib<Location>.so

- **boolean ShareAS**
  Setting the ShareAS property to false causes the provider module to run in its own copy of a provider agent process. No other provider module will be loaded into this process.

  Setting the ShareAS property to false has a major impact on the performance, so you should not set it to ‘false’ unless there is an urgent need for a provider module to be protected from other provider modules. The default setting of ShareAS is true.

  Setting ShareAS to false is only honored by the CIM server, if it is running with the configuration property forceProviderProcesses set to true.

- **uint16 UserContext**
  Defines the user context in which this provider module is invoked.

  Values:

  2 (Requestor), default
  The provider is invoked in the security context of the user requesting an operation.

  3 (Designated User)
  The provider is invoked in the security context of the user ID specified by the DesignatedUserContext property.

  See “Running providers in a designated user context” on page 42 for a general description on running a provider module with a designated user context.
string DesignatedUserContext

Specifies the user ID providing the context in which this provider module is invoked (regardless of which user requests an operation).

Values:

NULL when UserContext = 2

non-NULL value when UserContext = 3

See “Running providers in a designated user context” on page 42 for a general description on running a provider module with a designated user context.

string ModuleGroupName

Specifies a group name for the provider module, if the configuration property forceProviderProcesses is true. Else it has no effect.

This property controls which provider modules are running together in the same provider agent process.

- If the specified value is CIMServer, the provider module is loaded into the CIM server process.
- Provider modules having the same group name other than CIMServer are loaded into a single agent process.
- If no module group name is defined, the provider either runs in a single shared provider agent process together with all other providers without a module group name, or in its own distinct provider agent process in case ShareAS is true.

Can be set dynamically using the cimprovider command (see “cimprovider” on page 81).

Examples

Example of an instance of class PG_ProviderModule in MOF syntax:

instance of PG_ProviderModule
{
    Name = "TestClassProviderModule";
    //The library name on disk
    Location = "TestClassProvider";
    // (will be extended to libTestClassProvider.so)
    Vendor = "IBM";
    Version = "1.0.0";
    InterfaceType = "CMPI";
    InterfaceVersion = "2.0.0";
    ShareAS = true;
    UserContext = 2;
};
**PG_ProviderCapabilities**

**Purpose**

This class describes the specific capabilities of a provider. Multiple instances of PG_ProviderCapabilities can be created for each provider allowing the same provider to be registered, for example, for multiple CIM classes.

**Properties**

- **string ProviderModuleName**
  
  The name of the provider module as specified in the corresponding instances of classes PG_Provider and PG_ProviderModule.

- **string ProviderName**
  
  The name of the provider as specified in the corresponding instance of class PG_Provider.

- **string CapabilityID**
  
  A value that uniquely identifies this Capabilities instance within the set of Capabilities for the designated provider.

- **uint16[] ProviderType**
  
  Enumerates the kind of provider capabilities (=supported operations) defined for the associated provider:
  
  - 2 Instance
  - 3 Association
  - 4 Indication
  - 5 Method
  - 6 IndicationConsumer (not supported for z/OS)
  - 7 InstanceQuery

- **string ClassName**
  
  Describes the CIM class for which the associated provider supplies instances, associations or indications information.

- **string[] Namespaces**
  
  Describes the namespaces that are supported by the provider for this CIM class.

- **string[] SupportedProperties**
  
  Lists the properties supported by this provider. If this array is empty, the provider **must** support all of the properties defined in the class.

- **string[] SupportedMethods**
  
  Lists the methods supported by this provider. If this array is empty, the provider **must** support all the methods defined in the class.

**Examples**

Example of an instance of class PG_ProviderCapabilities in MOF syntax:

```mof
class PG_ProviderCapabilities

  // The provider module as defined in PG_ProviderModule
  ProviderModuleName = "TestClassProviderModule";

  // The provider name as defined in PG_Provider
  ProviderName = "TestClassProvider";

  // The CapabilityID as defined in the MOF
  CapabilityID = "1";

  // Name of the CIM class as defined in the MOF
  ClassName = "IBMzOS_TestClassB";

  // Namespaces supported by the provider
  Namespaces = {"root/cimv2","root/test"};

  // ProviderType: [2, 5]; // Instance, Method

  SupportedProperties = NULL; // All properties

  SupportedMethods = NULL; // All methods

```

Chapter 16. CMPI provider development for z/OS 283
instance of PG_ProviderCapabilities
{
  //The provider module as defined in PG_ProviderModule
  ProviderModuleName = "TestClassProviderModule";
  //The provider name as defined in PG_Provider
  ProviderName = "TestClassProvider";
  CapabilityID = "2";
  //Name of the CIM class as defined in the mof
  ClassName = "IBMzOS_TestIndication";
  Namespaces = {"root/cimv2"};
  ProviderType = {4}; // Indication
  SupportedProperties = NULL; // All properties
  SupportedMethods = NULL; // All methods
};
Using the out-of-process support for providers

When the CIM server is started in out-of-process mode using the `forceProviderProcesses` configuration property, providers may run in separate address spaces. Then, the z/OS-specific property `ShareAS` and the common property `ModuleGroupName` for class `PG_ProviderModule` are considered. You may specify them during provider registration via the registration MOF file. `ModuleGroupName` can also be set dynamically at runtime using the `-g` option of the `cimprovider`.

To specify that a provider shall always run in its own provider agent process,
- set the z/OS-specific property `ShareAS` to `false` during provider registration.

To define a group of providers sharing a provider agent process,
- assign the same module group name to the respective providers using the property `ModuleGroupName` during provider registration.

To specify that a provider shall run in the CIM server address space,
- assign the module group name `CIMserver` to the property `ModuleGroupName` of the provider during provider registration.

Examples of a provider registration MOF file with properties specified for the out-of-process support:

```cim
instance of PG_ProviderModule
{
    Name = "OSBase_TestClassProviderModule";
    //The library name on disk
    Location = "cmpiOSBase_TestClassProvider";
    Vendor = "IBM";
    Version = "2.0.0";
    InterfaceType = "CMPI";
    InterfaceVersion = "2.0.0";
    ShareAS = false;
    ModuleGroupName = "CMPITEST";
}
```

Samples

Examples for CMPI providers can be found on the OpenPegasus CVS Repository, located in the `pegasus/src/Providers/sample/CMPI` directory. You can access them in the same ways as described in “Obtaining the required header files” on page 272. Please note that these examples have been enabled for z/OS only in an OpenPegasus build environment and will need some minor adoptions for a custom build environment.

Additional examples are available from the SBLIM OpenSource project (packages sblim-cmpi-<xxx>) hosted on SourceForge.net. Although the CIM providers from SBLIM apply to Linux platforms only, they are examples for how to write CIM providers in general. The SBLIM project also provides a number of useful tools and documents related to provider development.
Chapter 17. CIM indications

Indications in CIM are represented as instances of class CIM_Indication. This abstract class serves as the base class for all indication classes.

Indications are transient instances used to distribute information from an indication generator to an arbitrary number of indication consumers. Therefore, they are typically very short-living. Indications have a source namespace, this is the value of the `SourceNamespace` property of the CIM_IndicationFilter instance that produced the indication. Although indications are instances of CIM classes, they are unique in that they cannot be addressed, but can only be received by subscription. Hence, indication instances cannot be enumerated, created, deleted, retrieved or modified by client operations.

Note that z/OS does not ship generic providers, that is, an indication subscription is only processed if the required indication provider exists and is registered with the CIM server for a certain CIM resource class.

The CIM Schema version provided with z/OS supports two types of indications (representing different types of events) which are modeled as CIM_Indication subclasses. These subclasses include:

**CIM_InstIndication**
- used to report life cycle events for CIM instances. Types of events include: instance creation, deletion, modification, method invocation and read access. For each of these types, a specific subclass of CIM_InstIndication is defined in the CIM Schema: CIM_InstCreation, CIM_InstDeletion, CIM_InstModification, CIM_InstMethodCall and CIM_InstRead. Only the first three are currently supported for z/OS.

**CIM_ProcessIndication**
- used to report the occurrence of any other event, typically alert type events. See "CIM_ProcessIndication" on page 288.
The CIM indication class hierarchy models the types of events that can be detected. An instance of `CIM_Indication` represents the occurrence of an event in general. Indication instances cannot be addressed, but they have a source namespace. Although indications are modeled using CIM classes, indications are unique in that they cannot be manipulated or retrieved, but they can only be received by subscription. The `CIM_Indication` class is the base class for all other indication classes. It includes the following properties:

- **IndicationIdentifier**
  identifies indication instances uniquely within their source namespace.

- **IndicationTime**
  describes, to the extent possible, the time and date of the creation of the underlying event for the indication.

- **CorrelatedIndications**
  specifies a list of other indications, referenced by their `IndicationIdentifier` property values, that are related to this indication. These `IndicationIdentifier` property values are interpreted to have the same source namespace as this indication.

While the `CorrelatedIndications` property values are to be interpreted in the context of a single CIM namespace, any instances of other classes of the CIM Event Model do not need to be located in the same namespace.

### CIM_ProcessIndication

CIM_ProcessIndication models any events other than life cycle events. In the CIM Schema version supported for z/OS, the following two subclasses of `CIM_ProcessIndication` are defined:

- **CIM_AlertIndication** – signals the occurrence of an alert type of event. Properties of this subclass include `PerceivedSeverity`, `ProbableCause`, `RecommendedAction` and `Trending`, describing an alerting situation.
CIM_SNMPTrapIndication – used to map SNMP traps to CIM indications. This is currently not supported by the z/OS CIM server.

**CIM_InstIndication (Lifecycle Event)**

An instance of CIM_InstIndication denotes the occurrence of a life cycle event on a CIM instance. The possible life cycle events are: creating an instance, deleting an instance, modifying an instance, reading an instance or invoking a CIM method on an instance. An instance of CIM_InstIndication includes an embedded copy (that is, a current snapshot) of the instance, SourceInstance, on which the life cycle event occurred.

Instances of CIM_InstModification include an embedded copy of the instance, PreviousInstance, before the modification occurred.

Lifecycle events on CIM instances include both, changes caused by a CIM client, and changes that happen spontaneously from a CIM client perspective due to volatile behavior of the CIM provider.

**CIM_InstModification**

Lifecycle events on CIM instances include both, changes caused by a CIM client, and changes that are caused by a change of the underlying system resource that is represented via a CIM instance.

---

**CIM subscription mechanism**

The CIM Event Model defines how CIM clients subscribe to receive indications as shown in [Figure 19 on page 291](#) and [Figure 20 on page 291](#). A CIM_IndicationFilter instance describes the set of conditions, a CIM_ListenerDestinationCIMXML instance defines the CIM listener and the communication protocol, that is, it describes the method and targets for distributing the indications. Finally, a CIM_IndicationSubscription association instance between the CIM_IndicationFilter instance and the CIM_ListenerDestinationCIMXML instance is used to subscribe for receiving these indications. The creation of this association instance activates the subscription.

![CIM subscription mechanism diagram]

*Figure 19. Indication subscription class diagram*
**CIM_IndicationFilter**

An instance of CIM_IndicationFilter describes the set of indications of interest by means of a query expression. This is also called the desired indication stream. The most relevant properties of CIM_IndicationFilter are:

- **Name**, **CreationClassName**, **SystemName**, **SystemCreationClassName** – key properties.
- **SourceNamespace** – defines the source namespace for the indications resulting from this indication stream.
- **Query** – query string, like “select * from CIM_InstModification where ...”; defines the indication class, filter condition and property list of the indication stream.
- **QueryLanguage** – defines the query language used in the **Query** property. The z/OS CIM server supports the query languages “DMTF:CQL” (CIM Query Language) and “WQL” (WBEM Query Language). For more information, see the [CIM Query Language Specification](#).
- **DeliveryRetryInterval** defines the minimum time between two delivery retries.
- **DeliveryRetryAttempts** defines the maximum number of delivery retries.

For information about the complete set of properties of a CIM_IndicationFilter, refer to the [CIM Event Model White Paper](#) or to the definition of this class in the CIM Schema. The white paper also contains an example of a CIM_IndicationFilter instance.

**CIM_ListenerDestinationCIMXML**

An instance of CIM_ListenerDestinationCIMXML defines “how and where” to send an indication. In particular, the CIM_ListenerDestinationCIMXML instance defines the desired indication destination, encoding and protocol for delivery of the indication stream. CIM_ListenerDestinationCIMXML specializes CIM_ListenerDestination and is used for indication consumers that support the CIM Operations over HTTP protocol (see [Specification for CIM Operations over HTTP, DSP0200](http://www.dmtf.org/standards/documents/WBEM/DSP200.html)).

The CIM_ListenerDestination class hierarchy can be extended to allow the definition of additional indication handling mechanisms.

The most relevant properties of CIM_ListenerDestinationCIMXML are:

- **Name**, **CreationClassName**, **SystemName**, **SystemCreationClassNam** – key properties
- **Destination** – URL to which the indications are to be delivered

For information about the complete set of properties of CIM_ListenerDestinationCIMXML, refer to the [CIM Event Model White Paper](#) or to the definition of this class in the CIM Schema.
**CIM_IndicationSubscription**

Primarily, an instance of CIM_IndicationSubscription defines the association between a CIM_IndicationFilter instance and a CIM_ListenerDestinationCIMXML instance. In addition, it includes a set of properties that further specify the behavior of a subscription. The most relevant properties of CIM_IndicationSubscription are:

- **The Repeat Notification** properties (those having “RepeatNotification” contained in their property name) define the behavior for handling indications that report the occurrence of the same underlying event (that is, the disk is still generating I/O errors and has not yet been repaired).

- **The Subscription State** properties (those having “SubscriptionState” contained in their property name) allow a CIM client to monitor and control the state of the subscription.

- **The Subscription Failure Handling** properties (OnFatalErrorPolicy, OtherOnFatalErrorPolicy, FailureTriggerTimeInterval) define the desired behavior when a fatal error occurs during subscription processing.

- **The Subscription Duration** properties (SubscriptionDuration, SubscriptionStartTime, SubscriptionTimeRemaining) allow to expire a subscription automatically, based upon elapsed time since its creation, and to monitor the elapsed times since creation and until expiration.

You can find more detailed information about these properties as well as the complete set of properties of CIM_IndicationSubscription in the CIM Event Model White Paper or in the definition of this class in the CIM Schema.
Part 6. Messages
Chapter 18. z/OS specific messages

Messages are written into the appropriate logs and also displayed at the z/OS console.

All messages issued by the CIM server are part of the underlying OpenPegasus code. This section documents only those messages that are specific while using the CIM server on z/OS, together with explanation, system action, (system) programmer and user response.

All other OpenPegasus messages are wrapped by one of the following generic z/OS messages.

CFZ00001I for INFORMATION log messages
CFZ00002W for WARNING log messages
CFZ00004E for SEVERE and FATAL log messages

CEZ-prefix messages

CEZ02000I Requesting CONFIG ONLINE for CPU CPU-address

Explanation: The IBMzOS_Processor method RequestStateChange has been issued with RequestedState=Enabled.

System action: None.

System programmer response: Issue a CF CPU(CPU-address),ONLINE command, or use your automation tool to set the CPU CPU-address online.

User response: None.

CEZ02000I Requesting CONFIG OFFLINE for CPU CPU-address

Explanation: The IBMzOS_Processor method RequestStateChange has been issued with RequestedState=Offline.

System action: None.

System programmer response: Issue a CF CPU(CPU-address),OFFLINE command, or use your automation tool to set the CPU CPU-address offline.

User response: None.

CEZ03000E Request user ID user-ID requires UPDATE permission on profile IOSCDR CL(FACILITY).

Explanation: A CIM operation was invoked that requires the use of an authorized IOSCDR service. The IOSCDR service is used by CIM providers to retrieve device identification information (such as the serial number and the model number) for an I/O device. Providers that instrument the CIM classes IBMzOS_SBProtocolEndpoint or IBMzOS_SBInitiatorTargetLogicalUnitPath are an example for this scenario.

System action: The requested CIM operation is returned in error.

System programmer response: Verify if the user should be permitted to perform operations using the IOSCDR service. If so, grant the user user-ID UPDATE permission to the profile IOSCDR in the class FACILITY. Then restart the CIM server.

User response: Report this problem to your system programmer.

CEZ03001E Internal error occurred. SMI-S Indication Data Cache error error-code.

Explanation: The SMI-S data cache and thread are in an unrecoverable error state. error-code describes the kind of error.

System action: The requested CIM operation is returned in error.

System programmer response: The error code indicates the kind of error:
1 (SMIS_CACHE_CONTROL_ERROR) Error in the data cache control structures
2 (SMIS_CACHE_ERROR) Error in the data cache data
3 (SMIS_THREAD_CREATION_ERROR) Error in data cache thread
Restart the CIM server. If the problem persists, contact IBM service for assistance.

User response: Report this problem to your system programmer.

CEZ03002W Lost connection to CEA, trying to reconnect. CIM Indications may get lost.

Explanation: The SMI-S CIM indication provider has lost the connection to CEA. Without this connection, no SMI-S CIM indications can be generated, for example for changes on port controllers.

System action: The CIM provider continuously attempts to reconnect to CEA until it becomes available.

System programmer response: Restart Common Event Adapter (CEA) in full function mode.

User response: Report this problem to your system programmer.

CEZ03003W Failed to reconnect to CEA. CIM Indications may get lost.

Explanation: The SMI-S CIM indication provider failed to reconnect to CEA. Without this connection, no SMI-S CIM indications can be generated, for example for changes on port controllers.

System action: The CIM provider continuously attempts to reconnect to the CEA until it becomes available.

System programmer response: Restart Common Event Adapter (CEA) in full function mode.

User response: Report this problem to your system programmer.

CEZ03004I Successfully reconnected to CEA.

Explanation: The SMI-S CIM indication provider has successfully reconnected to CEA.

System action: None.

System programmer response: None.

User response: None.

CEZ03005I Successfully re-established subscription to CEA.

Explanation: The SMI-S CIM indication provider has successfully renewed its subscriptions for ENF signals to CEA.

System action: None.

System programmer response: None.

User response: None.

CEZ03006E Subscription to CEA failed for handler module-name with reason code reason-code.

Explanation: The SMI-S CIM indication provider failed to subscribe to CEA in order to receive ENF signals through CEA subscription handler module-name.

System action: The requested CIM operation is returned in error.

System programmer response: The reason-code indicates why the CEA subscription to handler module-name failed. See “Appendix C. CEA reason codes” on page 339 for error details. After correcting the error, restart the CIM server.

User response: Report this problem to your system programmer.

CEZ03007E Failed to retrieve CEA event, reason code reason-code.

Explanation: The SMI-S CIM indication provider failed to receive a CEA event and therefore cannot process CIM indications.

System action: None.

System programmer response: See “Appendix C. CEA reason codes” on page 339 for error details. After correcting the error, restart the CIM server.

User response: Report this problem to your system programmer.

CEZ03008W Renewing CEA subscription after operator unsubscribe.

Explanation: The SMI-S CIM indication provider has detected an operator forced unsubscribe from CEA. Since this would leave orphaned CIM indication subscriptions, the subscription to CEA is automatically re-established.

System action: The SMI-S CIM provider automatically re-establishes the removed CEA subscriptions.

System programmer response: Remove SMI-S CIM subscriptions through the cimsub utility (see “cimsub” on page 110) or make sure the CIM clients are properly unsubscribed from SMI-S CIM indications for this system.

User response: Report this problem to your system programmer.

CEZ03009W Missed CEA event(s) caused loss of CIM indications.

Explanation: The SMI-S CIM indication provider was informed by the Common Event Adapter (CEA) that it has missed a number of events. This causes a loss of CIM indications for subscribed CIM Client applications.

System action: None.
System programmer response: None.

User response: If known, inform the owners of CIM client applications that have subscribed to this z/OS system.

CEZ03010E  User user-ID not authorized to connect to Common Event Adapter (CEA).

Explanation: The user user-ID is not authorized to connect to the Common Event Adapter (CEA). The CIM SMI-S indication providers depend on CEA for issuing indications about state changes related to FC Ports. After correcting the error, restart the CIM server.

System action: None.

System programmer response: Ensure that the user user-ID has READ access to profile CEA.CONNECT in the SERVAUTH class.

See "Setting up the CIM server for storage management" on page 41 for the authorizations required to connect to CEA.

User response: Contact your system programmer or security administrator.

CEZ03011E  User user-ID not authorized for subscription to Common Event Adapter (CEA).

Explanation: The user is not authorized to subscribe for ENF signals through the Common Event Adapter (CEA). The CIM SMI-S indication providers depend on the CEA for issuing indications about state changes.

System action: None.

System programmer response: Ensure the user user-ID has READ access to profiles

- CEA.SUBSCRIBE.ENF_0009*
- CEA.SUBSCRIBE.ENF_0027*
- CEA.SUBSCRIBE.ENF_0033*

in the SERVAUTH class.

See "Setting up the CIM server for storage management" on page 41 for the authorizations required to connect to CEA. After correcting the error, restart the CIM server.

User response: Contact your system programmer or security administrator.

CEZ03012E  Connection to CEA failed with reason code reason-code.

Explanation: The SMI-S CIM indication provider failed to connect to the Common Event Adapter and therefore cannot process CIM indications.

System action: CIM Indications for SMI-S are unavailable.

System programmer response: See "Appendix C. CEA reason codes" on page 339 for error details. After correcting the error, restart the CIM server.

User response: Contact your system programmer.

CEZ05000E  Internal error detected in provider module module-name when method method-name invoked system service service-name. The service returned

RC=return-code RSN=CEA-reason-code.

Additional diagnostic information:

- CEAERRO_Diag1=code1
- CEAERRO_Diag2=code2
- CEAERRO_Diag3=code3
- CEAERRO_Diag4=code4
- CEAERRO_Msg=text

Explanation: The system encountered an internal error while processing a CIM request. The following information is provided:

module-name
Name of CIM provider module

method-name
Name of CIM method invoked

service-name
Name of the internal service, usually in the CEA component

return-code
Internal return code

CEA-reason-code
Internal CEA reason code. See "Appendix C. CEA reason codes" on page 339 for details.

CEAERRO_Diag1-4
Internal values representing errors in system processing on behalf of the CIM request

CEAERRO_Msg
Textual information saved by system processing on behalf of the CIM request

System action: System processing ended with the error information described in this message.

System programmer response: See CEAERRO_Msg for more informational messages about the problem. If the problem is still unclear or no additional messages are available, contact IBM Service for assistance.

User response: Report this problem to your system programmer.

CEZ05001E  Internal error detected in provider module module-name when method method-name invoked system service service-name. The service returned

RC=return-code RSN=CEA-reason-code.

Explanation: The system encountered an internal error while processing a CIM request. The following information is provided:

module-name
Name of CIM provider module

method-name
Name of CIM method invoked

System action: System processing ended with the error information described in this message.

System programmer response: See CEAERRO_Msg for more informational messages about the problem. If the problem is still unclear or no additional messages are available, contact IBM Service for assistance.

User response: Report this problem to your system programmer.
**service-name**
Name of the internal service, usually in the CEA component

**return-code**
Internal return code

**CEA-reason-code**
Internal CEA reason code. See “Appendix C. CEA reason codes” on page 339 for details.

**System action:** The requested CIM operation is returned in error. System processing ended with the error information described in this message.

**System programmer response:** Contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05002E** Common Event Adapter (CEA) not available.

**Explanation:** A CIM method was invoked, but the CEA address space was not active to process the request.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Enter the command START CEA from the operator console to start the CEA address space. Verify that CEA is active through the command D A,CEA.

**User response:** Report this problem to your system programmer.

---

**CEZ05003E** User user-name not authorized for Common Event Adapter (CEA) request.

**Explanation:** A CIM method was invoked, but the user is not authorized to issue requests to the CEA component.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Ensure that the user has access to CEA. Refer to “RACF setup” on page 39.

**User response:** Report this problem to your system programmer.

---

**CEZ05004E** IPCS Sysplex Dump Directory cannot find incident information.

**Explanation:** A CIM method was invoked to locate a specific incident, but the Common Event Adapter (CEA) component cannot locate the incident in the sysplex dump directory (SYS1.DDIR). Common reasons include:
  - Sysplex dump directory SYS1.DDIR (or equivalent data set name) is not set up correctly
  - Dump incident is not in the directory
  - Incident could have been previously deleted from the directory.

**System action:** The requested CIM operation is returned in error. If the failure occurred while performing a set tracking number or set PMR number operation, the function ends without having updated either value.

**System programmer response:** Verify that the sysplex dump directory exists and is usable. Default name is SYS1.DDIR. For more information, see the topic on troubleshooting problems in z/OS Management Facility User’s Guide. If the problem persists, contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05005E** System REXX not available.

**Explanation:** A CIM method was invoked, requiring the invocation of a system REXX exec. However, the System REXX address space (AXR) or facilities that it provides are not available.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Enter the command START AXRPSTRT from the operator console to start System REXX. Verify that System REXX is active with the D A,AXR command.

**User response:** Report this problem to your system programmer.

---

**CEZ05006E** System REXX is not configured to support compiled REXX execs.

**Explanation:** A CIM method was invoked, requiring the invocation of a system REXX exec. However, the System REXX component cannot process the exec. This usually indicates that the run time support for compiled REXX has not been set up.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** The REXX library and the REXX Alternate library must be installed. Refer to the Program Directory of these optional products for installation instructions.

**User response:** Report this problem to your system programmer.

---

**CEZ05007W** The request method-name has timed out.

**Explanation:** A CIM method was invoked, requiring the invocation of a system REXX exec that timed out.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** This is an internal
problem related to the TIMEINT parameter on the AXREXX macro. Contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05008W** The request method-name could not be processed at this time.

**Explanation:** A CIM method was invoked, but System REXX is overloaded and cannot schedule the corresponding REXX exec to run at this time.

**System action:** System REXX limits the number of active and waiting requests to 5000. The requested CIM operation is returned in error.

**System programmer response:** Enter the command SYSREXX STATUS and check the value specified as "REQUESTS QUEUED" in message AXR0200I. Have the user retry the operation when there are fewer System REXX requests being processed. If still unsuccessful, contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05009E** SYS1.MIGLIB is not APF authorized.

**Explanation:** A CIM method was invoked that requires the use of an authorized service in SYS1.MIGLIB (such as AMATERSE). However, SYS1.MIGLIB is not APF authorized, which prevents CEA from invoking those programs.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** From the operator console, enter the command:

```
SETPROG APF,ADD,DSN=SYS1.MIGLIB,
VOL=<volser>
```

where `<volser>` is the volume on which MIGLIB resides.

**User response:** Report this problem to your system programmer.

---

**CEZ05010E** User user-name not authorized to view operator log snapshot logstream-or-dataset-name.

**Explanation:** A CIM method was invoked, referencing an OPERLOG snapshot for a specific incident, but the invoker is not SAF authorized to view information about the snapshot. OPERLOG diagnostic snapshots are stored in DASD log streams with data set names containing the high level data set qualifier specified in the CEAPRMxx PARMLIB member.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Avoid removing events that have outstanding subscriptions.

**User response:** Unsubscribe to the event specified in the message and resubscribe.

---

**CEZ05012E** The Common Event Adapter (CEA) event event-name was forced removed by the system operator.

**Explanation:** The system operator used the CEAunsubscribe console command to force the removal of this event while there was a CIM user subscribed to it. The following console command may have been issued:

```
f cea,diag,remove,client=clientname,
   event=eventname
```

**System action:** The CIM indication will no longer be surfaced.

**System programmer response:** Avoid removing events that have outstanding subscriptions.

**User response:** Unsubscribe to the event specified in the message and resubscribe.

---

**CEZ05013E** Common Event Adapter (CEA) is running in minimum mode.

**Explanation:** The system operator has forced CEA into 'minimum mode' by using the command:

```
f cea,mode=min
```

CIM indication processing is unavailable.
CIM indications will not be supported.

**System programmer response:** Change CEA to run in 'full mode'. The following console command can be used:

```
f cea,mode=full
```

**User response:** Contact your system programmer.

---

**CEZ05014E** Internal error detected in provider module `module-name` while invoking method `method-name`.

**Explanation:** A CIM method was invoked, but an internal provider error occurred in the CIM provider.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05015E** Target operating system version/release not supported for provider module `module-name` method `method-name`.

**Explanation:** A CIM method was invoked, but the provider requires the identified minimum operating system version/release.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Contact IBM Service for assistance.

**User response:** Report this problem to your system programmer.

---

**CEZ05016E** IPCS Sysplex Dump Directory failure.

**Explanation:** A CIM method was invoked to locate incident information, but the Common Event Adapter (CEA) component encountered a File Open Error when accessing the sysplex dump directory (SYS1.DDIR or equivalent data set name). A possible cause is that SYS1.DDIR is not set up correctly.

**System action:** The requested CIM operation is returned in error.

**System programmer response:** Ensure that no other users are attempting to access the sysplex dump directory at the same time by checking for an exclusive ENQ on SYS1.DDIR (using D GRS). If so, consider cancelling the suspect user or job.

**User response:** Report this problem to your system programmer.

---

**CEZ10000E** Unable to obtain a passticket for GPMSERVE. RACF permissions probably missing.

**Explanation:** The Monitoring providers were unable to obtain a valid passticket for the application GPMSERVE (RMF Distributed Data Server).

**System action:** The CIM request is not processed.

**System programmer response:** Make sure that the RMF Distributed Data Server is set up for accepting PassTickets as described in z/OS RMF User’s Guide.

**User response:** Contact your system programmer.

---

**CFZ-prefix messages**

- **CFZ00409E** Bind failed: subsequent message.
  **Explanation:** The CIM server is unable to bind the socket.
  **System action:** None.
  **System programmer response:** The reason for the bind failure is described in the subsequent message.
  **User response:** Report this problem to your system programmer.

- **CFZ2202W** Property value is not valid: `name=value`.
  **Explanation:** The value that was specified for the configuration property is not valid. See Chapter 9, “CIM server configuration,” on page 53 for the correct
values of configuration properties.

System action: None.

System programmer response: None.

User response: Re-enter the command specifying a correct value for the configuration property.

---

CFZ02207W The configuration property name is not dynamic.

Explanation: The configuration property name cannot be changed dynamically for a running CIM server. Instead the change has to be made as a planned value to become effective after a CIM server restart. See "cimconfig" on page 79 or "MODIFY console command" on page 113 for details on how to change planned values.

System action: None.

System programmer response: None.

User response: Change the planned configuration value and restart the CIM server.

---

CFZ02300I Configuration property conf-property is not supported. Setting ignored.

Explanation: The mentioned configuration property is no longer supported.

System action: The CIM server ignores this setting and continues.

System programmer response: Please remove the mentioned configuration property from the planned configuration of the CIM server's /etc/wbem/cimserver_planned.conf.

User response: None.

---

CFZ03029E Unsupported UserContext value: "value".

Explanation: A provider module was registered with a UserContext value of value, but that value is not supported by this version of the CIM server. Valid values are 2 for "Requestor" and 3 for "Designated User".

System action: The provider module is not registered.

System programmer response: Check the provider registration MOF and add a valid user ID for the DesignatedUserContext property to all provider modules that are registered with a UserContext value of 3.

User response: Contact your system programmer.

---

CFZ03030E Missing DesignatedUserContext property in PG_ProviderModule instance.

Explanation: A provider module was registered with a UserContext value of 3 ("Designated User"). The user ID of the designated user has to be specified in DesignatedUserContext, but no value was found (see "PG_ProviderModule" on page 281).

System action: The provider module is not registered.

System programmer response: None.

User response: None.

---

CFZ05000E A system error occurred. Retry the CIM operation at a later time.

Explanation: A CIM-XML operation exceeds the server's memory.

System action: Stop the CIM-XML operation.

System programmer response: Look for message CFZ08101E identifying the source of the CIM-XML request. Contact the owner of the application issuing the request and analyze the reason for the size of the operation. Limit the result objects for this request. Restart the server to clean it up.

User response: Contact your system programmer.

---

CFZ05203W The user user-ID is not authorized to run operation in the namespace namespace.

Explanation: The user ID that invoked CIM operation operation is not authorized to run this operation in namespace namespace of the CIM server.

System action: The CIM request is denied.

System programmer response: Check the system console for further detailed error messages that indicate which authorization is missing for user user-ID. In most cases, the user has no UPDATE authority for profile CIMSERV in class WBEM.

User response: Contact your system administrator for obtaining the required level of authorization.

---

CFZ06201W Command not recognized by CIM server.

Explanation: The command that was entered in the system console is not supported by the CIM server.

System action: None.

System programmer response: None.

User response: None.

---

CFZ06202I STOP command received from z/OS console, initiating shutdown.

Explanation: The CIM server received the STOP command from the console.

System action: The CIM server is shutting down.

System programmer response: None.
User response: None.

CFZ06203E CIM server Console command thread cannot be created: error-text (errno error-number, reason code X'reason-code').

Explanation: The CIM server cannot start the thread handling commands issued at the console. For a description of error error-text with error number error-number and the last four digits of the reason code X'reason-code', see z/OS UNIX System Services Messages and Codes, or enter the reason code in the BPXMTXT TSO command.

System action: None.

System programmer response: The CIM server cannot be stopped using the console command. To stop the CIM server, purge the address space or use a privileged UNIX user ID to issue the command cimserver -s from the UNIX System Services command prompt.

User response: Contact your system programmer.

CFZ06204E Console Communication Service failed: error-text (errno error-number, reason code X'reason-code').

Explanation: The CIM server is connected to the system console by using the Console Communication Service. The CIM server received the unrecoverable error error-text. For a description of error error-text with errno error-number and the last four digits of the reason code X'reason-code', see z/OS UNIX System Services Messages and Codes, or enter the reason code in the BPXMTXT TSO command.

System action: CIM server shuts down.

System programmer response: Errno error-number and the last four digits of the reason code X'reason-code' point out the reason for the error. Check the console for more messages indicating the problem.

User response: None.

CFZ06205E CIM MODIFY command rejected due to syntax error.

Explanation: A MODIFY command was entered for the CIM server that could not be recognized due to invalid syntax.

System action: None.

System programmer response: None.

User response: Enter the command with the correct syntax.

CFZ06206I Syntax is: MODIFY CFZCIM,APPL=CONFIG, name=value,PLANNED]

Explanation: This messages describes the expected format for CIM server MODIFY command.

System action: None.

System programmer response: None.

User response: None.

CFZ06207E Failed to update configuration value.

Explanation: The CIM server failed to update a configuration value that was entered through the system console.

System action: None.

System programmer response: Look for other messages indicating the problem.

User response: Look for other messages indicating the problem.

CFZ06208I Updated current value for name to value.

Explanation: A configuration value for a running CIM server has immediately been updated. The changed value will stay in effect as long as the CIM server is running. After a restart the value is reset to either the default or to the planned configuration value.

System action: The change requested by the MODIFY command is now in effect.

System programmer response: None.

User response: None.

CFZ06209I Updated planned value for name to value.

Explanation: A configuration value has been updated for the planned configuration of the CIM server. It will become active after the CIM server is restarted. This change is persistent until the planned value is changed again.

System action: The change requested by the MODIFY command becomes effective after the next CIM server restart.

System programmer response: None.

User response: None.

CFZ06210I This change will become effective after CIM server restart.

Explanation: The change requested by the MODIFY command will not be in effect until the CIM server is restarted.

System action: None.
CFZ06211E  MODIFY command failed: message
Explanation: A configuration update requested through the MODIFY command failed. The detailed cause is indicated by message.
System action: None.
System programmer response: None.
User response: None.

CFZ06212E  name is not a valid configuration property.
Explanation: The configuration property name is not recognized by the CIM server as a valid configuration property.
System action: None.
System programmer response: None.
User response: Use the correct name for the configuration property and enter the command again.

CFZ07801E  CIM HTTP or HTTPS connection failed to create the socket.
Explanation: The CIM server was unable to create a socket.
System action: None.
System programmer response: Check the PORT and PORTRANGE statements in the PROFILE.TCPIP configuration file to ensure that the ports specified by the httpPort and httpsPort CIM server configuration properties are accessible by the CIM server. Check the security product configuration to ensure that the CIM server is able to access the ports specified by the httpPort and httpsPort CIM server configuration properties.
User response: None.

CFZ07805E  Failed to bind socket on port port-number: error-text (error code error-code, reason code 0xreason-code).
Explanation: Before listening on network port port-number the CIM server failed to bind the socket with error-code and 0xreason-code. It therefore will not be able to communicate over this network port. Probably the port is already in use by another program or has been reserved by the TCP/IP configuration.
System action: The CIM server does not start.
System programmer response: Error code error-code and the last four digits of the reason code 0xreason-code point out the reason for the error.
For a description of error error-text with error code error-code and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.
User response: None.

CFZ07806E  Failed to set permission on local domain socket: error-text (error code error-code, reason code 0xreason-code).
Explanation: The CIM server is not able to set the permission on socket file socket for local communication.
System action: The CIM server does not start.
System programmer response: Error code error-code and the last four digits of the reason code 0xreason-code point out the reason for the error. For a description of error error-text with error code error-code and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.
User response: None.

CFZ07807E  Failed to listen on socket socket-number: error-text (error code error-code, reason code 0xreason-code).
Explanation: The CIM server failed to listen on socket socket-number. It therefore will not be able to communicate over this network port. Probably the port is already in use by another program or has been reserved by the TCP/IP configuration.
System action: The CIM server does not start.
System programmer response: Error code error-code and the last four digits of the reason code 0xreason-code point out the reason for the error. For a description of error error-text with error code error-code and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.
User response: Report this problem to your system programmer.
User response: None.

**CFZ08001W** CIM HTTP or HTTPS connector cannot connect to host:port. Connection failed.

**Explanation:** The CIM server or a CIM client could not connect to a CIM indication listener or CIM server on host:port.

**System action:** None.

**System programmer response:** None.

**User response:** Check that the target of the connection is a valid hostname, IP connectivity exists to that host and that a CIM indication listener or a CIM server is listening on port of the target system.

**CFZ08101E** Internal server error. Connection with IP address IP-address closed.

**Explanation:** An unrecoverable error occurred during the communication with the client connected by IP-address.

**System action:** The connection is closed.

**System programmer response:** This message provides the affected IP address. Look for a previous CFZ message describing details of the internal error.

**User response:** Contact your system programmer.

**CFZ09100I** TCP/IP temporary unavailable.

**Explanation:** The TCP/IP stack used by the CIM server is not available.

**System action:** The CIM server is waiting for a restart of the TCP/IP stack. The CIM server will be not able to handle any commands and requests until the restart of the TCP/IP stack has completed. Currently processed requests are terminated.

**System programmer response:** Restart the TCP/IP stack the CIM server was using. If this stack is no longer used, restart the CIM server.

**User response:** None.

**CFZ10024I** Unable to start the CIM server. CIM server is already running.

**Explanation:** The CIM server detects that another instance of the CIM server is already running. There can be only one running CIM server.

**System action:** None.

**System programmer response:** Do not start the CIM server again. If you want to start a new CIM server on the system, use the stop command at the system console (/p cfzcim) or look for the CIM server running in the UNIX System Services (/d omvs,a=all) and cancel the process (/c cfzcim).

**CFZ10025I** The CIM server is listening on HTTP port port-number.

**Explanation:** The CIM server is starting up and will listen on port port-number for incoming requests from clients. For information about how to configure the CIM server's HTTP connections see Chapter 9, "CIM server configuration," on page 53.

**System action:** None.

**System programmer response:** None.

**User response:** None.

**CFZ10026I** The CIM server is listening on HTTPS port port-number.

**Explanation:** The CIM server is starting up and will listen on port port-number for incoming requests from clients using SSL encryption. Please note that special TCP/IP configuration settings are required for enabling the CIM server to support SSL encryption for HTTPS. For information about how to configure the CIM server's HTTPS connections see "Configuring the CIM server HTTPS connection using AT-TLS" on page 29.

**System action:** None.

**System programmer response:** None.

**User response:** None.

**CFZ10028I** The CIM server is listening on the local connection socket.

**Explanation:** The CIM server is starting up and will listen for incoming requests from clients. For information about how to configure the CIM server's HTTP connections see Chapter 9, "CIM server configuration," on page 53.

**System action:** None.

**System programmer response:** None.

**User response:** None.

**CFZ10030I** Started CIM server version version.

**Explanation:** The CIM server is now started and accepts CIM client requests.

**System action:** None.

**System programmer response:** None.

**User response:** None.
CFZ10031I  CIM server - stopped.
Explanation: The CIM server is now stopped. CIM client requests are no longer accepted.
System action: None.
System programmer response: None.
User response: None.

CFZ10033I  The CIM server is not started: subsequent message.
Explanation: The CIM server was not started due to an error condition described in subsequent message.
System action: The CIM server is not started.
System programmer response: See the error condition as described in the subsequent message.
User response: Report this problem to your system programmer.

CFZ10034E  CIM server repository contains files with wrong tags. Unable to set file tags.
Stopping CIM server startup.
Explanation: The CIM server repository contains files tagged with the wrong CCSID. The CIM server tried to set the right CCSID (ISO8859-1) tag on this file, but was not successful.
System action: The CIM server stops.
System programmer response: Look for previously issued messages (CFZ10035E or equivalent LE messages) about access violations for path /var/wbem. Grant the denied access right to the user ID running the CIM server. Restart the CIM server.
User response: None.

CFZ10035E  Failed to change file tag for file-name. Error (error-number): error-message.
Explanation: The CIM server is not able to change the file tag for the file file-name. For the reason, see the system error number error-number and the system error message error-message.
System action: The CIM server stops.
System programmer response: Correct the reason for failing to change the file tag. The reason is indicated by the system error number error-number and the system error message error-message.
User response: None.

CFZ10036E  CIM server repaired file tags for number repository files.
Explanation: The CIM server was able to restore the correct CCSID (ISO8859-1) file tag for a number of repository files.
System action: None.
System programmer response: Repository file tags were missing or wrong. Please revise procedures handling files located in /var/wbem to preserve file tags. If file tags are preserved, this message will not be displayed again.
User response: None.

CFZ10037I  Failed to open repository directory repository-directory: error-text (error code error-code, reason code 0xreason-code).
Explanation: The CIM server is not able to open the directory repository-directory containing the repository.
System action: The CIM server does not start.
System programmer response: Error code error-code and the last four digits of the reason code 0xreason-code point out the reason for the error. For a description of error error-text with error code error-code and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.
User response: None.

CFZ10206W  No providers accepted the subscription.
Explanation: The subscription request for a CIM indication failed because there is no CIM indication provider that accepts the query contained in the indication filter. Either the filter contains an invalid or unsupported query, or an error has occurred during subscription processing.
System action: The indication subscription fails and the subscription is not persistent on the CIM server.
System programmer response: Check the z/OS console for other error messages that indicate the cause of the subscription failure.
User response: Check the query in the indication filter and make sure this query is supported by the target CIM server. If the problem persists contact the system programmer of the target system.

CFZ10215W  Subscription (name) in namespace namespace has no provider.
Explanation: During startup the CIM server has failed to re-establish a persistent CIM indication subscription because there is no CIM indication provider that accepts the query contained in the indication filter. Either the filter contains an invalid or unsupported
query, or an error has occurred during subscription processing.

System action: The subscription name is inactive.

System programmer response: Check the z/OS console for other error messages that indicate the cause of the subscription failure. Correct the error(s) and then restart the CIM server.

User response: None.

CFZ10405W Failed to deliver an indication:

message-details

Explanation: The CIM server was unable to deliver a CIM indication to a subscribed indication listener. See message-details for the potential cause.

System action: The CIM indication is not delivered and discarded.

System programmer response: Ensure the destination system of the indication subscription is available and reachable. To remove obsolete indication subscriptions use the cimsub command (see “cimsub” on page 110).

User response: None.

CFZ12500E Not loading dynamic load library library-name due to missing program control flag.

Explanation: The CIM server runs on a system with Enhanced Security and thus does not load dynamic libraries which are not audited by a system programmer.

System action: The system does not load the named dynamic library.

System programmer response: Set the program control flag on the dynamic library using the UNIX System Services command extattr +p <filename>.

User response: Contact a system programmer to audit the dynamic library and set the program control flag.

CFZ12501E Security profile CIMSERV in CLASS WBEM must be defined. Ending CIM server.

Explanation: The CIM server detected an incomplete security setup.

System action: The CIM server does not start.

System programmer response: Complete the security setup by defining the profile CIMSERV in class WBEM. Refer to Chapter 6, “CIM server security setup,” on page 23 for further details.

User response: Contact your system programmer.

CFZ12502E CIM server user ID requires either READ access to BPX.SERVER or must be UID 0. Ending CIM server.

Explanation: The CIM server user ID must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12503E CIM server address space dirty due to loading from a not program controlled load library. Ending CIM server.

Explanation: The CIM server loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer’s audit.

System action: The CIM server stops.

Programmer response: Check all dynamic libraries for their program control flag and ensure that no library changed. Make sure that the Language Environment® libraries SCEERUN and SCEERUN2 are program controlled.

User response: Contact your system programmer.

CFZ12504E CIM server does not have appropriate privileges to check SAF security environment. Ending CIM server.

Explanation: The CIM server user ID must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be privileged.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12505E CIM server user ID requires either READ access to BPX.SERVER or must be UID 0. Ending CIM server.

Explanation: The CIM server user ID must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be privileged.

System action: The CIM server stops.
System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

---

CFZ12506E CIM server address space dirty due to loading from a not program controlled load library. Ending CIM server.

Explanation: The CIM server has loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer's audit.

System action: The CIM server stops.

Programmer response: Check all dynamic libraries for their program control flag and ensure that no library has changed. Make sure the Language Environment libraries SCEERUN and SCEERUN2 are program controlled.

User response: Contact your system programmer.

---

CFZ12507W CIM server does not have surrogate for client user ID user-ID.

Explanation: A request sent from the user ID could not be processed. The CIM server does not have access to act as surrogate for the requesting user ID.

System action: The user request is ignored and an error message is sent to the client.

System programmer response: To permit the CIM server user ID to act as a surrogate for the client user, grant the user ID running the CIM server READ access to the RACF profile BPX.SRV.user-ID as described in “Switching identity (surrogate)” on page 29.

User response: Contact your system programmer.

---

CFZ12508W Failure error-number deleting thread security.

Explanation: The CIM server was not able to delete the thread level security built for a specific request.

System action: None.

System programmer response: None.

User response: None.

---

CFZ12509E The CIM server user ID requires either READ access to BPX.SERVER or must be UID 0. Stopping CIM server startup.

Explanation: The user ID that starts the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server does not start.

System programmer response: To complete the security setup, define the profile CIMSERV in class WBEM. Refer to Chapter 6, “CIM server security setup,” on page 23 for further details.

User response: Contact your system programmer.
CFZ12513E The CIM server user ID requires CONTROL access to security profile CIMSERV in CLASS WBEM. Stopping CIM server startup.

Explanation: The CIM server user ID requires CONTROL access to security profile CIMSERV in CLASS WBEM.

System action: The CIM server does not start.

System programmer response: To permit the CIM server user ID to perform administrative CIM tasks, give it CONTROL permission to profile CIMSERV in class WBEM. Refer to Chapter 6, "CIM server security setup," on page 23 for further details.

User response: Contact your system programmer.

CFZ12514E Security profile profile-name in CLASS WBEM must be defined.

Explanation: A provider defined a security profile at registration that is not defined for RACF class WBEM.

System action: None.

System programmer response: Create the RACF profile in class WBEM and permit users who should have access to the provider. Verify if the security profile is defined for RACF and make sure that the class WBEM has been refreshed. Verify if the provider really should be registered with the mentioned security profile and if it should be checked.

User response: Contact your system programmer.

CFZ12515W User user-ID not authorized to perform intrinsic CIM operation operation against provider provider-name. access-type access to profile-name in CLASS WBEM required.

Explanation: User user-ID is not authorized to perform CIM operation operation involving the provider provider-name. The user needs access-type access to SAF security profile profile-name that is defined in class WBEM.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Verify if the user should be permitted to perform the current request. If so, grant the user access-type access to the profile profile-name.

User response: Contact your system programmer.

CFZ12516E CIM server does not have appropriate privileges to check SAF security environment. Ending CIM server.

Explanation: The CIM server user ID must have READ access to the security profile BPX.SERVER, or if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: None.

CFZ12517E Missing IdentityContainer (no username) in request.

Explanation: The security component of the CIM server detected an invalid operation context that does not contain a username.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: None.

User response: None.

CFZ12519E An unexpected error occurs: error-text (error number error-number, reason code 0xreason-code). Stopping CIM server startup.

Explanation: During startup, the CIM server received the unrecoverable error error-text. For a description of error error-text with error number error-number and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.

System action: The CIM server does not start.

System programmer response: Error number error-number and the last four digits of the reason code 0xreason-code point out the reason for the error. Check the console for more messages indicating the problem.

User response: Contact your system programmer.

CFZ12520E CIM server did not set Must Stay Clean. Stopping provider agent startup.

Explanation: The provider agent determined that the Must Stay Clean flag was not set. The provider agent startup is not processed by the CIM server.

System action: The provider agent does not start.

System programmer response: Ensure that the provider agent can only be started by the CIM server.

User response: Contact your system programmer.
An unexpected error occurs: error-text (error number error-number, reason code X'\text{reason-code}'. Stopping provider agent startup.

**Explanation:** During startup, the provider agent received the unrecoverable error error-text. For a description of error error-text with error number error-number and the last four digits of the reason code X'\text{reason-code}', see [z/OS UNIX System Services Messages and Codes](https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.bpx.doc/) or enter the reason code in the BPXMTXT TSO command.

**System action:** The provider agent does not start.

**System programmer response:** Error number error-number and the last four digits of the reason code X'\text{reason-code}' point out the reason for the error. Check the console for more messages indicating the problem.

**User response:** Contact your system programmer.

---

CIM Runtime Environment user ID requires either READ access to BPX.SERVER or has to be UID 0. Stopping provider agent startup.

**Explanation:** The user ID that runs the provider agent must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

**System action:** The provider agent does not start.

**System programmer response:** Permit the user ID to run the provider agent by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

**User response:** Contact your system programmer.

---

Provider agent address space dirty due to loading from a not program controlled load library. Stopping provider agent startup.

**Explanation:** The provider agent has loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer's audit.

**System action:** The provider agent does not start.

**System programmer response:** Check all dynamic libraries for their program control flag and ensure that no library has changed. For details on program control look at [z/OS UNIX System Services Planning](https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.bpx.doc/) and [z/OS Security Server RACF Security Administrator's Guide](https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.racf CLI.doc).

**User response:** Contact your system programmer.

---

CIM Runtime Environment does not have appropriate privileges to check SAF security environment. Stopping provider agent startup.

**Explanation:** The user ID that runs the provider agent must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

**System action:** The provider agent does not start.

**System programmer response:** Permit the user ID to run the provider agent by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

**User response:** Contact your system programmer.

---

Unsupported UserContext value: "value".

**Explanation:** A provider module was registered with a UserContext value of value, but that value is not supported by this version of the CIM server. Valid values are 2 ("Requestor") and 3 ("Designated User").

**System action:** The addressed provider module is not correctly registered. The request fails and an error is sent back to the requestor.

**System programmer response:** Identify the failing provider module, remove the provider using the **cimprovider** utility (see "cimprovider" on page 81) and re-register the provider with a correct provider registration MOF.

**User response:** Contact your system programmer.

---

Missing DesignatedUserContext property in PG_ProviderModule instance.

**Explanation:** A provider module was registered with a UserContext value of 3 ("Designated User"). The user ID of the designated user has to be specified in DesignatedUserContext, but no value was found (see "PG_ProviderModule" on page 281).

**System action:** The request that is directed against the provider module in error will fail and an error is sent back to the requestor.

**System programmer response:** Identify the failing provider module, remove the provider using the **cimprovider** utility (see "cimprovider" on page 81) and re-register the provider with a correct provider registration MOF.

**User response:** Contact your system programmer.

---

Cannot switch to designated user user-ID. User is unknown to the security product, or has no OMVS segment.

**Explanation:** The CIM server failed to switch the security context to user-ID for a provider configured
with a designated user context. The user user-ID defined for the provider’s security context is not defined to the system or does not have an OMVS segment.

System action: The request fails and an authorization error is sent back to the requestor/client.

System programmer response: Check if the user user-ID is the right user ID to run with or check for the existence of the user user-ID within your security product with the appropriate OMVS segment. If the problem persists you may want to remove the failing provider using the cimprovider utility and re-register the provider with the correct designated user defined in the provider registration MOF.

User response: None.

CFZ12529E An unexpected error occurred when switching to user user-ID: error-text (error code error-code , reason code 0xreason-code).

Explanation: The CIM server failed to switch to user-ID for the designated user context of a provider.

System action: The request fails and an authorization error is sent back to the requestor/client.

System programmer response: Error code error-code and the last four digits of the reason code 0xreason-code point out the reason for the error. For a description of error error-text with error code error-code and the last four digits of the reason code 0xreason-code, see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.

User response: None.

CFZ12530E Cannot switch to user user-ID because a SAF authorization error occurred. For the reason, see the SAF RACROUTE EXTRACT service reason code reason-code.

Explanation: A SAF authorization error has occurred. The message returns the SAF specific reason code. For RACF, the two rightmost bytes contain the RACF return code and the RACF reason code.

Example: For reason code 0x0BE80820, the RACF return code is 08 and the RACF reason code is 20.

System action: The CIM server terminates the user request.

System programmer response: Use the reason-code for your SAF RACROUTE EXTRACT service to find more details to resolve the authorization error.

For RACF: For details of the authorization error, use the RACF return code and reason code. See z/OS UNIX System Services Programming: Assembler Callable Services Reference, table ‘RACF return and reason codes’, for the specific reason of the failure.

User response: Report this problem to your system programmer.

CFZ12531I CIM server successfully registered to ARM using element name CFZ_SRV_system-name.

Explanation: The CIM server successfully registered to the Automatic Restart Manager.

System action: None.

System programmer response: None.

User response: None.

CFZ12533I CIM server failed to register with ARM using element name CFZ_SRV_system-name: return code X'error-number', reason code X'reason-code'.

Explanation: The CIM server failed to register with the Automatic Restart Manager using the element name CFZ_SRV_system-name.

System action: None.

System programmer response: If you do not want to use the Automatic Restart Manager, you can ignore this message. If you want to use ARM, use X'error-number' and X'reason-code' to look up the return and reason codes for the IXCARM macro in the z/OS MVS Programming: Sysplex Services Reference for the reason to fail to register with ARM.

User response: None.

CFZ12534W Authorization failed: User ID user-ID does not have CONTROL permission to profile CIMSERV CL(WBEM).

Explanation: The user ID requesting an administrative task, for example, cimconfig or cimprovider, does not have the required permission.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Permit the user to perform administrative CIM tasks by giving him CONTROL permission to profile CIMSERV in class WBEM.

User response: Contact your system programmer.

CFZ12535W Authorization error: User ID user-ID cannot run the requested CIM operation because it lacks UPDATE permission to profile CIMSERV CL(WBEM).

Explanation: A client with the named user ID has sent a CIM request for a CIM write operation (SetProperty, InvokeMethod, CreateInstance, ModifyInstance,
DeleteInstance) to the CIM server without having the appropriate access rights.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: None.

User response: If you need to perform CIM write operations, ask your system programmer to grant you at least UPDATE access to profile CIMSERV CL(WBEM).

CFZ12540E ATTLS reset the connection due to handshake failure. Connection closed.

Explanation: ATTLS reset the connection with the client due to a handshake failure.

System action: The connection is closed.

System programmer response: This message documents an unsuccessful connect to AT-TLS. If this prevents a connection from a client to the server, switch on tracing at the AT-TLS policy to find the reason for this closure.

User response: Contact your system programmer.

CFZ12541E An unexpected error occurs: error-text (error number error-number, reason code X'reason-code'). Connection closed.

Explanation: While querying the AT-TLS connection using ioctl(), the CIM server received an unknown error. For a description of error error-text with error number error-number and the last four digits of the reason code X'reason-code', see z/OS UNIX System Services Messages and Codes or enter the reason code in the BPXMTEXT TSO command.

System action: The connection is closed.

System programmer response: Contact IBM support.

User response: Contact your system programmer.

CFZ12542E ATTLS policy is not active for the CIM server HTTPS port. Communication not secured. Connection closed.

Explanation: The CIM server is configured to use HTTPS by defining the configuration property enableHttpsConnection, but the AT-TLS policy is not configured correctly for the CIM server.

System action: The connection is closed.

System programmer response: Please refer to Chapter 6, "CIM server security setup," on page 23 for information about how to configure AT-TLS for the CIM server.

User response: Contact your system programmer.

CFZ12543E ATTLS policy not valid for CIM server.

Connection closed.

Explanation: The value of the property ApplicationControlled defined in the AT-TLS policy for the CIM server is 0. Hence, the CIM server is only aware of AT-TLS but does not control it.

System action: The connection is closed.

System programmer response: Change the property ApplicationControlled to 0FF in the AT-TLS policy defined for the CIM server. Refer to Chapter 6, "CIM server security setup," on page 23 for information about how to configure AT-TLS for the CIM server.

User response: None.

CFZ12544E ATTLS policy specifies the wrong HandshakeRole for the CIM server HTTPS port. Communication not secured. Connection closed.

Explanation: The property HandshakeRole defined in the inbound AT-TLS policy for the CIM server is not configured correctly.

System action: The connection is closed.

System programmer response: Change the property HandshakeRole to ServerWithClientAuth or to the server at the inbound AT-TLS policy defined for the CIM server. Refer to Chapter 6, "CIM server security setup," on page 23 for information about how to configure AT-TLS for the CIM server.

User response: None.

CFZ12545E Automatic repository upgrade failed at step step-number. Stopping CIM server startup.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. No actual migration action was run, because the basic setup is not correct.

System action: The CIM server does not start.

System programmer response: To find out the reason for this error, check the previously issued message. Correct the basic setup and restart the CIM server.

User response: None.


Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Migration started and ran to a certain point. Though they failed, the accomplished migration
actions were successfully rolled back.

**System action:** The CIM server does not start.

**System programmer response:** To find out the reason for this error, check the previously issued message. Correct the setup problem and restart the CIM server.

**User response:** Contact your system programmer.

---


**Explanation:** The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Migration started and ran into a critical break. The attempt to roll back the taken actions failed. Manual user intervention is required to roll back taken migration actions.

**System action:** The CIM server does not start.

**System programmer response:** To find out the reason for this error, check the previously issued message. Rollback the taken migration actions as described in that message. Fix the setup problem and restart the CIM server.

**User response:** None.

---

**CFZ12548E** Failed to initiate command: command with error: error-number.

**Explanation:** The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Processing of the named command failed with error error-number.

**System action:** The CIM server will roll back already taken migration actions. The CIM server does not start.

**System programmer response:** Investigate why the named command cannot perform successfully. Fix the system setup and restart the CIM server.

You can find further details in STDERR and STDOUT of the job output.

**User response:** Contact your system programmer.

---

**CFZ12549E** Failed to rename directory source-directory-name to target-directory-name with error: error-number.

**Explanation:** The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Renaming of source directory to target directory failed.

**System action:** The CIM server will roll back already taken migration actions. CIM server does not start.

**System programmer response:** Investigate the reason of the renaming failure. Possible reasons are missing file access rights, a full file system or missing access right to run a program in an extra UNIX System Services address space.

**User response:** Contact your system programmer.

---

**CFZ12550E** Failed to create repository status files with: error-text.

**Explanation:** The CIM server failed to write the repository status file while automatically migrating the old repository in /var/wbem to the new schema level. The migration is nearly complete, but writing the repository status file failed. The repository status file serves to avoid repeated attempts to migrate the repository.

**System action:** A message is logged to the system console. The CIM server startup continues.

**System programmer response:** Either fix the reason for the failed write of the repository status file and stop and restart the CIM server, or copy the file supplied in /usr/lpp/wbem/ to /var/wbem.

**User response:** Contact your system programmer.

---

**CFZ12551E** Command command failed with status status-code.

**Explanation:** The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. The processing of the named command failed with status status-code.

**System action:** The CIM server will roll back the already taken migration actions. CIM server does not start.

**System programmer response:** Investigate why the named command cannot perform successfully. Fix the system setup and restart the CIM server.

Further details can be found in STDERR and STDOUT of the job output.

**User response:** Contact your system programmer.

---

**CFZ12552I** Starting automatic repository upgrade.

**Explanation:** The CIM server will start to migrate the old repository to the new schema level.

**System action:** The CIM server starts to migrate the repository.

**System programmer response:** None.

**User response:** None.
CFZ12554E  Error during automatic repository upgrade. No reference repository found at directory-name.

Explanation: The CIM server could not locate the new repository at location directory-name. No actual migration action was run, because basic setup is not correct.

System action: The CIM server does not start.

System programmer response: Check the SMP/E installation. Directory and files should have been copied to the named location in the SMP/E APPLY step.

User response: Contact your system programmer.

CFZ12555E  Rename of previous repository to directory-name failed.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Even though migration successfully created the new repository, renaming the old repository for backup failed.

System action: The CIM server will remove the new repository to roll back the taken migration actions. The CIM server does not start.

System programmer response: Investigate why the CIM server was unable to rename the directory /var/wbem/repository to the directory directory-name. Probable causes are insufficient disk space or missing access rights.

User response: None.

CFZ12556E  Rename of new repository to directory-name failed.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Even though migration successfully created the new repository and backed up the old repository, renaming the new repository to /var/wbem/repository failed.

System action: The CIM server tries to roll back the taken migration actions and also removes the new repository and renames the backed up version to /var/wbem/repository.

System programmer response: Investigate why the CIM server was unable to rename the directory. Probable reasons are insufficient disk space or missing access rights. If fallback actions fail (indicated by message CFZ12547E), manually remove the directory named /var/wbem/repository_new and rename the latest backed up repository version to /var/wbem/repository.

User response: None.

CFZ12557E  Failure during automatic repository upgrade. Trying to recover.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level.

System action: The CIM server will try to roll back the taken migration actions.

System programmer response: Check former and further messages for details and possible required actions.

User response: None.

CFZ12558E  Failed to remove incomplete new repository at directory-name.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. Removing the new, migrated repository failed.

System action: The CIM server does not start.

System programmer response: Remove the directory /var/wbem/repository_new and its subfolders and files. Check the system log for earlier messages for details on the actual migration step that failed. Fix the situation and restart the CIM server. The most common reason for this problem is insufficient disk space at /var/wbem.

User response: None.

CFZ12559E  Failed to restore previous repository on recovery. Manual rename of source-directory-name back to target-directory-name required!

Explanation: The CIM server tried to roll back the migration actions. Renaming the backed up copy of the old repository to target directory name failed.

System action: The CIM server does not start.

System programmer response: Rename the source directory to the target directory name. Investigate the reason for the failure of the automatic repository migration by checking the system log for former error messages. Fix the system setup and restart the CIM server.

User response: Contact your system programmer.

CFZ12560E  Failed to create repository status file directory-name. Manual intervention required!

Explanation: The CIM server failed to write the repository status file while automatically migrating the old repository in /var/wbem to the new schema level. The migration is nearly complete, but writing the repository status file failed. The repository status file serves to avoid repeated tries to migrate the repository.
System action: A message is logged to the system console. The CIM server startup continues.

System programmer response: Either fix the reason for the failed write of the repository status file and stop and restart the CIM server, or copy the file supplied in /usr/lpp/wbem/ to /var/wbem.

User response: Contact your system programmer.

---

CFZ12561E Repository in directory directory-name is backlevel. Run migration job for repository upgrade.

Explanation: The CIM server failed to automatically migrate the old repository in /var/wbem to the new schema level. No actual migration action was run, because the basic setup is not correct. The old repository found at directory-name is not a z/OS 1.8 level repository.

System action: The CIM server does not start.

System programmer response: Use migration job CFZRCUST to migrate the repository.

User response: Contact your system programmer.

---

CFZ12562I Previous repository was renamed to directory-name for backup and can be removed.

Explanation: The CIM server successfully migrated the old repository to the new schema level. A backup copy of the old repository is stored at directory-name. The copy should be backed up and then can be deleted to free up disk space.

System action: The CIM server startup continues.

System programmer response: You may want to backup the old repository, and delete the copy on hard disk.

User response: None.

---

CFZ12563I Automatic repository upgrade completed successfully.

Explanation: The CIM server successfully migrated the old repository to the new schema level.

System action: The CIM server startup continues.

System programmer response: None.

User response: None.

---

CFZ12564W Failed to obtain information about file system path-name. Error: error-text.

Explanation: The CIM server failed to determine information about the file system at path-name. The cause of the failure was error error-text.

System action: Automatic repository upgrade continues.

System programmer response: None.

User response: None.

---

CFZ12565W File system at path-name is smaller than the recommended 102400 KB (100MB).

Explanation: The file system available at path-name should be at least 100MB large or be able to extend to that size. The CIM server might run out of space when automatically upgrading the repository.

System action: Automatic repository upgrade continues.

System programmer response: Make sure that there is enough space for data to be stored in the file system at path-name. Recommended is a system specific data set with at least 100MB space mounted at /var/wbem.

User response: None.

---

CFZ12566W Less free space than 61440 KB (60MB) available on file system path-name.

Explanation: The CIM server detected less than 60MB space available in the file system path-name. The CIM server might run out of space when automatically upgrading the repository.

System action: Automatic repository upgrade continues.

System programmer response: Make sure that there is enough space for data to be stored in the file system at path-name. Recommended is a system specific data set with at least 60MB space mounted at /var/wbem.

User response: None.

---

CFZ12568E ATTLS is not active for TCP-IP stack the CIM server is using for HTTPS connections. Communication not secured. Connection closed.

Explanation: The CIM server is configured to use HTTPS by defining the configuration property enableHttpsConnection, but the Communication Server Policy Agent was not enabled on the stack the CIM server is using when AT-TLS policy mapping was performed for the connection.

System action: The connection is closed.

System programmer response: Ensure that Communication Server Policy Agent is configured for the TCP/IP stack the CIM server is listening. Please refer to Chapter 6, “CIM server security setup,” on page 23 for information about how to configure AT-TLS for the CIM server.

User response: Contact your system programmer.
CFZ12569E There is no ATTLS policy found for the CIM server HTTPS connections. Communication not secured. Connection closed.

Explanation: The CIM server is configured to use HTTPS by defining the configuration property `enableHttpsConnection`, but the Communication Server Policy Agent did not find an AT-TLS policy for the CIM server when AT-TLS policy mapping was performed for the connection.

System action: The connection is closed.

System programmer response: Ensure that a Communication Server Policy Agent policy is defined for CIM server. Please refer to Chapter 6, “CIM server security setup,” on page 23 for information about how to configure AT-TLS for the CIM server.

User response: Contact your system programmer.

CFZ12570I Created directory `/var/wbem`.

Explanation: CIM server successfully created the directory `/var/wbem`.

System action: None.

System programmer response: None.

User response: None.

CFZ12571E Failed to create directory `/var/wbem` with error: error-message. Stopping CIM server startup.

Explanation: CIM server failed to create the directory `/var/wbem` with error `error-message`. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to create the environment variable setup file to remove all quotes.

System action: CIM server does not start.

System programmer response: Remove all quotes in file `file-name` manually or check the error condition described by `error-message` and fix the indicated problem in the system setup. Restart the CIM server.

User response: Contact your system programmer.

CFZ12572W Failed to read repository status file: error-message.

Explanation: CIM server failed to read information from the repository status file at `/var/wbem`.

System action: CIM server startup proceeds and the repository is automatically migrated to the latest level available from `/usr/lpp/wbem`.

System programmer response: Check the error condition described by `error-message` and fix the indicated problem in the system setup.

User response: Contact your system programmer.

CFZ12573F Failed to write all data to `file-name` file.

Explanation: CIM server failed to write to `file-name` for the reason named in `error-message`. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to write the environment variable setup file with all quote removed, but the file was written partially.

System action: CIM server does not start.

System programmer response: Check the error condition described by `error-message` and fix the indicated problem in the system setup. Create a new environment variables setup file `file-name` using the default shipped in `/usr/lpp/wbem/install`. Restart the CIM server.

User response: Contact your system programmer.

CFZ12574W File `file-name` contains quotes which should be removed. Removing quotes and stopping CIM server startup. Restart the CIM server.

Explanation: CIM server found quote characters in file `file-name`. Quotes can cause environment variable setup problems.

System action: CIM server tries to remove all quotes. The CIM server does not start.

System programmer response: None.

User response: Restart the CIM server.

CFZ12575E Failed to open `file-name` for write with error: `error-message`.

Explanation: CIM server failed to open `file-name` for writing. The reason is named in `error-message`. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to open the environment variable setup file to remove all quotes.

System action: CIM server does not start.

System programmer response: None.

User response: None.

CFZ12576F Failed to write all data to `file-name` file.

Explanation: CIM server failed to write to `file-name` for the reason named in `error-message`. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to write the environment variable setup file with all quote removed, but the file was written partially.

System action: CIM server does not start.

System programmer response: None.

User response: None.

CFZ12577I Successfully removed all quotes from `file-name`.

Explanation: CIM server removed all quote characters from file `file-name`. Quotes can cause environment variable setup problems for the started task procedure. To avoid issues caused by partially setup environment variables the CIM server is stopped and needs to be restarted.

CFZ12578I Created directory `/var/wbem`.

Explanation: CIM server successfully created the directory `/var/wbem`.

System action: None.

System programmer response: None.

User response: None.

CFZ12579E Failed to create directory `/var/wbem` with error: error-message. Stopping CIM server startup.

Explanation: CIM server failed to create the directory `/var/wbem` with error `error-message`. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to create the environment variable setup file to remove all quotes.

System action: CIM server does not start.

System programmer response: None.

User response: None.
System action: CIM server does not start.
System programmer response: Restart the CIM server.
User response: Restart the CIM server.

CFZ12578W Directory /var/wbem does not exist. CIM server will create it.
Explanation: On CIM server startup the automated migration procedure detected that path /var/wbem does not exist.
System action: CIM server creates the directory /var/wbem.
System programmer response: None.
User response: None.

CFZ12579W Failed switching to zIIP mode, RC=returncode. CIM server running on CP.
Explanation: An error occurred when the CIM server process tried to establish eligibility for running on zIIP processors.
RC=0x00000408 and RC=0x00000508 indicate a problem with the CIM server installation in the z/OS UNIX file system.
RC=0x00000708 indicates that CIM server library libcfzsys.so located in /usr/lpp/wbem/lib is not APF authorized.
System action: The CIM server process with all its threads is executing on CP processors.
System programmer response: For RC=0x00000708, use the command extattr +a /usr/lpp/wbem/lib/libcfzsys.so to restore the extended attribute to APF authorize the library.

All other return codes indicate a general problem during program execution, please contact IBM for service.
User response: Contact your system programmer.

CFZ13006W Request user ID user-ID doesn’t have READ permission to profile CIMSERV CL(WBEM).
Explanation: The user ID requesting a CIM operation using a remote connection is not permitted to use the CIM server.
System action: The request is not processed and an "Access Denied" notification is sent to the client.
System programmer response: Permit the user to perform CIM requests by giving the user ID READ access to profile CIMSERV CL(WBEM).
User response: Contact your system programmer to permit your user ID to perform CIM requests. Repeat your request.

CFZ13007W Request user ID user-ID doesn’t have READ permission to profile CIMSERV CL(WBEM).
Explanation: The user ID requesting a CIM operation using a local connection is not permitted to use the CIM server.
System action: The CIM request is not processed and an "Access Denied" notification is sent to the client.
System programmer response: Permit the user to perform CIM requests by giving the user ID READ access to profile CIMSERV CL(WBEM).
User response: Contact your system programmer to permit your user ID to perform CIM requests. Repeat your request.

CFZ13607E CIM server cannot execute out-of-process provider agent: error-text (error number error-number, reason code X'reason-code').
Explanation: The CIM server failed to process the out-of-process provider agent caused by the problem error-text. For further details, see the description of error number error-number and the last four digits of the reason code X'reason-code' in z/OS UNIX System Services Messages and Codes.
System action: None.
System programmer response: Stop the CIM server. Error number error-number and the last four digits of the reason code X'reason-code' point out the reason for the error. Check the console for more messages indicating the problem.
User response: Contact your system programmer.
CFZ17200W Authentication failed for user user-ID.

Explanation: The authentication for user user-ID against the z/OS system failed. Either the user ID or password contained in a request was invalid or revoked, or the user ID has not been authorized to use CIM.

System action: The CIM request is denied.

System programmer response: None.

User response: Check that you are using a valid user ID and password and that the user ID has been authorized to use CIM. If the problem persists, contact the system programmer of the target system to check for more detailed authentication error messages on the system console.

CFZ17201W Authentication failed for user user-ID because enableRemotePrivilegedUserAccess is not set to true.

Explanation: The CIM server refused login for user user-ID, because user-ID is a superuser (UID=0), and the current CIM server configuration prohibits superuser logins (the configuration option enableRemotePrivilegedUserAccess is false).

System action: The CIM request is denied.

System programmer response: To allow superuser logon to the CIM server set the enableRemotePrivilegedUserAccess configuration option to true, as described in Chapter 9, “CIM server configuration,” on page 53.

User response: Either use a non-superuser user ID for logon to the CIM server, or contact your system administrator to enable superuser login for the CIM server.

CFZ17202W Request user ID user-ID doesn't have READ permission to profile CIMSERV CL(WBEM).

Explanation: The user ID requesting a CIM operation using a remote connection is not permitted to use the CIM server.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Permit the user to perform CIM requests by giving the user READ access to profile CIMSERV CL(WBEM).

User response: Contact your system programmer to permit your user ID to perform CIM requests and afterwards repeat your request.

CFZ17203W Request user ID user-ID misses password.

Explanation: A request was sent to the CIM server with user user-ID but no password was specified.

System action: The request is rejected as unauthorized.

System programmer response: None.

User response: Specify a password with your request.

CFZ17204I CIM server authentication is using application ID OMVSAPPL.

Explanation: The CIM server is using the application ID 'OMVSAPPL' for authentication.

System action: Application ID 'OMVSAPPL' is used for authentication.

System programmer response: If the usage of application ID 'OMVSAPPL' is intended, no action has to be taken.

Otherwise, if you want to use the application ID 'CFZAPPL',
1. Set the configuration property enableCFZAPPLID to true
2. Restart the CIM server

User response: None.

CFZ17205W Authentication failed for user user-ID from client IP address IP-address.

Explanation: The authentication for user user-ID issued by the IP address IP-address against the z/OS system failed. Either the user ID or password contained in a request was invalid or revoked, or the user ID has not been authorized to use CIM.

System action: The CIM request is denied.

System programmer response: None.

User response: Check that you are using a valid user ID and password and that the user ID has been authorized to use CIM. If the problem persists, contact the system programmer of the target system to check for more detailed authentication error messages on the system console.

CFZ17400W Request user ID user-ID does not have READ permission to profile CIMSERV CL(WBEM).

Explanation: The user ID requesting a CIM operation using a local connection is not permitted to use the CIM server.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Permit the user to
perform CIM requests by giving the user READ access to profile CIMSERV CL(WBEM).

User response: Contact your system programmer to permit your user ID to perform CIM requests and afterwards repeat your request.

CFZ17600E Change owner action of security token file failed, which is required for local authentication.

Explanation: The CIM server cannot change the ownership of a file to the user requesting local authentication. The file is located at /tmp and the file name matches the pattern: cimclient_<USERID>_* . The file is only valid for a short time. The server should remove this file automatically. It can be deleted.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Either define CHOWN.UNRESTRICTED in RACF, or grant the CIM server runtime environment user ID READ access to the SUPERUSER.FILESYS.CHOWN resource in the UNIXPRIV RACF class. For details refer to "Configuring the CIM server's resource authorization model" on page 25.

User response: Contact your system programmer.

CFZ17805I Audit logging is enabled.

Explanation: Audit logging is enabled.

System action: The CIM server starts writing SMF 86 records. These records are only recorded if the SMF configuration contains record 86 and the security is set up accordingly. For details see "Audit logging with SMF record 86" on page 69.

System programmer response: None.

User response: None.

CFZ17806I Audit logging is disabled.

Explanation: Audit logging is disabled.

System action: The CIM server stops writing SMF 86 records.

System programmer response: None.

User response: None.

CFZ18202E CIM server registration with internal SLP failed.

Explanation: The CIM server failed to register itself as a service for the Service Location Protocol (SLP). Clients will not be able to detect the CIM server on the local networking using the SLP protocol.

System action: None.

System programmer response: Check the system log for further messages indicating CIM server configuration problems or general communication problems. This message usually indicates an issue with the CIM server setup.

User response: None.

CFZ18204I SLP registration initiated.

Explanation: The CIM server has successfully registered itself as a service for the Service Location Protocol (SLP). Clients using the SLP protocol can now detect this CIM server on the local network.

System action: None.

System programmer response: None.

User response: None.

CFZ18603E Could not get CLASSPATH from environment.

Explanation: Initialization of the Java Virtual Machine failed due to environment variable CLASSPATH not being set. The CIM client request cannot be answered as JMPI (Java Managed Provider Interface) providers do not run without a correctly set CLASSPATH.

System action: None.

System programmer response: Set the CLASSPATH as described by the provider.

User response: Contact your system programmer.

CFZ20400E A system error occurred. Retry the WS-Management operation at a later time.

Explanation: A WS-Management operation exceeds the server's memory.

System action: Stop the WS-Management operation.

System programmer response: Look for message CFZ08101E identifying the source of the WS-Management request. Contact the owner of the application issuing the request and analyze the reason for the size of the operation. Limit the result objects for this request. Restart the server to clean it up.

User response: Contact your system programmer.

IWMCP001E Internal error.

Explanation: An unspecified internal error occurred. The requested operation could not be completed.

System action: No action was performed.

System programmer response: None.

User response: No action required. The function may be successful if invoked again.
IWMCP002E Severe internal error.

Explanation: An unspecified internal error occurred. The requested operation might have been partly or completely processed.

System action: Operation was partly or fully completed.

System programmer response: None.

User response: Check the system state. If the operation was not fully completed, the function may be successful if invoked again.

IWMCP003E Memory shortage.

Explanation: Storage is not available for the requested operation. The requested operation could not be performed.

System action: No action was performed.

System programmer response: None.

User response: There is a storage shortage. The function may work successfully later on.

IWMCP004E Module IWMP2PCS missing.

Explanation: Unsupported operating system environment. The WLM CIM provider requires z/OS V1R10 or higher. It cannot be used on z/OS V1R9 or lower.

System action: No action was performed.

System programmer response: Install WLM CIM provider on z/OS V1R10 or higher.

User response: None.

IWMCP005E Invalid or missing parameter.

Explanation: One or several CIM provider method parameters are not valid.

System action: No action was performed.

System programmer response: None.

User response: Check the parameters passed to CIM provider methods.

IWMCP006E Insufficient access rights.

Explanation: The caller is not authorized to perform the requested operation. The RACF facility class is active and a profile has been defined for the MVSADMIN.WLM.POLICY RACF facility class profile to which the caller does not have sufficient read or update access.

System action: No action was performed.

System programmer response: Grant user appropriate access for RACF profile MVSADMIN.WLM.POLICY.
Part 7. Appendixes
Appendix A. Troubleshooting

This chapter contains the following subsections:

- "Garbage on the screen"
- "Typical error scenarios"

For problem determination, you can switch on tracing and logging. For details, see

- "Tracing" on page 64
- "Logging" on page 67

You can find further helpful information in Chapter 18, “z/OS specific messages,” on page 295.

Garbage on the screen

Since the z/OS CIM server and all of its command-line utilities operate in the enhanced ASCII environment, all output is written using ASCII encoding. This can lead to garbage being displayed when watching the output from the CIM server command-line utilities, sample programs or from the CIM server itself. By default, the configuration files `cimserver.env` and `profile.add` shipped with the CIM server provide the required settings for automatic conversion to the correct encoding. For details on how to enable the automatic conversion and about Enhanced ASCII in general, refer to [Using Enhanced ASCII functionality in z/OS UNIX System Services Planning](#).

One important issue is that automatic conversion so far only occurs for [UNIX System Service](#) applications. When the output of the CIM server or any of its clients should be consumed or displayed by applications other than [UNIX System Services](#) applications, the conversion must take place when the data are created. To achieve this, the output files need to be tagged as EBCDIC so that, for example, the CIM server's output is converted to EBCDIC before it is consumed by these applications.

Typical error scenarios

The following is a list of typical errors that can be observed when working with CIM:

Error:

BPXP014I ENVIRONMENT MUST REMAIN CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.

BPXP015I HFS PROGRAM /usr/lpp/wbem/provider/<provider_library> IS NOT MARKED PROGRAM CONTROLLED.

The provider `<provider_library>` is not marked program controlled.

When or where seen: Messages on the console.

Solution: Mark the dynamic load library `/usr/lpp/wbem/provider/<provider_library>` as program controlled by using the command `extattr +p <fully qualified dynamic load library name>`. Restart the CIM server and try again.

Error: CIM_ERR_ACCESS_DENIED
Access to a CIM resource was not available to the client: "Not authorized to run <name of a CIM operation> in the namespace root/PG_Internal"

When or where seen: Client application / Details in the CIM server trace log

Solution: Permit the user ID to execute a configuration command with CONTROL access to Security profile CIMSERV in class WBEM.

Error: CIM runtime environment user ID requires CONTROL access to profile CIMSERV in class WBEM.

When or where seen: The CIM server error log after CIM server fails to start

Solution: The CIM server startup fails because the CIM server user ID fails to have CONTROL access to profile CIMSERV in class WBEM. Grant the CIM server user ID CONTROL access to profile CIMSERV in class WBEM.

Error: CIM runtime environment user ID requires either READ access to BPX.SERVER or it must be user ID 0.

When or where seen: The CIM server error log after CIM server fails to start

Solution: Either permit the user ID READ access to BPX.SERVER if BPX.SERVER is set up, or run the command under a privileged user ID (UID 0).

Error: CFZ17201W: ACCESS IS NOT ENABLED FOR REMOTE USERS WITH SUPERUSER AUTHORITY.

When or where seen: On the client side.

Solution: The remote client uses a local user with UID=0. However, the CIM server is configured to reject remote access if the local user is a super-user (parameter enableRemotePrivilegedUserAccess=false). If you want to enable the local user with remote privileged access, then switch the parameter to true. Otherwise, change the local user to a non-super-user by setting the UID ≠ 0.

Error:
CFZ10033E: The CIM server is not started:
CFZ00409E: Bind failed:
CFZ07801E: CIM HTTP or HTTPS connection failed to create the socket.

When or where seen: CIM server startup console messages

Solution: The CIM server cannot start because it fails to listen on one of the ports 5988 (for http) or 5989 (for https). Either the CIM server is already running, another server is listening on one of these ports, or the ports have been blocked by the TCP/IP configuration.

The httpPort and httpsPort CIM server configuration properties define the HTTP port and HTTPS port numbers (see Chapter 9, “CIM server configuration,” on page 53).

- Check the PORT and PORTRANGE statements in the PROFILE.TCPIP configuration file to ensure that the specified ports are accessible to the CIM server.
- Check the security product configuration to ensure that the CIM server is able to access the specified ports. For example, OEM security product ACF2 may require "Stack & Port security authorization" for the CIM server.
Use the TCP/IP NETSTAT ALLCONN PORT command to check for servers using the specified ports, for example issue TSO NETSTAT ALLCONN (PORT 5988)

See “Configuring the ports for the CIM server” on page 45 for more information.

Error: HTTP Error (401 Unauthorized)

**When or where seen:** Client application

**Solution:** The user authentication failed. The client application either did not provide user ID and password on a request at all, or the supplied user ID and password are not valid for the z/OS system on which the CIM server is running.

Permit the user ID to execute a client request with at least READ access to Security profile CIMSERV in class WBEM. Check the server log for a detailed error report.

Error: ICH14080I

Warning: RACF detected a possible error in the dynamic class descriptor table, entry WBEM, error code 01. The class is available for further processing. The class name does not contain a national character nor a number. To assure IBM does not create an IBM-defined class in the future by this same name, you should choose a class name which contains at least one national character or a number.

**When or where seen:** RACF setup of dynamic class WBEM

**Solution:** Ignore the warning.

Error: ICH408I USER(CFZSRV) GROUP(CFZSRVGP) NAME(####################) CL(PROCESS ) INSUFFICIENT AUTHORITY TO NEWJOBNAME

**When or where seen:** Message on the console.

**Solution:** Grant the CIM server user ID READ access to profile BPX.JOBNAME in class FACILITY to be allowed to set the job name of the out-of-process agent to CFZOOPA (see “Running providers in separate address spaces” on page 62).

Error: IEF450I CFZCIM - ABEND=S1C7 U0000 REASON=FFFF0006

**When or where seen:** Message on the console.

**Solution:** Look for CSV042I and ICH422I program control messages. CSV042I message points out the module to be marked as program controlled. If no CSV042I and ICH422I messages occur contact IBM Service.

**Example:**

CSV042I REQUESTED MODULE BLSUXTID NOT ACCESSSED.
    THE MODULE IS NOT PROGRAM CONTROLLED
ICH422I THE ENVIRONMENT CANNOT BECOME UNCONTROLLED.
CSV028I ABEND306-42 JOBNAME=CFZCIM STEPNAME=
BPXP014I ENVIRONMENT MUST REMAIN CONTROLLED FOR
DAEMON (BPX.DAEMON) PROCESSING.
IEF450I CFZCIM - ABEND=S1C7 U0000 REASON=FFFF0006 TIME=14.16.12
Error: JGP00001W: Number of Instances Exceeded Threshold
This error message might be issued at enumeration of IBMzOS_Job instances, when the number of instances to be enumerated is greater than a configured limit. This limit has been defined to prevent the CIM server from resource exhaust.

It is recommended to change your enumeration to a subset of IBMzOS_Job.

To query the current limit of the IBMzOS_Job provider, receive the IBMzOS_JobsManagementSettings instance of the CIM server. The property MaxInstances contains the currently defined limit.

To change the limits, set the property MaxInstances to a new value by modifying the IBMzOS_JobsManagementSettings instance.

```
cimcli mi
IBMzOS_JobsManagementSettings.InstanceID="IBMzOS:JobsManagementSettings"
MaxInstances=<new_value>
```

Client Side Error: CIM_ERR_ACCESS_DENIED
Access to a CIM resource was not available to the client: "EDC5139I Operation not permitted."

When or where seen: Client application / Details in the CIM server trace log

Solution: Permit the CIM server runtime environment user ID as surrogate for the requesting client user ID to use the command: PERMIT BPX.SRV.<client uid> CL(SURROGAT) ID(<CIMServer UID>) ACCESS(READ)

Client side error: HTTP Error (413 Request Entity Too Large)
There wasn't enough memory available to the client to successfully read the entire response from the server into memory.

When or where seen: Client application, like for instance cimivp.

Solution: Allow the client to use more memory. If the application runs within a JOB, increase the REGION size. If the client runs from a UNIX System Services shell, increase the ASSIZEMAX value in the OMVS segment of the user running the shell.
Appendix B. Step-by-step explanation of the CFZSEC job

This appendix provides an explanation for each single step of the CIM security setup job CFZSEC.

Please note that the CFZSEC job provides a quick security setup for CIM. Because this job provides a solution for each configuration, necessarily the job steps which do not apply to your system will fail. But this does not affect the job’s functionality.

The job creates security profiles, users and groups required to run CIM and grants them the necessary permissions to system resources.

Step BASICSUP

```plaintext
/* Step BASICSUP dose set-up basic security settings.
  /* - Program control for runtime libraries.
  BASICSUP EXEC PGM=IKJEFT01,DYNAMNBR=99
  SYSPRINT DD SYSOUT=*  
  SYSTSPRT DD SYSOUT=*  
  SYSTSIN DD *  
  RALT PROGRAM * ADDMEM('SYS1.SCEERUN'/'******'/NOPADCHK) +  
    UACC(READ)  
  RALT PROGRAM * ADDMEM('SYS1.SCEERUN2'/'******'/NOPADCHK) +  
    UACC(READ)  
  SETROPTS WHEN(PROGRAM) REFRESH
  */
```

This sets up the basic security for the CIM server. To enable the CIM server to run in a program controlled environment, the Language Environment runtime libraries SCEERUN and SCEERUN2 must be program controlled.
Step CRUSR

```bash
/*
/* Step CRUSR creates default groups and users required for CIM
/* CFZSRVGP - CIM server ID's default group
/* CFZADMGP - CIM admin ID's default group
/* CFZUSRGP - CIM end-users ID's default group
/*
/* CFZSRV - CIM server User ID used by Started Task
/*
//CRUSR EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=* 
//SYSTSPRT DD SYSOUT=* 
//SYSTSIN DD *

ADDGROU CFZSRVGP OMVS(GID(9501))
ADDGROU CFZADMGP OMVS(GID(9502))
ADDGROU CFZUSRGP OMVS(GID(9503))

ADDUSER CFZSRV DFLTGRP(CFZSRVGP) OMVS(UID(0) PROGRAM('/bin/sh') + HOME('/u/cfzsrv')) NOPASSWORD NOOIDCARD
ADDSD 'CFZSRV.**' UACC(NONE)
PERMIT 'CFZSRV.**' CLASS(DATASET) ID(CFZSRV) ACCESS(ALTER)
SETPROPTS GENERIC(DATASET) REFRESH
ALTUSER CFZSRV DFLTGRP(CFZSRVGP) OMVS(UID(0) PROGRAM('/bin/sh') + HOME('/u/cfzsrv')) NOPASSWORD NOOIDCARD NOPHRASE */
```

This step creates or updates the user CFZSRV for running the CIM server as a started task. By default the UID for the CIM server user is set to 0 to run the CIM server with superuser privileges. While this may be sufficient for a simple setup, if you have defined the BPX.SERVER profile in the class FACILITY, and class FACILITY is activated, it is recommended to change the UID for CFZSERV to a non null value. The default in this case is 9500.

A default data set profile is created to ensure that the CIM server user ID can access its home profile and other relevant settings.

In addition this step creates distinct groups for the CIM server user (CFZSRVGP), CIM server administrators (CFZADMGP) and end users (CFZUSRGP). To grant a user access to CIM, simply connect the user to the according group, for example with the command

```
CONNECT (username) GROUP(CFZUSRGP) AUTHORITY(USE)
```

The CFZUSRGP grants a user access to all resources that are managed through CIM. Depending on how granular you want to control users' access to CIM, you may want to create additional groups that allow access only to a subset of resources managed through CIM.
**Step CRWBEM**

/* Step CRWBEM creates class WBEM and profile CIMSERV
** CRWBEM EXEC PGM=IKJEFT01,DYNAMNBR=99
** SYSPRINT DD SYSOUT=* 
** SYSTSPRT DD SYSOUT=* 
** SYSTSIN DD *

SETROPTS CLASSACT(CDT) RACLIST(CDT)

RDEFINE + 
CDT WBEM + 
UACC(NONE) + 
CDTINFO( CASE(UPPER) + 
MAXLENGTH(246) + 
FIRST(ALPHA) + 
OTHER(ALPHA,NUMERIC) + 
MAXLENX(246) + 
KEYQUALIFIERS(0) + 
PROFILESALLOWED(YES) + 
POSIT(200) + 
DEFAULTRC(8) + 
DEFAULTUACC(NONE) + 
RACLIST(REQUIRED))

SETROPTS RACLIST(CDT) REFRESH

SETROPTS CLASSACT(WBEM) RACLIST(WBEM)

RDEFINE WBEM CIMSERV UACC(NONE)

SETROPTS CLASSACT(WBEM) RACLIST(WBEM)

/*

This step creates the RACF class and profile required to control access to the CIM server.

If the POSIT value 200 for RACF is already in use on your system, change the value defined in this step.
Step PEUSR

This step grants CIM users the necessary permissions to run, to control and to access the CIM server.

In detail it grants the following permissions:

For the CIM server user:

- **CONTROL** access to profile CIMSERV in class WBEM
  This allows the user to start the CIM server.
- **READ** access to profile BPX.SRV.** in class SURROGAT
  This allows the CIM server to switch a TCB into a requestor’s user for running client requests under the authority of the client’s user.
- **UPDATE** access to profile BPX.SERVER in class FACILITY
  This authorizes the CIM server to validate user credentials and to verify user access to RACF profiles.
- **READ** access to profile BPX.SMF in class FACILITY
  This allows the CIM server to write SMF records when it is configured to do so. (See “Audit logging with SMF record 86” on page 69 for details on SMF support in CIM.)
- **READ** access to profile BPX.CONSOLE in class FACILITY
  This allows the CIM server to issue messages on the z/OS console when the BPX.CONSOLE profile is defined.
For the CIM administrator group:
- **CONTROL** access to profile CIMSERV in class WBEM
  This allows a user to perform administrative functions.

For the CIM users group:
- **UPDATE** access to profile CIMSERV in class WBEM
  This allows a user to access CIM as a regular user.

### Step PEAPPL

```plaintext
//** Step PEAPPL Permit CIM groups and users to net application CFZAPPL
/*
   This has no effect if class APPL is not active.
/PEAPPL EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=* 
//SYSTSPRT DD SYSOUT=* 
//SYSTSIN DD * 
RDEFINE APPL CFZAPPL UACC(NONE) 
PERMIT CFZAPPL CL(APPL) ACCESS(READ) ID(CFZSRV) 
PERMIT CFZAPPL CL(APPL) ACCESS(READ) ID(CFZADMGP) 
PERMIT CFZAPPL CL(APPL) ACCESS(READ) ID(CFZUSRGP) 
SETROPTS RACLIST(APPL) REFRESH
/*
```

When class APPL is active, the CFZAPPL profile protects access to the CIM server application. Any user who wants to access the CIM server requires at least **READ** access to the CFZAPPL profile in the APPL class. This job step grants this access for the CIM server user, the CIM administrator group, and the CIM users group.

### Step SETARM

```plaintext
//** Step SETARM establishes security setup required for ARM
/*
   A sample ARM policy (CFZARM) resides in the installed
   SYS1.SAMPLIB
/SETARM EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=* 
//SYSTSPRT DD SYSOUT=* 
//SYSTSIN DD * 
SETROPTS CLASSACT(FACILITY) RACLST(FACILITY) GENERIC(FACILITY) 
RDEFINE FACILITY IXCARM.DEFAULT.CFZ_SRV_* UACC(NONE) 
PERMIT IXCARM.DEFAULT.CFZ_SRV_* CLASS(FACILITY) +
  ID(CFZSRV) ACCESS(UPDATE)
/*
```

This step enables the CIM server for registering with the z/OS Automatic Restart Manager (ARM).

To completely enable the CIM server for ARM, additional customization is required as described in "Automatically restarting the CIM server" on page 70.
Step ENSTC

Step ENSTC

// * Step ENSTC establishes CFZSRV as the Started Task User for CIM  
// ENSTC EXEC PGM=IKJEFT01,DYNAMNBR=99  
// SYSPRINT DD SYSOUT=*  
// SYSTSIN DD SYSOUT=*  
// SYSTSIN DD SYSOUT=*  
SETROPTS CLASSACT(STARTED) RACLIST(STARTED) GENERIC(STARTED)  
RDEFINE STARTED CFZCIM.* STDATA(USER(CFZSRV) GROUP(CFZSRVGP))  
SETROPTS RACLIST(STARTED) REFRESH  
/

This step connects the CIM server started task procedure CFZCIM with the CIM server user CFZSRV.

For further details on configuring the CIM server started task procedure, see "Customizing the started task procedure CFZCIM" on page 49.
Step PECEA

```plaintext
/* Step PECEA permits CIM Cluster and JES jobs provider to access CEA*/

//PECEA EXEC PGM=IKJEFT01, DYNAMNBR=99
//SYSPRINT DD SYSOUT=*  
//SYSTSPRT DD SYSOUT=*  
//SYSTSIN DD *  

ADDS D 'CEA.*' UACC(NONE)
PERMIT 'CEA.*' CLASS(DATASET) ID(CFZUSRGP) ACCESS(ALTER)
PERMIT 'CEA.*' CLASS(DATASET) ID(CFZADMGP) ACCESS(ALTER)
SETROPTS GENERIC(DATASET) REFRESH

SETROPTS CLASSACT(SERVAUTH) RACLST(SERVAUTH) GENERIC(SERVAUTH)
RDEFINE SERVAUTH CEA.* UACC(NONE)

PERMIT CEA.* CLASS(SERVAUTH) ID(CFZADMGP) ACCESS(UPDATE)
PERMIT CEA.* CLASS(SERVAUTH) ID(CFZUSRGP) ACCESS(UPDATE)

PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZADMGP) ACCESS(UPDATE)
PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT CEA.CONNECTCLASS(SERVAUTH) ID(CFZUSRGP) ACCESS(UPDATE) + ACCESS(UPDATE)
PERMIT CEA.CEADOCMD CLASS(SERVAUTH) ID(CFZADMGP) ACCESS(UPDATE)
PERMIT CEA.CEADOCMD CLASS(SERVAUTH) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT CEA.CEADOCMDCLASS(SERVAUTH) ID(CFZUSRGP) ACCESS(UPDATE)

SETROPTS RACLST(SERVAUTH) REFRESH
/
```

This step permits CIM users and administrators to access CEA through the CIM providers for the OS management Jobs and Cluster classes described in "OS management Job classes" on page 160 and "OS management Cluster classes" on page 191.

**Note:** This step defines the generic resource profile CEA.* and permits the CIM default groups CFZADMGP and CFZUSRGP access to it.

For the case that you have already defined the specific resource profiles (CEA.CONNECT, etc), this step also permits the CIM default groups to these specific resource profiles.

Depending on what you have actually defined, you can customize this job step to match your environment by removing obsolete commands.

For granting users a more fine grained access to CIM you may consider to define an additional group here that grants access just for OS management Jobs and Cluster classes.
For further details on the required setup for using the OS management Jobs and Cluster classes see "Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers" on page 38.

---

**Step ENCLCDS**

```plaintext
//* Step ENCLCDS Setup for Cluster/Couple Dataset Providers
//*
//ENCLCDS EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=*  
//SYSTSRT DD SYSOUT=*  
//SYSTSN IN DD *

SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) GENERIC(FACILITY)

RDEFINE FACILITY MRCLASS.CLUSTER UACC(NONE) 
PERMIT MRCLASS.CLUSTER CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT MRCLASS.CLUSTER CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)

RDEFINE FACILITY MVSADMIN.* UACC(NONE)
PERMIT MVSADMIN.* CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT MVSADMIN.* CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)
PERMIT MVSADMIN.XCF.* CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT MVSADMIN.XCF.CFRM CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
PERMIT MVSADMIN.XCF.CFRM CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY) REFRESH
/*
```

This step permits CIM users and administrators to use the CIM providers for the OS management Cluster classes described in "OS management Cluster classes" on page 191.

For granting users a more fine-grained access to CIM, you may consider to define an additional group here that grants access just for OS management Cluster classes.

For further details on the required setup for using the OS management Cluster classes see "Setting up the CIM server for Cluster, CoupleDataset, and JES2-JES3Jobs providers" on page 38.
Step ENSMIS

```bash
//ENSMIS EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=* 
//SYSTSRT DD SYSOUT=* 
//SYSTSIN DD * 
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) GENERIC(FACILITY) 
RDEFINE FACILITY IOSCDR UACC(NONE) 
PERMIT IOSCDR CL(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE) 
PERMIT IOSCDR CL(FACILITY) ID(CFZADMGP) ACCESS(UPDATE) 
PERMIT IOSCDR CL(FACILITY) ID(CFZSRV) ACCESS(UPDATE) 
SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH) GENERIC(SERVAUTH) 
RDEFINE SERVAUTH CEA.* UACC(NONE) 
PERMIT CEA.* CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ) 
PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZSRV) ACCESS(READ) 
PERMIT CEA.SUBSCRIBE.ENF_0009* CLASS(SERVAUTH) ID(CFZSRV) + 
ACCESS(READ) 
PERMIT CEA.SUBSCRIBE.ENF_0027* CLASS(SERVAUTH) ID(CFZSRV) + 
ACCESS(READ) 
PERMIT CEA.SUBSCRIBE.ENF_0033* CLASS(SERVAUTH) ID(CFZSRV) + 
ACCESS(READ) 
SETROPTS RACLIST(FACILITY) REFRESH 
SETROPTS RACLIST(SERVAUTH) REFRESH 
/*
```

This step permits the CIM server user ID to access CEA through the CIM live cycle indication providers for the Storage management classes as described in "Storage management classes" on page 233.

In particular a CIM user requires this permission to access the CIM providers for the following storage management classes:

- IBMzOS_SBProtocolEndpoint
- Association IBMzOS_SBInitiatorTargetLogicalUnitPath

This step defines the generic resource profile CEA.* and permits the default CIM server user ID CFZSRV access to it. For the case that you have already defined the specific resource profiles such as CEA.CONNECT, this step also permits the default CIM server user ID to these specific resource profiles. Depending on what you have actually defined, you can customize this job step to match your environment by removing obsolete commands.

For granting users a more fine-grained access to CIM, you may consider to define an additional group that grants access just for Storage management classes.
**Step ENTCPIP**

`/* Step ENTCPIP enables the Network CIM providers
 //ENTCPIP EXEC PGM=IKJEFT01,DYNAMNBR=99
 //SYSPRINT DD SYSOUT=* 
 //SYSTSRT DD SYSOUT=* 
 //SYSTSIN DD *

 SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH) GENERIC(SERVAUTH) 
 RDEFINE SERVAUTH EZB.CIMPROV.* UACC(NONE)

 PERMIT EZB.CIMPROV.* CL(SERVAUTH) ID(CFZADMGP) ACCESS(READ)
 PERMIT EZB.CIMPROV.* CL(SERVAUTH) ID(CFZUSRGP) ACCESS(READ)

 SETROPTS RACLIST(SERVAUTH) REFRESH
/*`

This step permits CIM users and administrators to use the CIM providers for the OS management Network classes described in ["OS management Network classes" on page 154](#).

For granting users a more fine-grained access to CIM, you may consider to define an additional group here that grants access just for the OS management Network classes.

**Step ENWLM**

`/* Step ENWLM Setup for WLM Providers
 /*
 //ENWLM EXEC PGM=IKJEFT01,DYNAMNBR=99
 //SYSPRINT DD SYSOUT=* 
 //SYSTSRT DD SYSOUT=* 
 //SYSTSIN DD *

 SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY) GENERIC(FACILITY) 
 RDEFINE FACILITY MVSADMIN.* UACC(NONE)
 PERMIT MVSADMIN.* CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
 PERMIT MVSADMIN.* CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)
 PERMIT MVSADMIN.WLM.* CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
 PERMIT MVSADMIN.WLM.* CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)
 PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(CFZUSRGP) ACCESS(UPDATE)
 PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(CFZADMGP) ACCESS(UPDATE)
 RDEFINE PROGRAM BLSUXTID
 RALT PROGRAM BLSUXTID ADDMEM('SYS1.MIGLIB'/******'/NOPADCHK) + 
 UACC(READ)

 SETROPTS RACLIST(FACILITY) REFRESH
 SETROPTS WHEN(PROGRAM) REFRESH
/*`

This step permits CIM users and administrators to use the CIM providers for the WLM classes described in [Chapter 15, “WLM classes,” on page 261](#).

336  z/OS V1R13.0 Common Information Model User's Guide
For granting users a more fine-grained access to CIM, you may consider to define an additional group here that grants access just for the WLM classes.

**Step ENRMF**

```plaintext
//* Step ENRMF creates profiles necessary to allow passtickets being
generated for authentication with the DDS
//ENRMF EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPRINT DD SYSOUT=*  
//SYSTSPRT DD SYSOUT=* 
//SYSTSIN DD *
SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA) GENERIC(PTKTDATA) 
RDEFINE PTKTDATA GPMSERVE SSIGNON(KEVMASKED(#rkeymask))  
RDEFINE PTKTDATA IRRPTAUTH.GPMSERVE.* UACC(NONE) 
PERMIT IRRPTAUTH.GPMSERVE.* CL(PTKTDATA) ID(CFZSRV) ACCESS(UPDATE)  
SETROPTS RACLIST(PTKTDATA) REFRESH
/*
```

If you are not using the z/OS Resource Measurement Facility (RMF) optional element, remove this step from the job. Otherwise this step permits the CIM server access to the RMF Distributed Data Server using passtickets. For this, replace `#rkeymask` by a 16-digit (0-9,A-F) keymask value to setup connectivity between CIM and RMF via passtickets.

**Note:**
The keymask value is a secret passkey. In a secure environment it is recommended to execute step ENRMF separately to avoid storing the passkey in the job log in readable format.

The CIM classes implemented by RMF are described in the [z/OS RMF Programmer’s Guide](#) and [z/OS RMF User’s Guide](#).
## Appendix C. CEA reason codes

The following list of reason codes may be returned by the methods in the Jobs providers. The first four digits (X'xxxx') may be any value.

<table>
<thead>
<tr>
<th>Reason code (hex)</th>
<th>Description</th>
<th>User action</th>
<th>IBM Service Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'xxxx0100'</td>
<td>Common Event Adapter (CEA) communication unavailable.</td>
<td>Ensure CEA is active; Call IBM Service.</td>
<td>CEAUNAVAIL</td>
</tr>
<tr>
<td>X'xxxx0117'</td>
<td>Instrumentation is unable to accommodate additional CIM indication providers.</td>
<td>Remove unused/unnecessary indication provider connections from the instrumentation. Call IBM Service if this is a consistent problem.</td>
<td>CEAMAXCLIENTSCONNECTED</td>
</tr>
<tr>
<td>X'xxxx011F'</td>
<td>z/OS System Operator forced the unsubscribe of the event.</td>
<td>Resubscribe to the event.</td>
<td>CEASYSOPFORCEUNSUBSCRIBE</td>
</tr>
<tr>
<td>X'xxxx0121'</td>
<td>Common Event Adapter (CEA) is no longer able to communicate with CIM indication providers.</td>
<td>Adjust CEA by transitioning the component from minimum mode to full mode. Operator must use F CEA,MODE=FULL</td>
<td>CEAFORCEMINMODE</td>
</tr>
<tr>
<td>X'xxxx0126'</td>
<td>Instrumentation is unable to accept any more subscriptions to indication events.</td>
<td>Remove unused/unnecessary indication event subscriptions</td>
<td>CEAMAXPGMSUBSCRIBED</td>
</tr>
<tr>
<td>X'xxxx020A'</td>
<td>Common Event Adapter (CEA) was unable to find exit handler.</td>
<td>Ensure that the exit handler is installed properly by the SMP/E installation step. The handlers are usually installed in the LPA.</td>
<td>CEAHANDLERNOTFOUND</td>
</tr>
<tr>
<td>X'xxxx0300'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAREQUESTNOTRECOGNIZED</td>
</tr>
<tr>
<td>X'xxxx0301'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAREQUESTNOTIMPLEMENTED</td>
</tr>
<tr>
<td>X'xxxx0302'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYSTRUCTBADPTR</td>
</tr>
<tr>
<td>X'xxxx0303'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYSTRUCTBADEYE</td>
</tr>
<tr>
<td>X'xxxx0304'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYSTRUCTBADVERSION</td>
</tr>
<tr>
<td>X'xxxx0305'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYBADRESOURCE</td>
</tr>
<tr>
<td>X'xxxx0306'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYNOMATCH</td>
</tr>
<tr>
<td>X'xxxx0307'</td>
<td>Internal CIM error</td>
<td>Call IBM Service.</td>
<td>CEAPROPERTYSTRUCTEMPTY</td>
</tr>
<tr>
<td>X'xxxx0308'</td>
<td>Internal CEA error</td>
<td>Call IBM Service.</td>
<td>CEAENVBAD</td>
</tr>
</tbody>
</table>
### Table 8. Jobs providers’ reason codes (continued)

<table>
<thead>
<tr>
<th>Reason code (hex)</th>
<th>Description</th>
<th>User action</th>
<th>IBM Service Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'xxxx0309'</td>
<td>Internal CIM error</td>
<td>Call IBM Service</td>
<td>CEAFILTERSTRUCTBADEYE</td>
</tr>
<tr>
<td>X'xxxx030A'</td>
<td>Internal CIM error</td>
<td>Call IBM Service</td>
<td>CEAFILTERSTRUCTBADVERSION</td>
</tr>
<tr>
<td>X'xxxx030B'</td>
<td>Internal CIM error</td>
<td>Call IBM Service</td>
<td>CEAFILTERBADRESOURCE</td>
</tr>
<tr>
<td>X'xxxx030C'</td>
<td>Internal CIM error</td>
<td>Call IBM Service</td>
<td>CEAFILTERNOMATCH</td>
</tr>
<tr>
<td>X'xxxx030D'</td>
<td>Internal CIM error</td>
<td>Call IBM Service</td>
<td>CEABADPARMPTR</td>
</tr>
<tr>
<td>X'xxxx030E'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADSSISUBSYSTEM</td>
</tr>
<tr>
<td>X'xxxx030F'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADSSICALL</td>
</tr>
<tr>
<td>X'xxxx0310'</td>
<td>Internal CEA error</td>
<td>Ensure JES2/JES3 is active. Ensure that ExtendedSubsystem is available. Call IBM Service</td>
<td>CEANOSSI</td>
</tr>
<tr>
<td>X'xxxx0311'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADSSIENV</td>
</tr>
<tr>
<td>X'xxxx0312'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEAVBADSSI</td>
</tr>
<tr>
<td>X'xxxx0314'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Check storage indicators (monitors). Call IBM Service if external symptom not resolved</td>
<td>CEUNABLETOALLOCATE</td>
</tr>
<tr>
<td>X'xxxx0315'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEANOTJOBSTERSEELEMENT</td>
</tr>
<tr>
<td>X'xxxx0316'</td>
<td>Internal CEA error</td>
<td>SSI Abend. Look for SDUMP. Call IBM Service</td>
<td>CEAJOBCHAINBROKEN</td>
</tr>
<tr>
<td>X'xxxx0317'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEABADDATENV</td>
</tr>
<tr>
<td>X'xxxx0318'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEASYSOUTCHAINBROKEN</td>
</tr>
<tr>
<td>X'xxxx0319'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEANOTSYSOUTHDRELEMENT</td>
</tr>
<tr>
<td>X'xxxx031A'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADFREEPTR</td>
</tr>
<tr>
<td>X'xxxx031B'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADFREEBLK</td>
</tr>
<tr>
<td>X'xxxx031C'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADFREEENV</td>
</tr>
<tr>
<td>X'xxxx031D'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEUNABLETOFREE</td>
</tr>
<tr>
<td>X'xxxx031E'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADIEFQRY</td>
</tr>
<tr>
<td>X'xxxx031F'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEASSCHAINBROKEN</td>
</tr>
<tr>
<td>X'xxxx0320'</td>
<td>Internal CEA error</td>
<td>Look for SDUMP. Call IBM Service</td>
<td>CEAVBADJSQY</td>
</tr>
<tr>
<td>X'xxxx0321'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADFILTEROPER</td>
</tr>
<tr>
<td>X'xxxx0322'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADSS4SUBSYSTEM</td>
</tr>
<tr>
<td>X'xxxx0323'</td>
<td>Internal CEA error</td>
<td>Call IBM Service</td>
<td>CEABADSS4CALL</td>
</tr>
<tr>
<td>X'xxxx0324'</td>
<td>Internal CEA error</td>
<td>SSI not activated. Call IBM Service</td>
<td>CEANOS54</td>
</tr>
<tr>
<td>Reason code (hex)</td>
<td>Description</td>
<td>User action</td>
<td>IBM Service Information</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------</td>
<td>------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>X'xxxx0325'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEABADS$54ENV</td>
</tr>
<tr>
<td>X'xxxx0327'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEABADS$54STOR</td>
</tr>
<tr>
<td>X'xxxx0328'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEATIMEOUTMAXIMUMEXCEEDED</td>
</tr>
<tr>
<td>X'xxxx0329'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEANEEDSYSOUTFILTER</td>
</tr>
<tr>
<td>X'xxxx032A'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEABUFFERTOOLARGE</td>
</tr>
<tr>
<td>X'xxxx032B'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEACCMDS$DIAGRCSET</td>
</tr>
<tr>
<td>X'xxxx032C'</td>
<td>Internal CEA error.</td>
<td>Ensure SYSREXX is active/operational using the F AXR,DISPLAY command. Call IBM Service if AXREXX is active.</td>
<td>CEACCMDS$AXREXXRCSET</td>
</tr>
<tr>
<td>X'xxxx032D'</td>
<td>Client not authorized for instrumentation</td>
<td>Ensure user has access to instrumentation facilities.</td>
<td>CEANOINSTRAUTH</td>
</tr>
<tr>
<td>X'xxxx032E'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEATOOMUCHDATA</td>
</tr>
<tr>
<td>X'xxxx032F'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEAFILTERNOTSUPPORTED</td>
</tr>
<tr>
<td>X'xxxx0330'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEAPRIMARYTYPEPISIMATCH</td>
</tr>
<tr>
<td>X'xxxx0331'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEABADSUBSYSTEM</td>
</tr>
<tr>
<td>X'xxxx0332'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEAUNABLETOALLOCATE2</td>
</tr>
<tr>
<td>X'xxxx0333'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEABADBUFFER</td>
</tr>
<tr>
<td>X'xxxx0334'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEATIMEOUTLESSTHANMINIMUM</td>
</tr>
<tr>
<td>X'xxxx0335'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEACMDS$SYNTAXERROR</td>
</tr>
<tr>
<td>X'xxxx0336'</td>
<td>The CIM provider request was cancelled in-process.</td>
<td>Retry the command request. If it does not work, call IBM Service.</td>
<td>CEACMD$SHALTERROR</td>
</tr>
<tr>
<td>X'xxxx0337'</td>
<td>Internal CIM error.</td>
<td>Call IBM Service.</td>
<td>CEACMDS$UNINITERROR</td>
</tr>
<tr>
<td>X'xxxx0338'</td>
<td>Internal CEA error.</td>
<td>Call IBM Service.</td>
<td>CEAFILTERBADCOMBO</td>
</tr>
<tr>
<td>X'xxxx0339'</td>
<td>Underlying command did not complete in the time specified.</td>
<td>Increase timeout value in the CIM method request and retry request.</td>
<td>CEACMD$TIMEOUTDOUT</td>
</tr>
</tbody>
</table>
Related links

CIM Event Model White Paper

CIM Query Language Specification
http://www.dmtf.org/sites/default/files/standards/documents/DSP0202_1.0.0.pdf

Common Information Model (CIM) Standards
http://www.dmtf.org/standards/cim

DMTF website
http://www.dmtf.org

DMTF DSP0226: Web Services for Management (WS-Management) Specification
http://www.dmtf.org/sites/default/files/standards/documents/DSP0226_1.0.0.pdf

DMTF DSP0227: WS-Management CIM Binding Specification
http://www.dmtf.org/sites/default/files/standards/documents/DSP0227_1.0.0.pdf

DMTF DSP0230: WS-CIM Mapping Specification
http://www.dmtf.org/sites/default/files/standards/documents/DSP0230_1.0.1.pdf

eServer Common Information Model

LookAt website for online message explanations
http://www.ibm.com/systems/z/os/zos/bkserv/lookat/

OpenPegasus website
http://www.openpegasus.org

SNIA website
http://www.snia.org/

SourceForge.net
http://sourceforge.net/

Specification for CIM Operations over HTTP
http://www.dmtf.org/standards/published_documents/DSP0200_1.3.0.pdf

Storage Management Initiative Specification (SMI-S)
http://www.snia.org/tech_activities/standards/curr_standards/smi/

WBEM standards
http://www.dmtf.org/standards/wbem

Web Services for Management

WS-CIM Mapping specification
http://dmtf.org/sites/default/files/standards/documents/DSP0230_1.0.2.pdf
WS-Management CIM Binding Specification
http://dmtf.org/sites/default/files/standards/documents/DSP0227_1.1.0.pdf

z/OS information updates on the web
http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ZIDOCMST/CCONTENTS
Legend for graphics showing class structures

The graphics in this book showing class structures illustrate the CIM object modeling using the UML syntax:

Table 9. UML syntax

<table>
<thead>
<tr>
<th>Construct</th>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>association</td>
<td>A relationship between two or more classifiers that involves connections among their instances.</td>
<td></td>
</tr>
<tr>
<td>aggregation</td>
<td>A special form of association that specifies a whole-part relationship between the aggregate (whole) and the component part.</td>
<td></td>
</tr>
<tr>
<td>inheritance</td>
<td>A relationship among classes where one class shares the structure and/or behavior defined for one or more other classes. Inheritance is the mechanism that makes generalization, subclasses, and superclasses possible.</td>
<td></td>
</tr>
<tr>
<td>class</td>
<td>Denotes the representation of a CIM class in UML notation with title, properties, and methods.</td>
<td></td>
</tr>
</tbody>
</table>
How to read syntax diagrams

The following rules apply to the syntax diagrams used in this book:

Arrow symbols
Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

The arrow symbols indicate the beginning and the end of a statement.

Conventions
- Variables are shown in italics, for example, column-name. They represent user-defined parameters or suboptions.
- When entering commands, separate parameters and keywords by at least one blank if there is no intervening punctuation.
- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs) and numbers exactly as given.

Required items
Required items are displayed on the horizontal line (the main path).

Optional items
Optional items normally are shown below the main path.

Multiple required or optional items
If you can choose from two or more items, they are displayed vertically in a stack.

If you must choose one of the items, one item of the stack is on the main path.

If choosing one of the items is optional, the entire stack is below the main path.

Repeatable items
An arrow returning to the left, above the main line, indicates that an item can be repeated.
A repeat arrow above a stack indicates that you can repeat the items in the stack.

Keywords

- Keywords are shown in uppercase (for example, FROM). They specify field names in the field table and must be substituted by the field ID when coded.
- Variables are shown in all lowercase letters (for example, column name). They represent user-supplied names or values. Variables enclosed in quotes must be coded as character strings.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.
Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you may view the information through the z/OS Internet Library website or the z/OS Information Center. If you continue to experience problems, send an email to mhvrdfs@us.ibm.com or write to:

IBM Corporation  
Attention: MHVRCFS Reader Comments  
Department H6MA, Building 707  
2455 South Road  
Poughkeepsie, NY 12601-5400  
U.S.A.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

• Use assistive technologies such as screen readers and screen magnifier software
• Operate specific or equivalent features using only the keyboard
• Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to z/OS TSO/E Primer, z/OS TSO/E User's Guide, and z/OS ISPF User's Guide Vol I for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at:

http://www.ibm.com/systems/z/os/zos/bkserv/
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information about 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 give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

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 Corporation
Mail Station P300
2455 South Road
Poughkeepsie New York 12601-5400
U.S.A.

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.

For license inquiries regarding double-byte (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.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL
Programming Interface Information

This book is intended to help the customer to use the Common Information Model to write system management applications for z/OS systems.

The book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of CIM.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at [Copyright and trademark information] at [http://www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.
Terms and conditions for downloading and printing publications

Permissions for the use of the publications you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

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 works 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.

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 EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

All material copyrighted by IBM Corporation.

By downloading or printing a publication from this site, you have indicated your agreement with these terms and conditions.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.
Index

Special characters

_BPX_JOBNAME 49, 69
_BPX_SHAREAS 28, 62

A
accessibility 349
ADDUSER command 25
administrator access 28
ALTUSER command 25
APPL 35
Application Transparent Transport Layer
Security
See AT-TLS
ARM 39, 70
element name 71
policy 71
security 36
ASCII 65, 276
ASCII-EBCDIC conversion 323
association
classes 102, 103, 125
provider function 275
AT-TLS 11, 12, 29
audit logging 69
authentication 11, 12, 28
authorization 11, 13, 25, 42
check 43
model 25
model, provider-based 43
of users 20
automatic restart 70
Automatic Restart Manager
See ARM

B
backup
CIM server configuration 70
CIM server repository 73
Base classes 121, 126
BaseBoard classes 121, 138
BPX_DAEMON 27
BPX_JOBNAME 62
BPX_POE 35
BPX_SERVER 20, 25
BPX_SMF 69
BPX_SRV 25, 29

C
CDT 24
CDTINFO 24
CEA 38, 39, 41, 339
certificate 11, 29, 30, 31
Certificate Authority 30
CFRM 39
CFZAPPL 35
CFZARMP 71
CFZCIM 19, 21, 49, 61, 69, 70
customization 49
start 61
stop 61
CFZIVP 19, 22
CFZRCUST 19, 20
customization 46
CFZRCUST parameter
-noDS 48
-noSpaceCheck 47, 48
CFZSEC 19, 327
step BASICSUP 327
step CRUSR 328
step CRWBEM 329
step ENCLCS 334
step ENRMF 337
step ENSMIS 335
step ENSTC 332
step ENTCPPI 336
step ENWLIM 336
step PEAPPL 331
step PECEA 333
step PEUSR 330
step SETARM 331
CFZSRV 20, 25
CFZSRVG 20, 25
CIM
introduction 3
CIM client 4, 5
authentication 29, 31
request 75, 84
CIM client for Java 4
CIM Event Model 7
CIM indication
See indication
CIM instrumentation 117
CIM listener 9
CIM model 117
CIM operations 84, 125
CIM operations over HTTP 4
CIM provider
See provider
CIM Query Language 9
CIM registered profile
HBA 117, 119
HDR 117
CIM registered profiles 117
SMI-S 117
CIM Schema 4
CIM server 4
access 24, 28
administration 61
automatic restart 70
configuration 53
configuration backup 70
configuration for audit logging 69
logging configuration 67
logon 28
quick security setup 19
quick setup 19
CIM server (continued)
request 23
runtime configuration 29
runtime environment 28
runtime environment security 11
security 11
setup 23
setup verification 19
authorization to register to ARM 36
CIM_AlertIndication 288
CIM_ComputerSystem 127
CIM_ComputerSystemPackage 141
CIM_ControlledBy 234
CIM_DeviceSAMImplementation 234
CIM_ElementSoftwareIdentity 234
CIM_ElementStatisticalData 234
CIM_EthernetPort 156
CIM_FCPort 233
CIM_FCPortStatistics 233
CIM_HostedAccessPoint 234
CIM_HostedFileSystem 151
CIM_Indication 287, 288
CIM_IndicationFilter 9, 290
CIM_IndicationSubscription 9, 291
CIM_InitiatorTargetLogicalUnitPath 234
CIM_InstalledSoftwareIdentity 234
CIM_InstIndication 287, 289
CIM_InstModification 289
CIM_IPProtocolEndpoint 156
CIM_RemoteFileSystem 151
CIM_RunningOS 128
CIM_SNMPTrapIndication 289
CIM_StorageExtent 233
CIM_StorageExtentID 233
CIM_SystemDevice 143, 156, 235
cimcli 84

© Copyright IBM Corp. 2005, 2011 355
cimcli (continued)
a, associators 85
an, associatorNames 86
ci, createInstance 87
dc, deleteClass 88
dl, deleteInstance 89
dq, deleteQualifier 90
ec, enumerateClasses 91
ei, enumerateInstances 92
eq, enumerateQualifiers 93
gc, getClass 94
gi, getInstance 95
gq, getQualifier 96
im, invokeMethod 97
instance specification 109
mi, modifyInstance 98
nc, enumerateClassNames 99
ni, enumerateInstanceNames 100
ns, enumerateNamespaces 101
options 107
r, references 102
rn, referenceNames 103
sp, setProperty 104
ti, testInstance 105
xq, execQuery 106
cimconfig 63, 64, 79
cimconfig 63, 64, 79
CIMIVP 57
cimmof 77
cimprovider 43, 81
CIMSERV 11, 13, 25, 28, 75
definition 24
cimserver command 62
cimserver_planned.conf 70
cimserver.env 17, 18, 49, 51, 61, 70
cimserver.err 49
cimserver.out 49
cimserver.trc 65
cimsb 110
class descriptor table 24
class SURROGAT 25
client 4, 5
authentication 29, 31
user 29
Cluster classes 122, 191
Cluster provider setup 38
CMPI 271
header files 272
CMPI function signatures 274
initialization 274
samples 285
security 275
command-line utilities 75
commands 75
comments
how to send your comments to IBM xiii
Common Area Data Space 39
Common Event Adapter
See CEA
Common Information Model
See CIM
Common Manageability Programming Interface
See CMPI
compilation 77
concepts of CIM 3
configuration properties 53
dynamic 53, 63
for tracing 64
modification 113
modification, current 63
modification, planned 63, 113
configuration property
daemon 53
disableAuditLog 53, 69
disableHttpConnection 53
disableHttpsConnection 29, 54
disableIndicationService 54
disableRemotePrivilegedUserAccess 54
forceProviderProcesses 54, 62, 285
httpPort 45, 54
httpsPort 29, 45, 54
idleConnectionTimeout 54
logLevel 54, 67, 68
maxFailedProviderModuleRestarts 55
maxProviderProcesses 55
messageDir 55
providerDir 55
repositoryDir 55
repositoryIsDefaultInstanceProvider 55
shutdownTimeout 55
slp 55
socketWriteTimeout 55
traceComponents 56, 64
traceFacility 56, 65, 67
traceFilePath 56, 65
traceLevel 56, 64
traceMemoryBufferKbytes 56, 65
connectivity 117
console 67, 70, 113
contacting IBM xiii
CoupleDataset provider setup 38
CQL 9
createClass 84
current configuration property 63, 79
directory paths (continued)
SMP/E target library path 17
started task environment 48
disability 349
Distributed Data Server
See DDS
Distributed Management Task Force
See DMTF
DMTF xi, 3
dynamic
CDT 24
directory paths (continued)
ESMTERM termination type 71
enableAuditLog configuration property 53, 69
disableHttpConnection configuration property 53
disableHttpsConnection configuration property 29, 54
disableIndicationService configuration property 54
disableRemotePrivilegedUserAccess configuration property 54
encryption 11, 29, 31
key 35
Enhanced ASCII 323
Enhanced Security model 13, 25
EnumerateInstanceNames 125
EnumerateInstances 125
environment variable 51
_BPX_JOBNAME 69
_BPX_SHAREAS 28
LIBPATH 51
OSBASE_TRACE 51
OSBASE_TRACE/File 51
PATH 51
Pegas US_HOME 50, 51
RMF_CIM_BENCH 52
RMF_CIM_HOST 51
RMF_CIM_PORT 52
RMF_CIM_PROVIDER 51
RMF_CIM_TRACE 52
RMF_CIM_TRACE/File 52
RMF_INDICATION _POLLING_INTERVAL 52
RMF_INDICATION_RESTTIME 52
WLM_CIMPROVIDER _TRACE_FILE 52
WLM_CIMPROVIDER _TRACE_LEVEL 52
environment variables
file cimserver.env 17, 49, 51, 70
file profile.add 49, 50, 323
modification 51
EOTRACE 65
error messages 323
eServer CIM 6
event 7
subscription 7
Event Model 7
event provider function 275
INDEX

F
fallback 18
FATAL log level 67

G
GetInstance 125
GID
definition, CIM server 25
group ID See GID

H
HBA 117, 119
Hot Swap Events 119
instance diagram 119
HDR 117
header files
  CMPI 272
host adapter 117
Host Discovered Resources
  instance diagram 118
profile 117
Host-Bus-Adapter
  See HBA
HTTP 21, 28
HTTP port 45
HTTP_NOAUTH 37
httpPort
  configuration property 45
httpPort configuration property 45, 54
HTTPS 21, 28, 29
HTTPS port 45
httpsPort
  configuration property 45
httpsPort configuration property 29, 54

I
IBM_BaseBoard 139
IBMzOS
  _SBDeviceSAPIImplementation 256
  _SBInitiatorTargetLogicalUnitPath 149, 258
IBMzOS_BaseBoard 140
IBMzOS_CFRMCoupleDataset 192
IBMzOS_CFRMPolicy 194
IBMzOS_CFStrDependsOn 228
IBMzOS_CFStructure 196
IBMzOS_CFStructureConnector 206
IBMzOS_CollectionOfCFs 228
IBMzOS_CollectionOfSysplexNodes 229
IBMzOS_ComputerSystem 129
IBMzOS_ControlledBy 249
IBMzOS_CoupleDataset 209
IBMzOS_CouplingFacility 213
IBMzOS_CouplingFunction 217
IBMzOS_CSBaseBoard 141
IBMzOS_CSFCPort 250
IBMzOS_CSFCPortController 251
IBMzOS_ElementSoftwareIdentity 252
IBMzOS_EthernetPort 157
IBMzOS_FCPort 236
IBMzOS_FCPortStatisticalData 253
IBMzOS_FCPortStatistics 239
IBMzOS_HostedCFStrConnector 230
IBMzOS_HostedCFStructure 229
IBMzOS_InstalledSoftwareIdentity 254
IBMzOS_IPProtocolConnector 158
IBMzOS_JES2Job 161
IBMzOS_JES2SysoutDataset 180
IBMzOS_JES3Job 171
IBMzOS_JES3SysoutDataset 183
IBMzOS_Job 184
IBMzOS_JobsManagementSettings 185
IBMzOS_LogicalDisk 148
IBMzOS_LogicalDiskDevice 149
IBMzOS_NFS 153
IBMzOS_OperatingSystem 131
IBMzOS_OSProcess 133
IBMzOS_PortController 241
IBMzOS_Process 134
IBMzOS_Processor 144
IBMzOS_Product 243
IBMzOS_ProductElementComponent 255
IBMzOSRunningOS 136
IBMzOS_SHostedAccessPoint 257
IBMzOS_SBProtocolEndpoint 244
IBMzOS_SVMAttributes 220
IBMzOS_SoftwareIdentity 246
IBMzOS_Subsystem 186
IBMzOS_SubsystemJES2Jobs 189
IBMzOS_SubsystemJES3Jobs 189
IBMzOS_SysoutDataset 188
IBMzOS_Sysplex 221
IBMzOS_SysplexCoupleDataset 224
IBMzOS_SysplexNode 225
IBMzOS_UNIXLocalFileSystem 152
IBMzOS_UNIXProcess 137
IBMzOS_UseCFRMCoupleDatasets 231
IBMzOS_UseCFRM massively 231
IBMzOS_UseCFs 230
IBMzOS_UseCouplingFunctions 231
IBMzOS_UseCFsJES2SysoutDatasets 189
IBMzOS_UseCFsJES3SysoutDatasets 189
IBMzOS_UseCFsSysplexCoupleDatasets 231
IBMzOS_WLM 262
IBMzOS_WLMOS 267
ICHRIN03 started procedures table 49
identity switch 29
idleConnectionTimeout configuration property 54
indication 7, 287
end point 33
hierarchy 287
polling interval 52
provider function 9
secured 33
stream 289, 290
subscription 52, 289
indication delivery retry 8
INFORMATION log level 67
installation 17
directories 17
verification 19, 22, 57
installation verification program See IVP
instance provider function 274
introduction to CIM 3
IOS service 41
IOSCDR 41
IOSCP 41
IVP 19, 22, 57
IXCARM 36
IXCL1DSU format utility 39

J
Java
  CIM client 4
JES2Job provider setup 38
JES3Job provider setup 38
Job classes 122, 160
key 30, 35
key ring 30
keyboard 349

L
LIBPATH environment variable 17, 51
lifetime 8
log 67
level 68
FATAL 67
INFORMATION 67
SEVERE 67
TRACE 67
WARNING 67
level modification 67
message 65, 67, 68
records for audit logging 69
routing 66
logging 67, 68
See log
Logical Disk classes 146
logical storage resource 117
logLevel configuration property 54, 67
LogMessages trace component 66, 67
logon
to the CIM server 28, 35

M
managed object format
  See MOF
management instrumentation
  for additional z/OS resources 271
master repository 17
maxFailedProviderModuleRestarts configuration property 55
maxProviderProcesses configuration property 55

Index 357
memory dump 65
message
  shutdown 70
startup 70
messageDir configuration property 55
messages 29
troubleshooting 32
method provider function 27
migration 17
MLS 35, 54
MODIFY console command 63, 64, 113
modifyClass 84
modifyInstance 84
ModuleGroupName 63, 285
MOF 75
  compiler 77
  file, provider registration 43
  monitoring provider 51
multi level security
  See MLS
must-stay-clean 26

N
Network classes 122, 154
Network provider setup 38
NOOOICARD attribute 25
NOPASSWORD attribute 25
NOPHRASE attribute 25
O
oidcard 42
OOP
  See out-of-process support
OpenPegasus 4
OS (operating system) 5
OS management
  Base classes 121, 126
  BaseBoard classes 121, 138
  Cluster classes 122, 191
  File System classes 121, 150
  Job classes 122, 160
  Logical Disk classes 146
  Network classes 122, 154
  Processor classes 121, 142
  OSBASE_TRACE 51
  OSBASE_TRACE_FILE 51
  out-of-process support 26, 54, 62
    developing providers OOP 285
    tracing providers 65
  out-of-process support
    development 26, 54, 62
    environment 28
    environment security 11
performance implications 65
PG_Provider 43, 277, 280
PG_ProviderCapabilities 277, 283
PG_ProviderModule 43, 277, 281
planned configuration property 63, 79
  file cimserver_planned.conf 70
policy
  inbound 31, 32
  outbound 32, 33
Policy Agent
  setup 30
  port 45
  port 5980 21
  port 5988 21
  port number 45
  POSIT 24, 329
PreviousInstance 289
PreviousInstance property 288
priority for z/OS CIM 52
privileged user 25, 26
Processor classes 121, 142
profile.add 49, 50, 323
program control 27
protected user
definition 25
protection of resources 11
provider 4
  based authorization 13, 43
  CMPI 271
development 271
disabling 81
enabling 81
function signatures 274
function, association 275
function, indication 275
function, instance 274
function, method 275
IBM-supplied 121
initialization 274
listing 81
profile 13
registration 277
registration compilation 77
registration MOF file 43, 285
registration processing 278
registration schema 277
removal 81
RMF 51
samples 285
security 275
setup 38, 39
trace 65
WLM 52
provider agent 26, 65, 285
providerDir configuration property 55
ProviderModuleName 280
PTKTDATA 35, 37
publish/subscribe event paradigm 8
repository 17, 55, 77
backup 73
maintenance 73
recovery 73
repositoryDir configuration property 55
repositoryIsDefaultInstanceProvider
collection configuration property 55
requestor 23, 25, 42
user ID 42
resource
  access 42
  authorization model 25
  Resource Access Control Facility
  See RACF
  Resource Measurement Facility
  See RMF
restart
  automatic restart using ARM 70
REXX 39
RMF 28
  provider 51
  RMF monitoring 37
RMF_CIM_BENCH 52
RMF_CIM_HOST 37, 51
RMF_CIM_PORT 52
RMF_CIM_PROVIDER 51
RMF_CIM_TRACE 52
RMF_CIM_TRACE_FILE 52
RMF_INDICATION 52
RMF_INDICATION_POLLING_INTERVAL 52
RMF_INDICATION_RESTTIME 52
runtime
collection configuration 29
environment 28
environment security 11
S
SAF 11, 29, 30, 32, 36, 43, 69, 280, 299,
  306, 307, 308, 309
SAF profile
  BPX.SERVER 20, 25
  BPX.SMF 69
  CFZAPPL 35
  CIMSERV 24, 28
SAF profiles
  security 43
SB Multipath Management
  profile 117, 118
SBLIM 4, 285
security 11
  aspects for developing providers 275
  context 42
  of the network 12
  profile 11, 13, 43
  quick setup for the CIM server 19
  setup for CIM server 23
  setup for the CIM server, quick 19
  using ARM 36
SecurityAccessProfile 43, 277, 280
sequence identifier 8
service class for CIM priority 52
SEVERE log level 67
ShareAS 63, 285
shortcut keys 349
shutdown message 70
World Wide Port Number
See WWPN

WS-Enumeration 4
WS-Management xv, 4
WS-Transfer 4
WWPN 41, 244

X
XCF address space
See XCFAS
XCFAS 36

Z
z/OS Communications Server 125
  system logger
    See syslog
z/OS Security Server 24, 28
z/OS system console
  See console
zFS data set 46