Bull

Common Information Model Guide

AIX

ORDER REFERENCE
86 A2 50EM 01

Bull

Common Information Model Guide

AIX

Software

October 2005

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

ORDER REFERENCE
86 A2 50EM 01

Printed in France

Suggestions and criticisms concerning the form, content, and presentation of this book are invited. A form is provided at the end of this book for this purpose.

To order additional copies of this book or other Bull Technical Publications, you are invited to use the Ordering Form also provided at the end of this book.

Trademarks and Acknowledgements

We acknowledge the right of proprietors of trademarks mentioned in this book.

AIX® is a registered trademark of International Business Machines Corporation, and is being used under licence.

UNIX is a registered trademark in the United States of America and other countries licensed exclusively through the Open Group.

Linux is a registered trademark of Linus Torvalds.

The information in this document is subject to change without notice. Bull will not be liable for errors contained herein, or for incidental or consequential damages in connection with the use of this material.

Contents

About This Book

This book provides system administrators with complete information about how to perform tasks including installing, configuring, securing, and troubleshooting the Common Information Model.

Highlighting

The following highlighting conventions are used in this book:

Bold Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.

Italics Identifies parameters whose actual names or values are to be supplied by the user.

`Monospace` Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

Case-Sensitivity in AIX

Everything in the AIX® operating system is case-sensitive, which means that it distinguishes between uppercase and lowercase letters. For example, you can use the **ls** command to list files. If you type `LS`, the system responds that the command is ″not found.″ Likewise, **FILEA**, **FiLea**, and **filea** are three distinct file names, even if they reside in the same directory. To avoid causing undesirable actions to be performed, always ensure that you use the correct case.

ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

Chapter 1. Common Information Model overview

The Common Information Model (CIM) is a conceptual information model for describing management properties that is not bound to a particular implementation. This allows for the interchange of management information between management systems and applications through the Common Information Model Object Manager (CIMOM), which is an object management engine that exists between the managed system and the management application. Management applications can be local or remote, but currently CIMOM and the programs that collect management data, or *providers*, must be located on the machine that is being instrumented. Pegasus is one of the open source implementations of the CIMOM that adheres to the Distributed Management Task Force (DMTF) CIM and Web-based Enterprise Management (WBEM) standards. Pegasus is designed to be inherently portable and builds and runs on the AIX, Linux, and Microsoft Windows operating systems.

The CIM Standard Schema provides the actual model descriptions. The CIM Schema supplies a set of classes with properties and associations that provide a conceptual framework within which it is possible to organize the available information about the managed environment. Platform-specific objects, such as AIX, that must be managed are defined as extensions to this standard CIM model.

Providers collect the management data from the underlying platform resources and populate the CIM objects described in the conceptual CIM model. These objects are then ready to be served by the CIMOM to the client management applications for managing the resources of the underlying platform. This mechanism provides an open-standard way for a management application to manage the resources of the underlying platform.

CIM on AIX 5.3 includes the following:

- An open-source implementation of the CIMOM called Pegasus Version 2.5
- A CIM schema, version 2.9, that defines an information model for representing system management resources
- Providers that instrument a set of AIX resources and AIX diagnostics events based on CIM schema, version 2.9

For additional information about Pegasus and WBEM, see the following Web sites:

- The OpenPegasus Web site at http://www.openpegasus.org/
- The DMTF Web site at http://www.dmtf.org/standards/cim
- The DMTF Web site at http://www.dmtf.org/standards/wbem
- The WBEM Web site at http://www.wbemsolutions.com/tutorials/CIM/cim.html

Chapter 2. Install the Pegasus CIM Server and base providers

AIX ships the following packages to support Pegasus:

- **sysmgt.pegasus.cimserver** installs the Pegasus CIM Server filesets in the **/opt/freeware/cimom/pegasus** directory
- **sysmgt.pegasus.osbaseproviders** installs the base providers for AIX filesets in the **/usr/pegasus/provider** directory

You can install the packages using either the System Management Interface Tool (SMIT) or the **installp** command.

For more information about using the **installp** command, see the installp command in *AIX 5L Version 5.3 Commands Reference, Volume 3*.

Note: Before continuing with the installation, review the license information.

To install the packages using SMIT, complete the following:

1. At the command line, type `smitty`.
2. Select **Software Installation and Maintenance**>**Install and Update Software**>**Install Software**.
3. At the Input Device/directory for software field, press the F4 key to view a list of options.
4. Select the option that reflects the location or media that contains the CIM packages.
5. At the Software to Install field, press the F4 key to view a list of package options.
6. Select the **sysmgt.pegasus.cimserver** and **sysmgt.pegasus.osbaseproviders** packages by pressing the F7 key.

To verify that the CIM Server filesets were installed correctly, use the **lslpp** command as follows:

```
lslpp -al sysmgt.pegasus.cimserver.rte
```

- If the installation completed successfully, a message similar to the following is returned:

  ```
  lslpp -l sysmgt.pegasus.cimserver.rte
    Fileset                      Level  State      Description
    ----------------------------------------------------------------------------
  Path: /usr/lib/objrepos  sysmgt.pegasus.cimserver.rte  2.3.1.0    COMMITTED \
      Pegasus CIM Server Runtime Environment
  ```

- If the installation did not complete successfully, a message similar to the following is returned:

  ```
  lslpp: Fileset sysmgt.pegasus.cimserver.rte not installed.
  ```

To verify that the base providers for AIX filesets were installed correctly, use the **lslpp** command as follows:

```
lslpp -al sysmgt.pegasus.osbaseproviders
```

- If the installation completed successfully, a message similar to the following is returned:

  ```
  lslpp -l sysmgt.pegasus.osbaseproviders
    Fileset                      Level  State      Description
    ----------------------------------------------------------------------------
  Path: /usr/lib/objrepos  sysmgt.pegasus.osbaseproviders  1.2.3.0   COMMITTED \
      Base Providers for AIX OS
  ```

- If the installation did not complete successfully, a message similar to the following is returned:

  ```
  lslpp: Fileset sysmgt.pegasus.osbaseproviders not installed.
  ```

Install the OpenSSL RPM file

In order for the CIM Server to run, the OpenSSL rpm file must be installed. To determine if the rpm file is installed on your system, run the following commands:

```
rpm -q -f /opt/freeware/lib/libssl.a
rpm -qa | grep -i openssl
```

If both the libssl.a library and the openssl-0.9.6*XXX* rpm, where *XXX* indicates the build level, are found, then OpenSSL is installed on your system.

If OpenSSL is not already installed, you can find this rpm file on the *AIX Linux® ToolBox* CD. It can also be downloaded from the AIX Toolbox for Linux Applications Web site at http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html.

On this Web site, select **AIX Toolbox Cryptographic Content** under the Sorted Download heading on the right of the page. After you have registered and accepted the license, you can download the ″openssl - Secure Sockets Layer and cryptography libraries and tools″, such as **openssl-0.9.6k-1.aix4.3.ppc.rpm**, or later version.

To install the OpenSSL rpm file, run the following command:

```
rpm -ivh openssl-0.9.6XXX.rpm
```

where *XXX* indicates the build level.

Chapter 3. Configure the CIM Server

Your CIM Server requires limited configuration. After you install both fileset packages for the server and the providers, your Pegasus CIM Server is ready for use.

The AIX Pegasus CIM Server is SSL-ready and requires the **libssl.a** library to be available on the system for the CIM Server to run even if you decide not to use SSL. The **libssl.a** library is a part of the OpenSSL rpm file. For more information about enabling the CIM Server to run with SSL, see “Secure your CIM Server.”

Start and stop the CIM Server

The CIM Server does not start or stop automatically on your system. To start or stop the CIM Server, you must run the following commands:

cimserver
Starts the CIM Server

cimserver -s
Stops the CIM Server

The CIM Server runs as a daemon in the background. By default, Basic Authentication is enabled. When Basic Authentication is enabled, the CIM Server authenticates the user ID and password of each request.

To disable Basic Authentication, set `enableAuthentication=false` in the configuration files before starting the CIM Server. If the CIM Server is already running, it must be restarted for the change to take effect. Client requests to the CIM Server must include a user ID and password if Basic Authentication is enabled.

Set resource limits

To maximize the CIM Server’s processing capacity, the root user’s **ulimit** for the Soft DATA segment must be set to -1, which is ″unlimited″.

To set resource limits in SMIT, complete the following:

1. Type `smitty user` at a command prompt.
2. Select **Change/Show Characteristics of a User**.
3. Type `root` in the **User NAME** field.
4. Scroll to the **Soft DATA segment** field and change the value to `-1`.

In order for the change to take effect, you are required to log out and log back in again.

For more information about user and system resource limits, refer to the **ulimit** command in the Commands Reference.

Secure your CIM Server

To properly secure your CIM Server, it is recommended that you enable SSL when you are running the CIM Server. If you prefer to run your CIM Server in an unsecure mode, you can do so by either not enabling SSL, or disabling SSL after you enable it.

Create SSL certificates

To enable your server to run in SSL mode, you need a certificate and two keys: one public key and one private key. Because the private key contains the public key information in OpenSSL, a separate public key is not required. For more information about public and private keys in OpenSSL, see the OpenSSL Web site at http://www.openssl.org.

To create a self-signed test certificate called **cert.pem** and an RSA private key called **file.pem**, run the following commands:

```
cd /opt/freeware/cimom/pegasus/etc
CN="Common Name"
EMAIL="test@email.address"
HOSTNAME=`hostname`
sed -e "s/$CN/$HOSTNAME/" -e "s/$EMAIL/root@$HOSTNAME/" orig/ssl.cnf > ssl1.cnf
/opt/freeware/bin/openssl req -x509 -days 365 -newkey rsa:512 -nodes \
                                 -config ssl1.cnf -keyout file.pem -out cert.pem

cp cert.pem client.pem

rm ssl1.cnf
```

The **ssl.cnf** file is a sample configuration file for OpenSSL. Two fields in the file, `CN="Common Name"` and `EMAIL="test@email.address"`, are used to specify the identity to be certified. Use the **sed** command string to replace the placeholder information with your system's information. You can customize the common name and the e-mail address to match your environment.

Output from the **sed** command is directed to the temporary **ssl1.cnf** file.

The default location for the key and certificate files is the **/opt/freeware/cimom/pegasus/etc** directory. If another directory is used, it must be specified using the **cimconfig** command before the CIM Server is started.

For example, if you choose to store the certificate and key files in a directory called **/var/pegasus**, specify the location of these files using the following commands before starting the CIM Server:

- `cimconfig -p -s sslCertificateFilePath=/var/pegasus/cert.pem`
- `cimconfig -p -s sslKeyFilePath=/var/pegasus/file.pem`

The **cp cert.pem client.pem** command creates the **client.pem** trust file, which is a copy of the self-signed test certificate. Its default location is the **/opt/freeware/cimom/pegasus/etc** directory.

Enable the CIM Server with SSL

After the SSL certificates are created, enable the CIM Server with SSL by setting the following parameters to the specified values:

- *enableHttpsConnection*=true
- *enableHttpConnection*=false

These parameters are set in one of the following places:

- the **cimconfig** command. For more information about the **cimconfig** command, see Appendix B, "cimconfig Command," on page 35.
- command-line options to the **cimserver** command during the startup of the CIM Server. For more information about the **cimserver** command, see Appendix F, "cimserver Command," on page 47.

Set and configure CIM Server tracing

Tracing should only be used for debugging purposes and is, by default, disabled. You can, however, enable the tracing mechanism by specifying the trace level and the components that you want traced with the *traceLevel* configuration parameter. Choose from one of the following trace levels:

level 1
Function entry and exit

level 2
Basic flow trace messages, low data detail

level 3
Interfunction logic flow, medium data detail

level 4
All information, high data detail

Trace data is saved in the file specified by the *traceFilePath* configuration parameter. By default, the *traceFilePath* parameter is set to the **/opt/freeware/cimom/pegasus/logs/cimserver.trc** file.

To specify the component or components that you want to trace, use the *traceComponents* parameter. Some of the components that you can specify with this parameter include the following:

- Channel
- XmlWriter
- CimData
- ProvManager
- Authorization
- Authentication
- WQL
- Thread

You can also choose to trace all of the components by using the word ALL in place of a specific component name. If the *traceComponents* parameter is not set to any component, tracing is off regardless of the *traceLevel* parameter setting.

You can modify the trace configuration parameters by using one of the following methods:

- In the configuration files before the CIM Server is started. Use your favorite editor to change the values of the parameters.
- As command line options to the **cimserver** command during the startup of the CIM Server.
- As command line options to the **cimconfig** command while the CIM Server is running.

For example, to set the trace level to trace all information with high data detail in the **Thread** and **ProvManager** components, type the following commands:

```
cimconfig -s traceLevel=4
cimconfig -s traceComponents=Thread,ProvManager
```

Similarly, to disable all tracing, type the following command:

```
cimconfig -s traceComponents=
```

For more information about setting configuration parameters with the **cimconfig** command, see Appendix B, “cimconfig Command,” on page 35.

Configure logging

The logging utility that is available for the Pegasus CIM Server is initially enabled and cannot be disabled. However, you can configure the utility by choosing the level of logging to use and specifying the directory in which to store the log files.

The following are the available levels of logging:

- TRACE
- INFORMATION
- SEVERE
- FATAL

You can change the logging level during the CIM startup by specifying the logging level with the **logLevel** parameter through one of the following methods:

- In the configuration files before the CIM Server is started
- As an option to the **cimserver** command during the CIM Server startup
- As an option to the **cimconfig** command while the CIM Server is running

For example, if you chose to use the **cimconfig** command method, you would type the following while the CIM Server is running:

```
cimconfig -s logLevel=INFORMATION
```

The log data is saved in the following files:

- **PegasusDebug.Log**
- **PegasusError.log**
- **PegasusStandard.log**
- **PegasusTrace.log**

These files are located in the directory that you specify with the **logdir** configuration parameter. By default, the **logdir** parameter is set to the **/opt/freeware/cimom/pegasus/logs** directory.

Chapter 4. Restore and rebuild the repository

The CIM Repository is located at **/opt/freeware/cimom/pegasus/etc/repository**, and is pre-loaded with the CIM Schema, and the AIX extended classes of the CIM Schema. This repository is the active repository that can be modified with the **cimmof** and **cimmofl** commands.

An identical copy of the CIM Repository that is not used or modified by the CIM Server is also included. This repository copy exists in the **/opt/freeware/cimom/pegasus/etc/orig/repository** directory for recovery purposes only.

The server repository can be restored to its original state by using either of the following options:

- **Restore the repository.** This option is recommended for all users.

  To restore your repository, completing the following steps:

  1. Verify that the CIM Server is not running. To stop the CIM Server, type the following command:

     ```
     cimserver -s
     ```

     For more information about stopping the CIM Server, see Appendix F, "cimserver Command," on page 47.
  2. Remove the active repository by typing the following command:

     ```
      rm -rf /opt/freeware/cimom/pegasus/etc/repository
     ```
  3. Copy the original repository by typing the following command:

     ```
     cp -pr /opt/freeware/cimom/pegasus/etc/orig/repository /opt/freeware/cimom/pegasus/etc
     ```
  4. After the repository is restored, start the CIM Server.

  This option restores the repository to the original state that was shipped with the package. Any additional classes that were added to the repository must be reloaded using the **cimmof** or **cimmofl** commands. For more information about these commands, see Appendix C, "cimmof Command," on page 37 and Appendix D, "cimmofl Command," on page 41.
- **Rebuild the repository.**

  To rebuild your repository, complete the following steps:

  1. Verify that the CIM Server is not running. To stop the CIM Server, type the following command:

     ```
     cimserver -s
     ```

     For more information about stopping the CIM Server, see Appendix F, "cimserver Command," on page 47.
  2. Run the **rebuild_repository** script by typing the following command:

     ```
     /opt/freeware/cimom/pegasus/etc/orig/rebuild_repository
     ```

     Before the script continues, you will be asked to confirm the request to remove the active repository.
  3. After the repository is rebuilt, start the CIM Server. For more information about starting the CIM Server, see Appendix F, "cimserver Command," on page 47.

  This option restores the repository to the original state that was shipped with the package. Any additional classes that were added to the repository must be reloaded using the **cimmof** or **cimmofl** commands. For more information about these commands, see Appendix C, "cimmof Command," on page 37 and Appendix D, "cimmofl Command," on page 41.

Chapter 5. Providers for AIX

The Pegasus CIM Server uses providers to manage system resources. These providers must follow the implementation rules defined by the CIMOM. The Pegasus CIM Server uses its own proprietary C++ provider Application Programming Interface (API), but also works with the Common Manageability Programming Interface (CMPI), a C provider interface. Providers written to the CMPI are interoperable with the Pegasus CIM Server and other CIMOMs without any changes. For more information about CMPI, see the www.openpegasus.org Web site.

MOF files

A **.mof** file is a text file that defines the class name and attributes of a managed resource. The format adheres to the CIM standard of the DMTF industry consortium. An example of a managed resource is an operating system, and the MOF file can include the following types of attributes:

- Operating system type
- Version
- Date installed
- Number of users

The **.mof** files that define the IBM providers are located in the **/usr/pegasus/provider** directory.

Providers

The following provider types are included in the **sysmgt.pegasus.osbaseproviders** package and are used by the Pegasus CIM Server:

Instance Providers
Provide a dynamic list of instances of a class that are available in a CIMOM. For example, an operating system Instance provider returns information about the instance of the operating system that is running on the system.

Association Providers
Build associations between instances of different classes dynamically. For example, the Pegasus CIM server contains information about the computer system and the operating system. An association provider associates the computer system and the operating system and returns information about this association.

Indication Providers
Generates indications (events) for the defined error conditions in a specific resource (for example, the operating system).

The following CIM classes have been implemented by IBM-supplied providers.

CIM Class	IBM provider
IBMAIX_AFS	"OSBase_AFSProvider" on page 13
IBMAIX_BlockStorageStatisticalData	"OSBase_BlockStorageStatisticalDataProvider" on page 13
IBMAIX_BootOSFromFS	"OSBase_BootOSFromFSProvider" on page 14
IBMAIX_CDFS	"OSBase_CDFSProvider" on page 14
IBMAIX_CSNetworkPort	"OSBase_CSNetworkPortProvider" on page 14
IBMAIX_CSProcessor	"OSBase_CSProcessorProvider" on page 15
IBMAIX_ComputerSystem	"OSBase_ComputerSystemProvider" on page 15

CIM Class	IBM provider
IBMAIX_DFS	“OSBase_DFSProvider” on page 15
IBMAIX_EthernetPort	“OSBase_EthernetPortProvider” on page 16
IBMAIX_FCPort	“OSBase_FCPortProvider” on page 16
IBMAIX_HostedFileSystem	“OSBase_HostedFileSystemProvider” on page 17
IBMAIX_IPProtocolEndpoint	“OSBase_IPProtocolEndpointProvider” on page 17
IBMAIX_JFS2	“OSBase_JFS2Provider” on page 18
IBMAIX_JFS	“OSBase_JFSProvider” on page 18
IBMAIX_LocalLoopbackPort	“OSBase_LoopBackPortProvider” on page 19
IBMAIX_NFS	“OSBase_NFSProvider” on page 20
IBMAIX_NetworkPortImplementsIPEndpoint	“OSBase_NetworkPortImplementsEndpointProvider” on page 20
IBMAIX_OSProcess	“OSBase_OSProcessProvider” on page 20
IBMAIX_OperatingSystem	“OSBase_OperatingSystemProvider” on page 20
IBMAIX_OperatingSystemStatisticalData	“OSBase_OperatingSystemStatisticalDataProvider” on page 21
IBMAIX_PROCFS	“OSBase_PROCFSProvider” on page 22
IBMAIX_Processor	“OSBase_ProcessorProvider” on page 22
IBMAIX_RunningOS	“OSBase_RunningOSProvider” on page 23
IBMAIX_TokenRingPort	“OSBase_TokenRingPortProvider” on page 23
IBMAIX_UnixProcess	“OSBase_UnixProcessProvider” on page 24
IBMPSG_BIOS	“IBMPSG_BIOSProvider” on page 24
IBMPSG_Baseboard	“IBMPSG_BaseboardProvider” on page 25
IBMPSG_CacheMemory	“IBMPSG_CacheMemoryProvider” on page 25
IBMPSG_ComputerSystem	“IBMPSG_ComputerSystemProvider” on page 26
IBMPSG_FRU	“IBMPSG_FRUProvider” on page 26
IBMPSG_LogicalDisk	“IBMPSG_LogicalDiskProvider” on page 27
IBMPSG_OperatingSystem	“IBMPSG_OperatingSystemProvider” on page 28
IBMPSG_PhysicalDisk	“IBMPSG_PhysicalDiskProvider” on page 28
IBMPSG_PhysicalMemory	“IBMPSG_PhysicalMemoryProvider” on page 29
IBMPSG_Processor	“IBMPSG_ProcessorProvider” on page 30
IBMPSG_Slot	“IBMPSG_SlotProvider” on page 30
IBMP_CSServiceEventIndication	“IBMPSG_IndicationProvider” on page 31
IBMPSG_FanEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_MemoryPFEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_NetworkAdapterFailedEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_PowerSupplyEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_ProcessorPFEvent	“IBMPSG_IndicationProvider” on page 31
IBM_SystemEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_TemperatureEvent	“IBMPSG_IndicationProvider” on page 31
IBMPSG_VoltageEvent	“IBMPSG_IndicationProvider” on page 31

OSBase_AFSProvider

This provider is an instance provider for the AIX_AFS class. It conforms to the CMPI interface to provide information about remote AFS® file systems.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_AFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_BlockStorageStatisticalDataProvider

This provider is a statistical provider for the IBMAIX_BlockStorageStatisticalData class. It provides statistical data for the block storage system.

The following provider properties are implemented:

Property	Type
TotalIOs	uint64
KBytesTransfered	uint64
IOTimeCounter	uint64
ReadIOs	uint64
ReadHitIOs	uint64
ReadIOTimeCounter	uint64
ReadHitIOTimeCounter	uint64
KBytesRead	uint64
WriteIOs	uint64
WriteHitIOs	uint64
WriteIOTimeCounter	uint64
WriteHitIOTimeCounter	uint64
IdleTimeCounter	uint64
MaintOp	uint64
MaintTimeCounter	uint64

For more information about the properties descriptions, see the IBMAIX_BlockStorageStatisticalData class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_BootOSFromFSProvider

This provider is an instance provider for the IBMAIX_BootOSFromFS class. It conforms to the CMPI interface to provide the association between the current, running operating system and the file system from which the operating system is booted.

When an instance is returned, the properties of the IBMAIX_OperatingSystem and subclasses of CIM_FileSystem classes, listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_BootOSFromFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_CDFSProvider

This provider is an instance provider for the IBMAIX_CDFS class. It conforms to the CMPI interface to provide information about local CDFS file systems.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_CDFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_CSNetworkPortProvider

This provider is an instance provider for the IBMAIX_CSNetworkPort class. It conforms to the CMPI interface to provide the association between a computer system and its network ports.

When an instance is returned, the properties of the IBMAIX_ComputerSystem and subclasses of CIM_NetworkPort (Ethernet, TokenRing, FCPort and LoopBackPort) classes listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_CSNetworkPort class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_CSProcessorProvider

This provider is an instance provider for the IBMAIX_CSProcessor class. It conforms to the CMPI interface to provide the association between a computer system and the processors that the computer system contains.

When an instance is returned, the properties of the IBMAIX_ComputerSystem and subclasses of IBMAIX_Processor classes, listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_CSProcessor class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_ComputerSystemProvider

This provider is an instance provider for the IBMAIX_ComputerSystem class. It conforms to the CMPI interface to provide computer system properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
ElementName	string
Status	string
Name	string
CreationClassName	string
NameFormat	string
Dedicated[]	uint16A
OtherDedicatedDescriptions	string

For more information about the properties descriptions, see the IBMAIX_ComputerSystem class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_DFSProvider

This provider is an instance provider for the IBMAIX_DFS class. It conforms to the CMPI interface to provide information about remote DFS™ file systems.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64

Property	Type
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_DFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_EthernetPortProvider

This provider is an instance provider for the IBMAIX_EthernetPort class. It conforms to the CMPI interface to provide the ethernet port properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
Element Name	string
Name	string
Status	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
MaxSpeed	uint64 (bps)
Speed	uint64 (bps)
OtherNetworkPortType	string
LinkTechnology	uint16
OtherLinkTechnology	string

For more information about the properties descriptions, see the IBMAIX_EthernetPort class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_FCPortProvider

This provider is an instance provider for the IBMAIX_FCPort class. It conforms to the CMPI interface to provide fibre channel port properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string

Property	Type
Element Name	string
Name	string
Status	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
MaxSpeed	uint64 (bps)
Speed	uint64 (bps)
OtherNetworkPortType	string
LinkTechnology	uint16
OtherLinkTechnology	string

For more information about the properties descriptions, see the IBMAIX_FCPort class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_HostedFileSystemProvider

This provider is an instance provider for the IBMAIX_HostedFileSystem class. It conforms to the CMPI interface to provide the association between the computer system container and its hosted local and remote file systems.

When an instance is returned, the properties of the IBMAIX_ComputerSystem and subclasses of CIM_FileSystem classes, listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_HostedFileSystem class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_IPProtocolEndpointProvider

This provider is an instance provider for the IBMAIX_IPProtocolEndpoint class. It conforms to the CMPI interface to provide IP protocol end point properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
ElementName	string
Status	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16

Property	Type
EnabledDefault	uint16
CreationClassName	string
SystemCreationClassName	string
SystemName	string
CreationClassName	string
Name	string
NameFormat	string
Protocol Type	uint16
OtherTypeDescription	string
IPv4Address	string
IPv6Address	string
SubnetMask	string
PrefixLength	uint8
IPVersionSupport	uint16

For more information about the properties descriptions, see the IBMAIX_IPProtocolEndpoint class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_JFS2Provider

This provider is an instance provider for the IBMAIX_JFS2 class. It conforms to the CMPI interface to provide information about local enhanced journaled file system.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_JFS2 class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_JFSProvider

This provider is an instance provider for the IBMAIX_JFS class. It conforms to the CMPI interface to provide information about local journaled file system.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_JFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_LoopBackPortProvider

This provider is an instance provider for the IBMAIX_LoopBackPort class. It conforms to the CMPI interface to provide loopback port properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
ElementName	string
Name	string
Status	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
LinkTechnology	uint16
OtherLinkTechnology	string

For more information about the properties descriptions, see the IBMAIX_LoopBackPort class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_NFSProvider

This provider is an instance provider for the IBMAIX_NFS class. It conforms to the CMPI interface to provide information about remote NFS file systems.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_NFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_NetworkPortImplementsEndpointProvider

This provider is an instance provider for the IBMAIX_NetworkPortImplementsEndpoint class. It conforms to the CMPI interface to provide the association between the IP protocol end point and the network ports.

When an instance is returned, the properties of the IBMAIX_IPProtocolEndpoint and subclasses of CIM_NetworkPort (ethernet, token-ring, FCport and loopbackport) classes listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_NetworkPortImplementsEndpoint class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_OSProcessProvider

This provider is an instance provider for the IBMAIX_OSProcess class. It conforms to the CMPI interface to provide the association between an operating system and the processes running in the operating system.

When an instance is returned, the properties of the IBMAIX_OperatingSystem and subclasses of IBMAIX_UnixProcess classes, listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_OSProcess class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_OperatingSystemProvider

This provider is an instance provider for the IBMAIX_OperatingSystem class. It conforms to the CMPI interface to provide AIX operating system properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
Name	string
CSCreationClassName	string
CSName	string
CreationClassName	string
OSType	uint16
Version	string
LastBootUpTime	datetime
LocalDateTime	datetime
CurrentTimeZone	sint16
NumberOfProcesses	uint32
MaxNumberOfProcesses	uint32
TotalVirtualMemorySize	uint64
FreeVirtualMemory	uint64 (in KBytes)
FreePhysicalMemory	uint64 (in KBytes)
TotalVisibleMemorySize	uint64 (in KBytes)
SizedStoredInPagingFiles	uint64 (in KBytes)
FreeSpaceInPagingFiles	uint64 (in KBytes)
MaxProcessorsMemorySize	uint64 (in KBytes)
Distributed	Boolean
CodeSet	string
LanguageEdition	string
DefaultPageSize	uint32

For more information about the properties descriptions, see the IBMAIX_OperatingSystem class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_OperatingSystemStatisticalDataProvider

This provider is a statistical provider for the IBMAIX_OperatingSystemStatisticalData class and provides statistical data for the operating system.

The following provider properties are implemented:

Property	Type
CPUUserTime	uint64
CPUSystemTime	uint64
CPUWaitTime	uint64
CPUIdleTime	uint64
RunQueueLength	uint64
BlockQueueLength	uint64

Property	Type
PagesPagedIn	uint64
PagesPagedOut	uint64

For more information about the properties descriptions, see the IBMAIX_OperatingSystemStatisticalData class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_PROCFSProvider

This provider is an instance provider for the IBMAIX_PROCFS class. It conforms to the CMPI interface to provide information about local PROCFS file system.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
Root	string
FilesystemSize	uint64
AvailableSpace	uint64
ReadOnly	Boolean
FilesystemType	string

For more information about the properties descriptions, see the IBMAIX_PROCFS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_ProcessorProvider

This provider is an instance provider for the IBMAIX_Processor class. It conforms to the CMPI interface to provide individual process properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
ElementName	string
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
Role	string
Family	uint16

Property	Type
OtherFamilyDescription	string
MaxClockSpeed	uint32 (Mhz)
CurrentClockSpeed	uint32 (Mhz)
LoadPercentage	uint16
Stepping	string
CPU Status	uint16

For more information about the properties descriptions, see the IBMAIX_UnixProcess class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_RunningOSProvider

This provider is an instance provider for the IBMAIX_RunningOS class. It conforms to the CMPI interface to provide the association between the computer system and the operating system.

When an instance is returned, the properties of the IBMAIX_ComputerSystem and subclasses of IBMAIX_OperatingSystem classes, listed in their respective providers, are returned.

For more information about the properties descriptions, see the IBMAIX_RunningOS class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_TokenRingPortProvider

This provider is an instance provider for the IBMAIX_TokenRingPort class. It conforms to the CMPI interface to provide token-ring port properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
Element Name	string
Name	string
Status	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
MaxSpeed	uint64 (bps)
Speed	uint64 (bps)
OtherNetworkPortType	string
LinkTechnology	uint16
OtherLinkTechnology	string

For more information about the properties descriptions, see the IBMAIX_TokenRingPort class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

OSBase_UnixProcessProvider

This provider is an instance provider for the IBMAIX_UnixProcess class. It conforms to the CMPI interface to provide individual process properties.

The following provider properties are implemented:

Property	Type
Caption	string
Description	string
Name	string
CSCreationClassName	string
CSName	string
CreationClassName	string
OSCreationClassName	string
OSName	string
Handle	string
CreationDate	datetime
KernelModeTime	uint64
UserModeTime	uint64
ParentProcessID	string
RealUserID	uint64
ProcessGroupID	uint64
ProcessTTY	string
ModulePath	string
Parameters[]	string

For more information about the properties descriptions, see the IBMAIX_UnixProcess class in the **/usr/pegasus/provider/mof/IBMAIX_OSBase.mof** file.

IBMPSG_BIOSProvider

This provider is an instance provider that writes to the CMPI interface and provides information about BIOS on the system.

The following provider properties are implemented:

Property	Type
SMBIOSBIOSVersion	string
Version	string
Manufacturer	string
PrimaryBIOS	boolean
Name	string

Property	Type
Version	string
TargetOperatingSystem	uint16
Description	string
ElementName	string
Caption	string
CreationClassName	string
SMBIOSPresent	boolean

For more information about the properties descriptions, see the IBMPSG_BIOS class in the **/usr/pegasus/provider/mof/IBMPSG_BIOS.mof** file.

IBMPSG_BaseboardProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the baseboard. This provider is used by the IBM® Director agent.

The following provider properties are implemented:

Property	Type
PoweredOn	boolean
Tag	string
CreationClassName	string
ElementName	string
Manufacturer	string
Model	string
SerialNumber	string
Name	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_BaseBoard class in the **/usr/pegasus/provider/mof/IBMPSG_BaseBoard.mof** file.

IBMPSG_CacheMemoryProvider

This provider is an instance provider that writes to the CMPI interface and provides information about cache memory on the system.

The following provider properties are implemented:

Property	Type
SystemCreationClassName	string
SystemName	string
CreationClassName	string
OtherEnabledState	string

Property	Type
RequestedState	uint16
EnabledDefault	uint16
OtherEnabledState	string
Status	string
Caption	string
Description	string
InstalledSize	uint32

For more information about the properties descriptions, see the IBMPSG_CacheMemory class in the **/usr/pegasus/provider/mof/IBMPSG_CacheMemory.mof** file.

IBMPSG_ComputerSystemProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the computer system.

The following provider properties are implemented:

Property	Type
Model	string
NameFormat	string
CreationClassName	string
Name	string
PrimaryOwnerName	string
PrimaryOwnerContact	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
Status	string
Caption	string
Description	string
ElementName	string
LPARID	string

For more information about the properties descriptions, see the IBMPSG_ComputerSystem class in the **/usr/pegasus/provider/mof/IBMPSG_ComputerSystem.mof** file.

IBMPSG_FRUProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the FRUs on the system.

The following provider properties are implemented:

Property	Type
FRUNumber	string
IdentifyingNumber	string
Vendor	string
Description	string
Name	string
Caption	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_FRU class in the **/usr/pegasus/provider/mof/IBMPSG_FRU.mof** file.

IBMPSG_LogicalDiskProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the logical disks on the system. This provider is used by the IBM Director agent.

The following provider properties are implemented:

Property	Type
DataOrganization	uint16
Purpose	string
Access	uint16
BlockSize	uint64
NumberOfBlocks	uint64
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
Availability	uint16
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
Name	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_LogicalDisk class in the **/usr/pegasus/provider/mof/IBMPSG_LogicalSystem.mof** file.

IBMPSG_OperatingSystemProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the operating system.

The following provider properties are implemented:

Property	Type
CodeSet	string
LanguageEdition	string
CSCreationClassName	string
CSName	string
CreationClassName	string
Name	string
OSType	uint16
OtherTypeDescription	string
Version	string
LastBootUpTime	datetime
LocalDateTime	datetime
CurrentTimeZone	sint16
NumberOfLicensedUsers	uint32
NumberOfUsers	uint32
NumberOfProcesses	uint32
TotalVirtualMemorySize	uint64
FreeVirtualMemory	uint64
FreePhysicalMemory	uint64
TotalVisibleMemorySize	uint64
SizeStoredInPagingFiles	uint64
FreeSpaceInPagingFiles	uint64
MaxProcessMemorySize	uint64
Distributed	boolean
MaxProcessesPerUser	uint32
EnabledState	uint16

For more information about the properties descriptions, see the IBMPSG_OperatingSystem class in the **/usr/pegasus/provider/mof/IBMPSG_OperatingSystem.mof** file.

IBMPSG_PhysicalDiskProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the physical disks on the system.

The following provider properties are implemented:

Property	Type
MaxMediaSize	uint64
DefaultBlockSize	uint64

Property	Type
UnitsDescription	string
MaxUnitsBeforeCleaning	uint64
UnitsUsed	uint64
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
Status	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_PhysicalDisk class in the **/usr/pegasus/provider/mof/IBMPSG_PhysicalDisk.mof** file.

IBMPSG_PhysicalMemoryProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the physical memory on the system. This provider is used by the IBM Director agent.

The following provider properties are implemented:

Property	Type
Capacity	uint64
DeviceLocator	string
PartNumber	string
SerialNumber	string
Capacity	uint64
Tag	string
CreationClassName	string
ElementName	string
SerialNumber	string
Version	string
PartNumber	string
Name	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_ComputerSystem class in the **/usr/pegasus/provider/mof/IBMPSG_PhysicalMemory.mof** file.

IBMPSG_ProcessorProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the processors on the system.

The following provider properties are implemented:

Property	**Type**
Role	string
Family	uint16
OtherFamilyDescription	string
MaxClockSpeed	uint32
CurrentClockSpeed	uint32
LoadPercentage	uint16
Stepping	string
CPUStatus	uint16
SystemCreationClassName	string
SystemName	string
CreationClassName	string
DeviceID	string
EnabledState	uint16
OtherEnabledState	string
RequestedState	uint16
EnabledDefault	uint16
Name	string
Status	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_Processor class in the **/usr/pegasus/provider/mof/IBMPSG_Processor.mof** file.

IBMPSG_SlotProvider

This provider is an instance provider that writes to the CMPI interface and provides information about the slots on the system.

The following provider properties are implemented:

Property	**Type**
ConnectorType[]	uint16
SupportsHotPlug	boolean
OtherTypeDescription	string
Tag	string

Property	Type
CreationClassName	string
ElementName	string
Caption	string
Description	string
ElementName	string

For more information about the properties descriptions, see the IBMPSG_Slot class in the **/usr/pegasus/provider/mof/IBMPSG_Slot.mof** file.

IBMPSG_IndicationProvider

This provider generates CIM indications of the class types and properties in the following list:

- **IBMPSG_FanEvent** indication is generated when a fan device error is detected by an AIX diagnostic event.
- **IBMPSG_VoltageEvent** indication is generated when a voltage alert is detected by an AIX diagnostic event (JS20 only).
- **IBMPSG_TemperatureEvent** indication is generated when a temperature alert is detected by an AIX diagnostic event (JS20 only).
- **IBMPSG_NetworkAdapterFailedEvent** indication is generated when a network adapter failed error is detected by an AIX diagnostic event.
- **IBMPSG_PowerSupplyEvent** indication is generated when a power supply error is detected by an AIX diagnostic event.
- **IBMPSG_ProcessorPFEvent** indication is generated when a processor error is detected by an AIX diagnostic event.
- **IBMPSG_MemoryPFEvent** indication is generated when a memory error is detected by an AIX diagnostic event.
- **IBM_SystemEvent** indication is generated for all other errors detected by an AIX diagnostic event. For this indication, the EventID property contains the resource name that caused the error.
- **IBMP_CSServiceEventIndication** is generated when the errors are serviced by a customer engineer. For this indication, the EventID property contains the resource name that was serviced.

The following provider properties are implemented:

Property	Type
Description	string
SystemCreationClassName	string
SystemName	string
IndicationIdentifier	string
IndicationTime	dateTime
AlertingManagedElement	string
EventID	string
AlertType	uint16
PerceivedSeverity	uint16
UniqueSystemID	string
Trending	uint16
OtherAlertType	string

Property	Type
ProbableCause	uint16
ProbableCauseDescription	string
RecommendedActions	string

For more information about the properties descriptions, see the **/usr/pegasus/provider/mof/IBMPSG_AlertIndication.mof** file.

Configure AIX Provider Tracing

Tracing should only be used for debugging purposes and is, by default, disabled. You can, however, enable the tracing utility for providers by setting the following environment variables before starting the CIM Server:

Note: You must use the appropriate shell command to set these variables. For example, for a Bourne shell or Korn shell, use the **export** command, and for the C shell, use the **setenv** command.

- **AIX_PROVIDER_TRACE_FILE**=*filepath*, where *filepath* is the file where you want trace data saved.
- **AIX_PROVIDER_TRACE_LEVEL**=*level*, where *level* is one of the trace levels.

Choose from one of the following trace levels:

level 1
Function entry and exit

level 2
Basic flow trace messages, low data detail

level 3
Interfunction logic flow, medium data detail

level 4
All information, high data detail

For example, to set the trace level to trace all information with high data detail using a Korn shell, type the following commands:

```
export AIX_PROVIDER_TRACE_FILE=/tmp/FILE_NAME
export AIX_PROVIDER_TRACE_LEVEL=4
```

Appendix A. cimauth Command

Purpose

Adds, modifies, removes or lists CIM user authorizations.

Syntax

cimauth -a -u *username* **-n** *namespace* [**-R**] [**-W**]

cimauth -m -u *username* **-n** *namespace* [**-R**] [**-W**]

cimauth -r -u *username* [**-n** *namespace*]

cimauth -l

Description

The **cimauth** command provides a command line interface to manage CIM user authorizations on a namespace.

This command does not configure or list CIM user password information. For more information on managing users, see the cimuser command.

The **cimauth** command can be used to remove authorizations of one user on one namespace or all the namespaces on which the user has authorizations. If no namespace is specified, then authorizations on all the namespaces for the specified user will be removed.

Specifying no options with the **cimauth** command will show the usage of the command.

Flags

- a
Adds authorizations for a user on a namespace.

- l
Displays the authorizations of all CIM users.

- m
Modifies the authorizations for a user on a namespace.

- n *namespace*
Specifies the namespace for which you are adding authorizations to users.

A namespace is a logical unit for grouping classes and instances to control their scope and visibility. These namespaces are not physical locations, but are similar to logical databases containing specific classes and instances.

- r
Removes the authorizations for a user on a namespace.

- R
Grants read authorization.

- u *username*
Specifies the user name for which you are adding, deleting, or modifying authorizations.

- W
Grants write authorization.

Return Values

0
The command completed successfully.

1
The command did not complete successfully. An explanatory error message is written to stderr.

Examples

1. To add read-write authorization to the user named **guest** on the **root/system** namespace, type the following:

   ```
   cimauth -a -u guest -n root/system -R -W
   ```
2. To add read authorization to the user named **guest** on the **root/cimv2** namespace, type the following:

   ```
   cimauth -a -u guest -n root/cimv2
   ```
3. To add read access only to the user named **guest** on the **root/system** namespace, type the following:

   ```
   cimauth -m -u guest -n root/system -R
   ```
4. To remove all access granted to the user named **guest** on the **root/system** namespace, type the following:

   ```
   cimauth -r -u guest -n root/system
   ```
5. To display the list of authorized user names, namespaces and authorizations, type the following:

   ```
   cimauth -l
   ```

Related Information

The cimuser command.

Appendix B. cimconfig Command

Purpose

Obtains, sets, unsets, or lists CIM Server configuration properties.

Syntax

cimconfig -g *name* [**-c**] [**-p**] [**-d**]

cimconfig -s *name=value* [**-c**] [**-p**]

cimconfig -u *name* [**-c**] [**-p**]

cimconfig -l [**-c** | **-p**]

Description

The **cimconfig** command provides a command line interface to manage CIM Server configuration properties. An error message is returned if the CIM Server is not running. Specifying no flags with the **cimconfig** command displays the command's usage statement.

Flags

-c
Indicates that the action taken applies to the current configuration property value. This flag must be used in conjunction with one of the **-g**, **-s**, **-u**, or **-l** flags.

-d
Indicates that the action taken applies to the default configuration property value. This flag must be used in conjunction with the **-g** flag.

-g *name*
Displays the current, planned, or default value of the specified configuration property depending on whether the **-c**, **-p**, or **-d** flag is specified. If no other flag is specified, the current value of the specified configuration property is displayed.

-l
Lists the names of all the configuration properties when used without any other flag. When this flag is paired with the **-c** flag, a paired listing of all the current configuration properties and their values is displayed. When this flag is paired with the **-p** flag, a paired listing of all the planned configuration properties and their values is displayed.

-p
Indicates that the action taken applies to the planned configuration property value. This flag must be used in conjunction with one of the **-g**, **-s**, **-u**, or **-l** flags.

-s *name=value*
Sets the current or planned value of the specified configuration property to the specified value, depending on whether the **-c** or **-p** flag is specified. If no other flag is specified, the current value of the specified configuration property is set to the specified value. The *name* parameter is the name of the configuration property, and *value* is the new value for the configuration property. An error message is returned if the specified property cannot be set dynamically.

-u *name*
Resets the current or planned value of the specified configuration property to the default value depending on whether the **-c** or **-p** flag is specified. If no other flag is specified, the current value of the specified configuration property is reset to the default value. An error message is returned if the specified property cannot be updated dynamically.

Exit Status

0
The command completed successfully.

1
The command did not complete successfully, and an explanatory message is written to stderr.

Examples

1. To view the current value for the **port** configuration property, type one of the following commands:
   ```
   cimconfig -g port
   cimconfig -g port -c
   ```
2. To view the planned value of the **traceLevel** configuration property, type the following:
   ```
   cimconfig -g traceLevel -p
   ```
3. To change the current value of the **traceLevel** configuration property to the value `2`, type one of the following commands:
   ```
   cimconfig -s traceLevel=2
   cimconfig -s traceLevel=2 -c
   ```
4. To change the planned value of the **traceLevel** property to a value of `3`, type the following:
   ```
   cimconfig -s traceLevel=3 -p
   ```
5. To reset the current value of the **traceLevel** property to the default value, type one of the following commands:
   ```
   cimconfig -u traceLevel
   cimconfig -u traceLevel -c
   ```
6. To reset the planned value of the **traceLevel** property to the default value, type the following:
   ```
   cimconfig -u traceLevel -p
   ```
7. To list all of the current configuration properties and their values, type the following:
   ```
   cimconfig -l -c
   ```
8. To list all of the planned configuration properties and their values, type the following:
   ```
   cimconfig -l -p
   ```

Appendix C. cimmof Command

Purpose

Compiles MOF files into the CIM repository through the CIM Server.

Syntax

cimmof -h

cimmof [**-w**] [**-I** *path*] [**-n** *namespace*] *file*...*file*

Description

The **cimmof** command is the command line interface to the Managed Object Format (MOF) Compiler. The MOF Compiler is a utility that compiles MOF files (using the MOF format defined by the DMTF CIM Specification) into CIM classes and instances that are stored in the CIM Repository.

Notes:

1. A superuser or user with write access to the default or specified namespace is the only user who can run the **cimmof** command to compile MOFs in the CIM Repository.
2. Superclasses must be compiled before subclasses. Otherwise, the compile will fail.
3. It is strongly recommended that MOF files include all necessary subclasses, so they can compile properly even if certain classes are not in the CIM Repository.

The **cimmof** command can be used to compile MOF files at any time after installation. If no input file is specified, stdin is used as the input.

The MOF Compiler requires that the input MOF files be in the current directory or that a fully qualified path be given. To simplify the specification of multiple MOF files in the **cimmof** command line, the MOF Compiler allows compiling from files containing a list of MOF files using the **include** pragma, similar to the following:

- #pragma include (″application.mof″)
- #pragma include (″server.mof″)

MOF files using the include pragma must be in the current directory or in a directory specified by the **-I** flag.

The **-n** *namespace* flag can be used to specify a namespace in which the CIM classes and instances will be compiled. A namespace is a logical unit for grouping classes and instances to control their scope and visibility. These namespaces are not physical locations, but are similar to logical databases containing specific classes and instances. If this flag is not specified, the default namespace is **root/cimv2**, except for the provider registration schemas.

For provider registration schemas, if the **-n** flag is not specified, the default namespace is **root/PG_InterOp**. If the **-n** flag is specified, the namespace specified must be **root/PG_InterOp**, otherwise, the following error message is returned:

```
The requested operation is not supported.
```

For provider MOF files, the namespace specified must match one of the namespaces specified in the **PG_ProviderCapabilities** class schema definition.

Flags

-h
Displays the command usage and the version number of the MOF compiler.

-I *path*
Specifies the path to the included MOF files. This path can be relative or absolute.

If the input MOF file has #pragma includes, and the included files do not reside in the current directory, this option must be used to specify a path to them.

-n *namespace*
Specifies the namespace and overrides the default CIM Repository namespace. Any namespace specified here must be a valid CIM namespace.

For provider registration schemas, specify the **root/PG_InterOp** namespace.

-w
Suppresses warning messages. When MOF files are compiling, if there are CIM elements such as classes or instances defined in the MOF files that already exist in the CIM Repository, the **cimmof** command returns warning messages. Use this flag to suppress those warning messages.

Exit Status

0
The command completed successfully.

1
The command did not complete successfully, and an error message is written to stderr.

Standard Error

Error	**Description**
Error trying to create Repository in path localhost:5988: Cannot connect to: localhost:5988 Failed to set DefaultNamespacePath.	The CIM Server is not running. Start the CIM Server and run the **cimmof** command again.
Line number	An error was found at the line number line in the MOF file that the MOF compiler is parsing.
Operation cannot be carried out since the specified superclass does not exist.	The MOF compiler compiled a MOF file with one or more superclasses that are not in the CIM Repository.

Examples

1. To compile an MOF file named **processinfo.mof** into the default namespace in the CIM Repository, type the following:

   ```
   cimmof processinfo.mof
   ```

2. To compile the MOF file name **processinfo.mof** into the **root/application** namespace, type the following:

   ```
   cimmof -n root/application processinfo.mof
   ```

3. To compile the **CIMSchema25.mof** file defined in the **./MOF** directory that also contains #pragma includes for other MOF files in the **./MOF** directory, type the following:

   ```
   cimmof -w -I ./MOF MOF/CIMSchema25.mof
   ```

4. To list the arguments to the **cimmof** command and display the version of the MOF Compiler, type the following:

   ```
   cimmof -h
   ```

Related Information

The cimserver command.

Appendix D. cimmofl Command

Purpose

Compiles MOF files into the CIM repository. Run this command while the CIM Server is not running.

Syntax

cimmofl -h

cimmofl [**-w**] [**-I** *path*] [**-n** *namespace*] *file*...*file*

Description

The **cimmofl** command is the command line interface to the Managed Object Format (MOF) Compiler. The MOF Compiler is a utility that compiles MOF files (using the MOF format defined by the DMTF CIM Specification) into CIM classes and instances that are stored in the CIM Repository. To see changes that are made to the repository, you must restart the CIM server.

Notes:

1. A superuser or user with write access to the default or specified namespace is the only user who can run the **cimmofl** command to compile MOFs in the CIM Repository.
2. Superclasses must be compiled before subclasses. Otherwise, the compile will fail.
3. It is strongly recommended that MOF files include all necessary subclasses, so they can compile properly even if certain classes are not in the CIM Repository.

The **cimmofl** command can be used to compile MOF files at any time after installation. If no input file is specified, stdin is used as the input.

The MOF Compiler requires that the input MOF files be in the current directory or that a fully qualified path be given. To simplify the specification of multiple MOF files in the **cimmofl** command line, the MOF Compiler allows compiling from files containing a list of MOF files using the **include** pragma, similar to the following:

- #pragma include (″application.mof″)
- #pragma include (″server.mof″)

MOF files using the include pragma must be in the current directory or in a directory specified by the **-I** flag.

The **-n** flag can be used to specify a namespace in which the CIM classes and instances will be compiled. A namespace is a logical unit for grouping classes and instances to control their scope and visibility. These namespaces are not physical locations, but are similar to logical databases containing specific classes and instances. If this flag is not specified, the default namespace is **root/cimv2**, except for the provider registration schemas.

For provider registration schemas, if the **-n** flag is not specified, the default namespace is **root/PG_InterOp**. If the **-n** flag is specified, the namespace specified must be **root/PG_InterOp**, otherwise, the following error message is returned:

```
The requrested operation is not supported.
```

For provider MOF files, the namespace specified must match one of the namespaces specified in the **PG_ProviderCapabilities** class schema definition.

Flags

-h
Displays the command usage and the version number of the MOF compiler.

-I *path*
Specifies the path to the included MOF files. This path can be relative or absolute.

If the input MOF file has #pragma includes, and the included files do not reside in the current directory, this option must be used to specify a path to them.

-n *namespace*
Specifies the namespace and overrides the default CIM Repository namespace. Any namespace specified here must be a valid CIM namespace.

For provider registration schemas, specify the **root/PG_InterOp** namespace.

-w
Suppresses warning messages. When MOF files are compiling, if there are CIM elements such as classes or instances defined in the MOF files that already exist in the CIM Repository, the **cimmofl** command returns warning messages. Use this flag to suppress those warning messages.

Exit Status

0
The command completed successfully.

1
The command did not complete successfully, and an error message is written to stderr.

Standard Error

Error	**Description**
Error trying to create Repository in path localhost:5988 Cannot connect to: localhost:5988 Failed to set DefaultNamespacePath.	The CIM Server is not running. Start the CIM Server and run the **cimmofl** command again.
Line number	An error was found at the *line number* line in the MOF file that the MOF compiler is parsing.
Operation cannot be carried out since the specified superclass does not exist.	The MOF compiler compiled an MOF file with one or more superclasses that are not in the CIM Repository.

Examples

1. To compile an MOF file named **processinfo.mof** into the default namespace in the CIM Repository, type the following:

   ```
   cimmofl processinfo.mof
   ```

2. To compile the **processinfo.mof** file into the **root/application** namespace, type the following:

   ```
   cimmofl -n root/application processinfo.mof
   ```

3. To compile the **CIMSchema25.mof** file defined in the **./MOF** directory that also contains #pragma includes for other MOF files in the **./MOF** directory, type the following:

   ```
   cimmofl -w -I ./MOF MOF/CIMSchema25.mof
   ```

4. To list the arguments to the **cimmofl** command and display the version of the MOF Compiler, type the following:

```
   cimmofl -h
```

Related Information

The cimserver command.

Appendix E. cimprovider Command

Purpose

Disables, enables, removes and lists registered CIM providers or CIM provider modules and module status

Syntax

cimprovider -d -m *module*

cimprovider -e -m *module*

cimprovider -r -m *module* [**-p** *provider*]

cimprovider -l [**-s** | **-m** *module*]

Description

The **cimprovider** command provides a command line interface to disable, enable, unregister, and list registered CIM providers. If a CIM provider is disabled, the CIM Server rejects any requests to the provider. If a CIM provider is enabled, the CIM Server forwards requests to the provider. If a CIM provider is unregistered, the CIM server will no longer have information about the provider.

In order to use the **cimprovider** command, the CIM Server has to be running and the specified provider or provider module, which is a grouping of providers in the same shared library, must be registered with Web-based Enterprise Management (WBEM) Services.

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

Specifying no options with the **cimprovider** command displays the command usage.

Note: The **-l** flag is available to any user, however, all other flags are available only to superusers.

Flags

-d
Disables the specified CIM provider module, placing all of its contained providers in the Stopped state. When a specified provider module is in the disabled state, any new requests to its contained providers are rejected.

If a user attempts to disable a module that is already disabled, an error message is returned and no action is taken.

-e
Enables the specified CIM provider module, placing all of its contained providers in the OK state. Providers contained in the enabled provider module are available to accept new requests.

If a user attempts to enable a module that is already enabled or attempts to enable a module that is disabling, an error message is returned and no action is taken.

-l
Displays all the registered provider modules when it is not used with the **-m** *module* option.

Displays all the providers in the specified provider module when used with the **-m** *module* option.

-m *module*
Specifies the provider module for the operation.

-p *provider*
Specifies the provider for the operation.

-r
Removes, or un-registers, the specified provider module and all of its contained providers. If a provider is specified, only that provider is removed. Other providers in the same provider module are not affected.

To re-register a provider or provider module, the registration schema for that provider or provider module must be reloaded using the **cimmof** command.

-s
Displays the status of provider modules.

Exit Status

0
The command completed successfully.

1
The command did not complete successfully, and an explanatory message is written to stderr.

Examples

1. To disable the **OperatingSystemProvider** provider module and all of its contained providers, type the following:

   ```
   cimprovider -d -m OperatingSystemProvider
   ```
2. To enable the **OperatingSystemProvider** provider module and all of its contained providers, type the following:

   ```
   cimprovider -e -m OperatingSystemProvider
   ```
3. To remove the **OperatingSystemProvider** provider module and all of its contained providers, type the following:

   ```
   cimprovider -r -m OperatingSystemProvider
   ```
4. To remove the **PG_OperatingSystemProvider** provider that is contained within the **OperatingSystemProvider** provider module, type the following:

   ```
   cimprovider -r -m OperatingSystemProvider -p PG_OperatingSystemProvider
   ```
5. To list the registered provider modules, type the following:

   ```
   cimprovider -l
   ```
6. To list the registered provider modules and their status, type the following:

   ```
   cimprovider -l -s
   ```
7. To list the registered providers contained within the **OperatingSystemProvider** provider module, type the following:

   ```
   cimprovider -l -m OperatingSystemProvider
   ```

Related Information

The cimmof and cimserver commands.

Appendix F. cimserver Command

Purpose

Starts and stops the CIM Server.

Syntax

cimserver [[**-v** | **-h** | **-s**] | [*configProperty=value* . . .]]

Description

The **cimserver** command allows you to start and stop the CIM Server.

Flags

-h Prints the usage statement for this command.

-s Stops the CIM Server.

-v Displays the CIM Server version number.

Parameters

configProperty=value Starts the CIM Server with the *configProperty* set to *value*. Separate multiple *configProperty=value* pairs with a blank space. This command line option overrides existing or default values in the **cimserver_current.conf** and **cimserver_planned.conf** configuration files.

Exit Status

0 The command completed successfully.

>0 The command did not complete successfully, and an explanatory message is written to stderr.

Examples

To start the CIM Server with a trace of level 4 on all components, type the following:

```
cimserver traceLevel=4 traceComponent=ALL
```

Related Information

The cimconfig command.

Appendix G. cimuser Command

Purpose

Adds, modifies, removes or lists authorized users of the Pegasus CIM Server.

Syntax

cimuser -a -u *username* [**-w** *password*]

cimuser -m -u *username* [**-w** *password*] [**-n** *newpassword*]

cimuser -r -u *username*

cimuser -l

Description

The **cimuser** command allows you to manage CIM users by adding, deleting, listing, and modifying the passwords of users you authorize to use the Pegasus CIM Server.

The default location of the CIM password file is **/opt/freeware/cimom/pegasus/etc/cimserver.passwd**, but can be changed by setting the passwordFilePath property using the **cimconfig** command. For more information about **cimconfig**, see Appendix B, “cimconfig Command,” on page 35.

If you are completing a task that requires a password and you do not specify the password with the **-w** *password* flag, you will be prompted to enter the password. This password must be no more than 8 characters. If more than 8 characters are entered, only the first 8 characters will be recognized as the password.

When adding a new CIM user, the CIM user must be a valid user on the local system. There is no default authorization permissions set for newly added users. For more information on adding authorization information, see Appendix A, “cimauth Command,” on page 33.

Specifying no options with the **cimuser** command will display the usage statement.

Flags

- a	Adds a CIM user.
- l	Lists all current CIM users.
- m	Modifies the password for the user specified in the **- u** *username* option.
- n *newpassword*	Identifies the new password for the user specified in the **- u** *username* option.
- r	Removes a CIM user. Any authorizations for the specified user will be deleted.
- u *username*	Identifies the user you want to add, remove, or for whom you want to modify the password.
- w *password*	Identifies the password for the user specified in the **- u** *username* option.

Return Values

The **cimuser** command returns one of the following values:

0 The command completed successfully.

1 An error occurred. An explanatory message is written to stderr.

Examples

1. To add a new user named **guest** with a password of **guest**, type the following:

   ```
   cimuser -a -u guest -w guest
   ```

2. To add a new user named **guest** without specifying the password in the command, type the following:

   ```
   cimuser -a -u guest
   Please enter your password:
   Please re-enter your password:
   ```

   Because you did not specify the password in the initial command string with the **- w** *password* option, you are prompted to enter and re-enter the password.

3. To change the password for user **guest** to **bar**, type the following:

   ```
   cimuser -m -u guest -w guest -n bar
   ```

4. To remove the user named **guest**, type the following:

   ```
   cimuser -r -u guest
   ```

Related Information

The cimauth command, the cimconfig command.

Index

Vos remarques sur ce document / Technical publication remark form

Titre / Title : Bull Common Information Model Guide

Nº Reférence / Reference Nº : 86 A2 50EM 01

Daté / Dated : October 2005

ERREURS DETECTEES / ERRORS IN PUBLICATION

AMELIORATIONS SUGGEREES / SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Vos remarques et suggestions seront examinées attentivement.
Si vous désirez une réponse écrite, veuillez indiquer ci-après votre adresse postale complète.

Your comments will be promptly investigated by qualified technical personnel and action will be taken as required.
If you require a written reply, please furnish your complete mailing address below.

NOM / NAME : ______________________________ Date : __________

SOCIETE / COMPANY : ______________________________

ADRESSE / ADDRESS : ______________________________

____

Remettez cet imprimé à un responsable BULL ou envoyez-le directement à :

Please give this technical publication remark form to your BULL representative or mail to:

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

Technical Publications Ordering Form

Bon de Commande de Documents Techniques

To order additional publications, please fill up a copy of this form and send it via mail to:
Pour commander des documents techniques, remplissez une copie de ce formulaire et envoyez-la à :

BULL CEDOC
ATTN / Mr. L. CHERUBIN
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

Phone / Téléphone :	+33 (0) 2 41 73 63 96
FAX / Télécopie	+33 (0) 2 41 73 60 19
E–Mail / Courrier Electronique :	srv.Cedoc@franp.bull.fr

Or visit our web sites at: / Ou visitez nos sites web à:

http://www.logistics.bull.net/cedoc

http://www–frec.bull.com   http://www.bull.com

CEDOC Reference # Nº Référence CEDOC	**Qty** Qté	**CEDOC Reference #** Nº Référence CEDOC	**Qty** Qté	**CEDOC Reference #** Nº Référence CEDOC	**Qty** Qté
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	
__ __ ____ _ [__]		__ __ ____ _ [__]		__ __ ____ _ [__]	

[_ _] : **no revision number means latest revision** / pas de numéro de révision signifie révision la plus récente

NOM / NAME : Date :

SOCIETE / COMPANY :

ADRESSE / ADDRESS :

PHONE / TELEPHONE : FAX :

E–MAIL :

For Bull Subsidiaries / Pour les Filiales Bull :

Identification:

For Bull Affiliated Customers / Pour les Clients Affiliés Bull :

Customer Code / Code Client :

For Bull Internal Customers / Pour les Clients Internes Bull :

Budgetary Section / Section Budgétaire :

For Others / Pour les Autres :

Please ask your Bull representative. / Merci de demander à votre contact Bull.

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

ORDER REFERENCE
86 A2 50EM 01