Operations Agent

Software Version: 12.05

Windows®, HP-UX, Linux, Solaris and AIX operating systems

Reference Guide

Document Release Date: February 2018
Software Release Date: December 2017

Hewlett Packard
Enterprise

Legal Notices

Warranty

Restricted Rights Legend

Copyright Notice

Trademark Notices

Acknowledgements

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes software written by Tim Hudson (tjh@cryptsoft.com).

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright ©1995-2002 Jean-loup Gailly and Mark Adler.

Documentation Updates

To check for recent updates or to verify that you are using the most recent edition of a document, go to: https://softwaresupport.hpe.com/.

This site requires that you register for an HPE Passport and to sign in. To register for an HPE Passport ID, click **Register** on the HPE Software Support site or click **Create an Account** on the HPE Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HPE sales representative for details.

Support

Visit the HPE Software Support site at: https://softwaresupport.hpe.com/.

Most of the support areas require that you register as an HPE Passport user and to sign in. Many also require a support contract. To register for an HPE Passport ID, click **Register** on the HPE Support site or click **Create an Account** on the HPE Passport login page.

To find more information about access levels, go to: https://softwaresupport.hpe.com/web/softwaresupport/access-levels.

HPE Software Solutions Now accesses the HPE Software Solution and Integration Portal website. This site enables you to explore HPE Product Solutions to meet your business needs, includes a full list of Integrations between HPE Products, as well as a listing of ITIL Processes. The URL for this website is https://softwaresupport.hpe.com/km/KM01702731.

Contents

Chapter 1: Introduction

The Operations Agent introduces many services, processes, and utilities on the system. Command-line utilities help you configure the operation and monitor the performance of the agent. Using certain command-line utilities, you can view the real-time system performance data captured by the agent. Utilities like the tracing tools help you view the diagnostic information of the agent for troubleshooting.

The Operations Agent offers you a series of configuration variables; these variables help you control the behavior of the agent. You can use the `ovconfchg` command to assign desired values to these variables.

This guide contains information on the command-line utilities, services, and processes introduced on the system by the Operations Agent. The guide also provides you with a list of configuration variables that you can use while configuring the default behavior of the Operations Agent.

Convention Used in this Document

The following convention is used in this document.

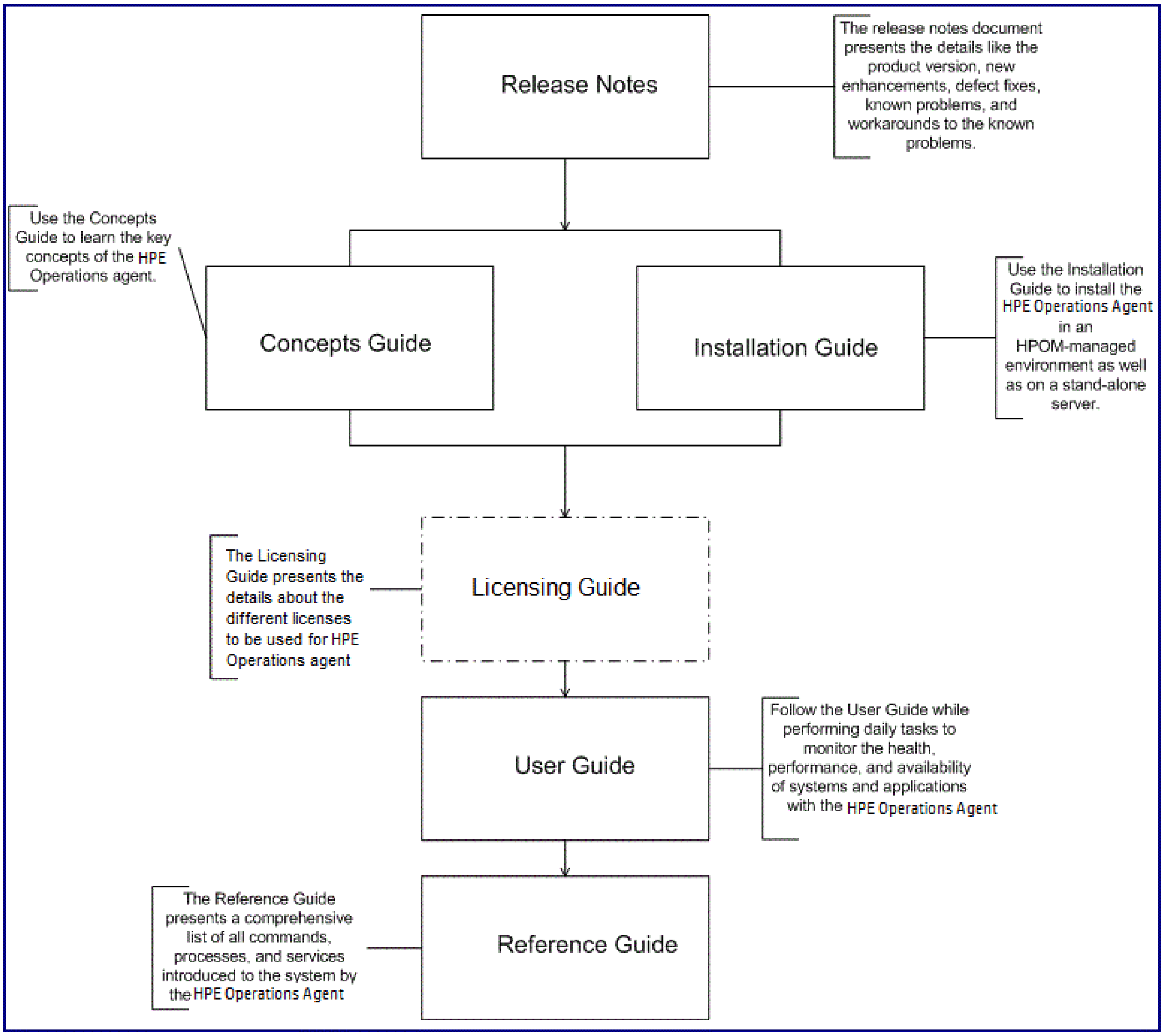
Convention	Description

`<OvDataDir>`
Or
`$OvDataDir`
	<OvDataDir> or **$OvDataDir** is used in this document to imply the following location:
	• **On Windows:**
	`%OvDataDir%`
	• **On Unix/Linux:**
	`/var/Opt/OV`

Documentation Map

The documentation map presents a list of all the major documents for the Operations Agent. You can use the map to identify the necessary document when you need assistance.

Documentation Map for the Operations Agent


Release Notes
The release notes document presents the details like the product version, new enhancements, defect fixes, known problems, and workarounds to the known problems.
Use the Concepts Guide to learn the key concepts of the HPE Operations agent.
Concepts Guide
Installation Guide
Use the Installation Guide to install the HPE Operations Agent in an HPOM-managed environment as well as on a stand-alone server.
The Licensing Guide presents the details about the different licenses to be used for HPE Operations agent
Licensing Guide
User Guide
Follow the User Guide while performing daily tasks to monitor the health, performance, and availability of systems and applications with the HPE Operations Agent
The Reference Guide presents a comprehensive list of all commands, processes, and services introduced to the system by the HPE Operations Agent
Reference Guide

Chapter 2: Components of the Operations Agent

The Operations Agent consists of two major operational components: Operations Monitoring Component and Performance Collection Component. The Operations Monitoring Component builds up the monitoring and messaging capabilities of the agent and the Performance Collection Component provides the data collection and storage functionality.

Table 1 Operations Agent Components

Component	Sub-components	Process Name
Operations Monitoring Component	Monitor Agent	opcmona
	Action Agent	opcacta
	Message Agent	opcmsga
	Message Interceptor	opcmsgi
	SNMP Trap Interceptor	opctrapi
	WMI Interceptor	opcwbemi
	Logfile Encapsulator	opcle
	Event Correlation Agent	opceca
Performance Collection Component	oacore collector	oacore
	Measurement Interface Daemon	midaemon
	Transaction Tracking Daemon	ttd
Real-Time Metric Access (RTMA)	Multi-platform system performance metric server	perfd
Real-Time Measurement (RTM)	real-time measurement	hpsensor

Processes

The Operations Agent starts different processes on the managed node. The below table lists all the processes (daemons on UNIX and Linux nodes) contributed by the Operations Monitoring Component.

Table 2 Operations Monitoring Component Processes

Process	Description
opcacta	The action agent is responsible for starting and stopping automatic actions, operator-initiated actions, and scheduled actions (that is, scripts and programs). The action agent is also used for command broadcasting and for configured applications (Input/Output).
opceca	The event correlation agent connects to the agent MSI (Message Stream Interface) in the same way that the ECS (Event Correlation Services) runtime library is integrated into the OM server. This connection allows access to and modification of messages from the OM message flow on the agent. The messages modified by this process display in the Message Details window (available from the Message Browser) with the message source “MSI: opceca”. Like all agent processes, this process is controlled by the control agent.
opcle	The logfile encapsulator scans one or more application or system-logfiles (including the Windows Eventlog) for messages or patterns specified by the OM administrator. The logfile encapsulator forwards the scanned and filtered messages to the message agent.
opcmona	The monitor agent monitors the following: • System parameters (for example, CPU load, disk utilization, kernel parameters) • SNMP MIBs • WMI classes • Performance counters from Windows performance monitors • Scheduled policies • Other parameters, if specified
opcmsga	The message agent receives messages from the logfile encapsulator, monitor agent, event interceptor, and message interceptor on the local system. The messages are forwarded to the message receiver running on the management server. If the connection to the management server is lost, the messages are buffered locally. If the primary server is down, the message agent sends the messages to the backup server. The message agent triggers local automatic actions by forwarding the task to the action agent.
opcmsgi	The message interceptor receives and processes incoming messages. The `opcmsg` command and `opcmsg` API can be used to forward messages to OM. Conditions can be set up to integrate or suppress chosen message types.

Table 2 Operations Monitoring Component Processes, continued

Process	Description
opctrapi	The SNMP trap interceptor is the message interface for feeding SNMP events to OM. Conditions can be set to integrate or suppress selected message types.
opcwbemi	The WMI interceptor enables you to process WMI classes originating from systems that are not managed nodes.

Table 3 lists all the processes (daemons on UNIX and Linux nodes) contributed by the Performance Collection Component.

Table 3 Performance Collection Component Processes

Process	Description
oacore	**oacore** process is invoked by the `ovpa` script. The **oacore** process continuously collects performance and health data across your system and stores the collected data in the Metrics Datastore. The user-controlled configuration file—the **parm** file—is used to control logging of data by oacore.
midaemon	The Measurement Interface daemon, `midaemon`, provides an interface between ARM transaction tracking and the Performance Collection Component. This process translates trace data into Measurement Interface counter data using a memory-based MI Performance Database to hold the counters. The database is accessed by collector programs such as glance, xglance, and oacore.

ttd	The transaction tracking daemon, `ttd`, reads and registers transaction definitions from the following configuration file:
	On UNIX:
	`/var/opt/perf/ttd.conf`
	On Windows:
	`%ovdatadir%\ttd.conf`
	The **ttd** process also assigns IDs to the transaction names passed to it through the `arm_getid` calls that come from the ARM library. The **ttd** process synchronizes these transaction definitions with the **midaemon** process. **ttd** runs in the background mode by default.

perfalarm	The alarm generator server, `perfalarm`, helps in scanning the information in the `alarmdef` file and sending alerts to the destinations based on the configuration information in the `alarmdef` file. For more information, see "*Performance Alarms*" in *Operations Agent Use Section*.
hpsensor	**hpsensor** process helps in accessing real time performance metrics, through a secure communication channel, locally or remotely.
perfd	The **perfd** process, provided by the RTMA component, helps you access real-time system performance metrics, locally or remotely.

Chapter 3: Using the Command-Line Utilities

Your use of the *Using the Command-Line Utilities* section is subject to the license terms for the Operations Agent software.

The Operations Agent introduces several command-line utilities to the node with which you can perform different configuration tasks. These utilities are present in the following directories:

On Windows:

`%ovinstalldir%\bin`

On HP-UX/Linux/Solaris:

`/opt/OV/bin` and `/opt/perf/bin`

On AIX:

`/usr/lpp/OV/bin` and `/usr/lpp/perf/bin`

These utilities are primarily introduced by different operational components of the Operations Agent.

Utilities Provided by the Operations Monitoring Component

This section provides information on command-line utilities presented by the Operations Monitoring Component of the Operations Agent.

ovbbccb

NAME

`ovbbccb` – Controls HTTPS communication using Communication Broker proxies on local nodes.

SYNOPSIS

```
ovbbccb -h|-help
ovbbccb -version

ovbbccb -install|-remove [-v|-verbose]

ovbbccb -daemon|-nodaemon [-debug] [-v|-verbose]

ovbbccb -start|-stop <ovrg> [<hostname>|<ip>] [-v|-verbose]

ovbbccb -kill|-reinit [<hostname>|<ip>] [-v|-verbose]

ovbbccb -listovrg [<hostname>|<ip>] [-v|-verbose]

ovbbccb -ping {[<hostname>|<ip>[:<port>]] | [<uri>} [-v|-verbose]]

ovbbccb -status {[<hostname>|<ip>[:<port>]] | [<uri>} [-v|-verbose]]

ovbbccb -retryfailedrcp -ovrg [<resource_group>]
```

DESCRIPTION

ovbbccb command is used to control HTTPS communication using Communication Broker proxies on local nodes. It controls starting of the Communication Broker as a background daemon process or in normal mode, stopping, and re-initializing of the Communication Broker. ovbbccb is also used to start and stop resource groups in the Communication Broker.

ovbbccb can also be used to list all active resource groups and all applications registered to a Communication Broker, to check whether specified communication services are alive and to display details about the current state of the server.

> **Note:** On Windows, after agent installation, the firewall configuration changes when **HTTP Communication Broker** is added to the firewall inbound rules.

Parameters

The ovbbccb command incorporates the options in the following list. The syntax for the [<hostname>|<ip>][:<port>]] string, for example; in the options -registrations or -ping, can be a hostname and a port separated by a colon (:) but can also be a full URL path including protocol. for example:

```
https://merlin.guilford.mycom.com:383/Hewlett-Packard/OpenView/Coda/
```

ovbbccb recognizes the following options:

`-h|-help`

Displays and describes the available options for the ovbbccb command.

`-version`

Displays the version of the communication component in use.

`-install`

Installs the Communications Broker program as a service on a Microsoft Windows machine.

`-remove`

Removes the Communications Broker program from the services on a Microsoft Windows machine.

`-daemon`

Starts the Communication Broker either as a background daemon process on a UNIX machine or a service on a Microsoft Windows machine.

`-nodaemon`

Starts the Communication Broker as a foreground process (*default*).

`-debug`

Disable Control-C signal handler for debugging.

`-verbose`

Shows more detailed output.

`-start <ovrg> [<hostname>|<ip>]`

Starts the resource group specified by *<ovrg>* in the Communication Broker on the host specified by *<hostname>* or *<ip>*. If the hostname or IP is not specified, ovbbccb uses the local host as the host. You must configure the resource group on a cluster node to use this option.

`-stop <ovrg> [<hostname>|<ip>]`

Stops the resource group specified by *<ovrg>* in the Communication Broker on the host specified by *<hostname>* or *<ip>*. If the hostname or IP is not specified, ovbbccb uses the local host as the host. You must configure the resource group on a cluster node to use this option.

`-kill [<hostname>|<ip>]`

Stops the Communication Broker on the host specified by *<hostname>* or *<ip>*. If the hostname or IP is not specified, ovbbccb uses the local host as the host. You must set the LOCAL_CONTROL_ONLY parameter to false to make this option work on a remote node.

`-reinit [<hostname>|<ip>]`

The Communication Broker specified in *<hostname>* or *<ip>* reloads the configuration data and is re-initialized. If the hostname or IP is not specified, ovbbccb uses the local host as the host.

The `SIGHUP` signal may also be used on UNIX systems to re-initialize the Communication Broker process.

You must set the LOCAL_CONTROL_ONLY parameter to false to make this option work on a remote node.

`-listovrg [<hostname>|<ip>]`

Displays a list of all active resource groups for the Communication Broker on the node specified by *<hostname>* or *<ip>*. If the hostname or IP is not specified, ovbbccb uses the local host as the host. You must set the LOCAL_CONTROL_ONLY parameter to false to make this option work on a remote node.

`-ping {[<hostname>|<ip>[:<port>]] | [<uri>]}`

Pings the specified server process. A hostname or IP address with an optional port number or a URI may be given to locate the server process to ping. If a URI is given with the path of a valid process registered with the Communication Broker, the Communication Broker will automatically forward the ping to the registered process. The node may be specified with a hostname or IP address. Default for the node is "localhost". Default for the port is the Communication Broker port on the specified node.

`-status {[<hostname>|<ip>[:<port>]]|[<uri>]}[-v|-verbose]`

Displays the status of the specified server process. A hostname or IP address with an optional port number may be given to locate the server process. Default for the node is "localhost". Default for the port is the Communication Broker port on the specified node.

The status message presents the details of all the active and attempted reverse channel connections. For every connection, the following details are listed:

Source machine

The details of the machine that tries to establish the reverse channel connection.

Time and date

The time and date when the node started trying to connect to the Communication Broker through a reverse channel.

Time duration

The time interval for which a node attempted to establish a connection to the Communication Broker through the reverse channel (in milliseconds).

The `verbose` option displays the following details of every failed connection:

Type of failure

A connection failure can be a time-out, rejection, or a reset. This information helps you identify the true nature of the failure.

Cause of failure

The cause of failure helps you diagnose the underlying problem that triggered the connection failure.

Attempts

The number of attempts made by the node to reinstate the communication is presented within parenthesis.

`-retryfailedrcp[-ovrg`*<resource_group>*`]`

This option starts to restore all failed reverse channel connections to the specified resource group. If you do not specify a resource group name, the command tries to restore all failed reverse channel connections to the default resource group.

EXIT STATUS

The following exit values are returned:

`0`	ovbbccb exited normally with no error.
`1`	Command syntax error encountered. See command syntax for more details on possible values.
`2`	Command partially succeeded.
`3`	Command failed. See command output for more detailed information.
`4`	The Communication Broker start command failed because a Communication Broker process is already running.
`5`	The Communication Broker failed to start because a Local Location

	Broker process is already running. The Communication Broker is not supported on systems running the LLB. Stop the LLB before attempting to start the Communication Broker.
`6`	The Communication Broker failed to stop because the Communication Broker process is already stopped.
`7`	The Communication Broker failed to start due to a bind exception on the Communication Broker port to be opened.
`8`	The Communication Broker could not complete the command due to an authorization error.
`100`	An exception was encountered causing the Communication Broker to exit. Corresponding error messages are written to stderror.

EXAMPLES

The following examples show you how to use the `ovbbccb` command:

- To start the Communication Broker as a daemon process on the local system:

```
ovbbccb -daemon
```

- To start the resource group `WebCluster1` in the Communication Broker on host `merlin:`

```
ovbbccb -start WebCluster1 merlin
```

- To display the status of the specified server process:

```
ovbbccb -status
```

The following output appears:

```
Status OK

(Namespace, Port, Bind Address, Open Sockets)

<default>    383    ANY    2HP OpenView HTTP Communication Incoming Connections

 To machine1.example.hp.com:

    localhost:17282  76bb6662-2cd3-7531-1221-b67340fb721f  BBC 06.10.209; ovbbccb
06.10.209


HTTP Communication Reverse Channel Connections

 Opened from machine1.example.hp.com:
```

```
machine31.example.hp.com:8188     BBC 06.10.143; ovbbcrcp 06.10.143   (1)       30 Jan
2009 15:38:13 GMT  317 ms

machine32.example.hp.com:8196     BBC 06.10.143; ovbbcrcp 06.10.143   (1)       30 Jan
2009 15:38:13 GMT  241 ms

 Failed from:

machine21.example.hp.com:8188     BBC 06.10.143; ovbbcrcp 06.10.143   (1)       30 Jan
2009 15:38:13 GMT  307 ms

machine22.example.hp.com:8196     BBC 06.10.143; ovbbcrcp 06.10.143   (1)       30 Jan
2009 15:38:13 GMT  291 ms


Pending from :

machine11.example.hp.com:6244   Connection Refused / remote RCProxy not listening
(1)     30 Jan 2009 15:37:58 GMT  3 ms

machine12.example.hp.com:6252 Connection Refused / remote RCProxy not listening (1)
30 Jan 2009 15:37:58 GMT 2 ms
```

ovbbcrcp

NAME

ovbbcrcp – a tool to manage Reverse Channel Proxy (RCP) and monitor RCP connections.

SYNOPSIS

```
ovbbcrcp -h|-help

ovbbcrcp -v|-version

ovbbcrcp -kill

ovbbcrcp -status
```

DESCRIPTION

You can use the ovbbcrcp tool to manage RCPs and monitor RCP connections. Many HPE BTO Software products that follow a client-server architecture use the Black Box Communication component for communication. You can use a Reverse Channel Proxy (RCP) to satisfy the advanced security requirements for communication across trusted zones separated by firewalls. An RCP allows you to establish a two-way communication (outbound and inbound) channel across a firewall configured to allow only outbound communication.

The RCP functions as a channel between the communication server and the requests to the communication server. An established RCP channel is referred to as a reverse channel. A reverse channel through which RCPs request the communication server to initiate more reverse channels is referred to as a reverse administration channel.

You can deploy an RCP on one of the following:

- Any client systems
- A dedicated RCP server

To establish a reverse channel, you must configure the communication server, the communication client, and the RCP.

Configuring a communication Server to Enable RCP Communication

To enable communication from clients to the communication server through an RCP, you must configure each communication server. The communication server loads the configuration from the `bbc.<server>` namespace and establishes reverse administration channels during startup. Use the following options to configure a communication server:

- `ENABLE_REVERSE_ADMIN_CHANNELS`- You can set this option to true to establish a permanent reverse administration channel with the RCPs specified in the `RC_CHANNELS` option. By default, this option is set to `false` for all communication servers, except for the Communication Broker (CB). Refer to the following example for more information about this option.

[bbc.cb]

ENABLE_REVERSE_ADMIN_CHANNELS=true

RC_CHANNELS=pnode:9090

The options specified in the example instructs Communication Broker on the management server to contact the RCP on the `pnode` node and port 9090 when starting up.

- `RC_CHANNELS`- Use this option to specify the list of RCPs with which you can establish reverse channels. If the OvCoreID is specified, the communication server validates this ID against the core ID of the RCP. You can specify multiple RCPs by separating the RCPs using the semicolon (;). You can specify the list of RCPs in the following format:

`<RCP_hostname>:<RCP_port>[,<RCP_OvCoreID>][;<RCP2>…..],` where <*RCP_hostname*> specifies the RCP host name, <*RCP_port*> specifies the RCP port number, and <*RCP_OvCoreID*> specifies the core ID of the RCP.

You must use the `-ovrg server` option with the `ovconfchg` command if the OM server runs on a High Availability (HA) cluster. If the OM server runs as an HA resource group, then use the `ovconfchg -`

`ovrg server –ns bbc.cb –set RC_CHANNELS <value>` command, where <*value*> specifies the RCPs specified in the `RC_CHANNELS` option.

- `RC_MAX_WORKER_THREADS/RC_MIN_WORKER_THREADS`- The Communication Broker uses different threads to enhance the performance of a reverse channel connection. The `RC_MAX_WORKER_THREADS` option specifies the maximum number of threads that can be used by the Communication Broker and the `RC_MIN_WORKER_THREADS` option specifies the number of threads that will always remain active. By default, `RC_MAX_WORKER_THREADS` is set to one and `RC_MIN_WORKER_THREADS` is set to zero. You can set these options to higher values to enhance the reverse channel communication.
- `RC_CHANNELS_CFG_FILES`- Use this option to specify the list of configuration files. A configuration file can contain a list of one or more RCPs with which you can establish reverse channels. You must place the specified configuration files in the following directory:

On UNIX/Linux:

`/var/opt/OV/conf/bbc`

On Windows:

`%ovdatadir%conf\bbc`

If you are running the `ovbbcrcp` command on a management server in a cluster, you must use the `-ovrg` *`<resource_group>`* option. In that case, you must place the configuration file in the following location:

On UNIX/Linux:

`/var/opt/OV/shared/` *`<resource_group>`*`/conf/bbc`

On Windows:

`%ovdatadir%shared\` *`<resource_group>`*`\conf\bbc`

You must use this option in place of the `RC_CHANNELS` option if you use multiple RCPs that require a frequent hostname change. You can specify a list of configuration files by separating the configuration file names using the comma (,) in the following format:

`<filename>[,<filename>….], where` *<filename>* specifies the name of the configuration file.

Each line in the configuration file can contain only one RCP name. For each RCP, you must specify a port number. The OvCoreID is an optional parameter that you can specify, which must be separated from the port number by a comma as follows:

<RCP_hostname>:<port>[,<RCP_OvCoreID>]

If you change only a few RCP host names inside one or more files specified in the RC_CHANNELS_CFG_FILES option, you must use the `ovconfchg` command to trigger the communication server to refresh the configuration as follows:

```
ovconfchg ns bbc.cb –set ENABLE_REVERSE_ADMIN_CHANNELS TRUE
```

- `RETRY_INTERVAL`- Use this option to specify the retry interval in minutes to establish a reverse channel with an RCP.
- `RC_ENABLE_FAILED_OVEVENT`- Set this option to TRUE to forward the RCP connection failure messages to the OM message browser.

Enabling Communication Broker Connections to the RCP

The Communication Broker (ovbbccb) runs with `/var/opt/OV` as the root directory. The configuration files that are necessary to open Transmission Control Protocol (TCP) connections are present in the `/etc` directory. This prevents ovbbccb from creating connections to the RCP. You must do as follows to resolve this problem:

- Create the directory named `etc` under `/var/opt/OV`
- Copy the name service relevant configuration files (for example, files such as `resolv.conf, hosts, nsswitch.conf)` from `/etc to /var/opt/OV/etc`

Alternatively, you can also disable the `ovbbccb chroot` feature by running the following command. This method resolves the problem of preventing ovbbccb from creating connections to the RCP.

```
>ovconfchg -ns bbc.cb -set CHROOT_PATH /
```

> **Note:** Do not use the `-ovrg` option with this command even when the server exists in a cluster or server pooling setup. Disabling the `ovbbccb chroot` feature must be done for the local system.

Configuring a Communication Client to Enable RCP Communication

To configure a communication client, you must specify the hosts that must be connected through an RCP. You can specify the list of RCPs in the XPL configuration database under the bbc.http namespace. Use the syntax of the normal proxy configuration to specify the RCP configuration. If you do not specify the port number of the RCP, it is assumed that Communication Broker is running on the current node. If you configure the OvCoreID, communication client verifies the OvCoreID of the RCP. If the port number of the RCP is not specified in the configuration file or Communication Broker, the communication component fails to open the connection to RCP.

You can configure a communication client using the following options:

`PROXY`- Use this option to specify the RCP and port name for a hostname. The format to specify this option is shown in the following example:

```
PROXY=pnode.hp.com:9090-(pnode.hp.com,*.noallow.hp.com)+(*.hp.com)
```

In the example shown above, the parameters specified are as follows:

– `pnode.hp.com` is the name of the RCP

– `9090` is the port number

– `-(*.noallow.hp.com)` specifies that the RCP must not be used to connect to all hostnames ending with `.noallow.hp.com`. You can separate multiple hostnames with commas (,) or semicolons (;).

– `+(*.hp.com)` specifies that the specified RCP must be used to connect to all hostnames ending with .hp.com. You can separate multiple hostnames with commas (,) or semicolons (;).

The communication client connects to the RCP that first matches the specified set of conditions.

In the example shown in this section, the communication client connects to any host name that ends with `.hp.com` by using the RCP on the system `pnode` and the port `9090`.

You can also use IP addresses instead of hostnames to specify the hosts. For example, +(15.*.*.*) specifies that the RCP must be used to connect to hosts with an IP address that starts with 15. You must not configure a normal proxy server and an RCP on the same system. You must also make sure that you specify the RCP system name in the list of hostnames for which the RCP must not be used. This helps to ease the communication through the RCP.

Configuring RCP

You can use the following option in the `bbc.rcp` namespace to configure RCP.

`SERVER_PORT`- Use this option to specify the RCP port number.

Starting and Stopping RCPs

You can start or stop the RCP process by using the `ovc` command. This command registers the RCP process as `ovbbcrcp` under the `RCP` category.

By default, the **ovbbcrcp** process is not registered with Operations Control (OvCtrl). You must register the **ovbbcrcp** process with the `ovctrl` daemon by using the following command:

```
$OvInstallDir/bin/ovcreg –add $OvInstallDir/newconfig/DataDir/conf/bbc/ovbbcrcp.xml
```

`$OvInstallDir` is the directory in which HPE BTO Software is installed.

Refer to the following commands to start or stop a process:

- `ovc -start ovbbcrcp`- Use this command to start the RCP process.
- `ovc -stop ovbbcrcp`- Use this command to stop the RCP process.

Parameters

The `ovbbcrcp` command recognizes the following options:

`-h|-help`

Displays and describes the available options for the `ovbbcrcp` tool.

`-v|version`

Displays the version of the RCP.

`-kill`

Stops the RCP on the local node.

`-status`

Displays the RCP status.

EXIT STATUS

The following exit values are returned:

`0`	`ovbbcrcp` exited normally with no error.
`1`	Command syntax error encountered. Refer to command syntax for more details on possible values.
`2`	Command partially successful.
`3`	Command failed. See command output for additional information.
`4`	The command to start RCP failed due to an existing RCP process.
`6`	The RCP failed to start due to a bind exception on the RCP port to be opened.
`100`	An exception encountered resulted in an RCP exit.

Corresponding error messages are written to stderror.

EXAMPLES

The following example shows you how to use the `ovbbcrcp` tool.

- To display the status of the RCP:

```
ovbbcrcp -status

Status: OK

(Namespace, Port, Bind Address, Open Sockets)

bbc.rcp    9090    ANY    1
```

```
Admin Reverse Channel Connections Accepted

machine.example.hp.com:383  e91b67e4-a337-750a-163c-c3bbd2c257cc  BBC 06.00.030;
ovbbccb 06.00.030

Admin Reverse Channel Connections Opened


Normal Connections

Incoming

localhost:55464  e91b67e4-a337-750a-163c-c3bbd2c257cc  BBC 06.00.030; ovbbcrcp
06.00.030


Outgoing

Queued CONNECT connections

+------------------------------------+---------------------+

|Source Address | Target Address

+------------------------------------+--------------------

HTTP Tunnelled Connections

+---------------------------+---------------------------+---+

| Source Address | Destination Address | Target Address|

+---------------------------+---------------------------+---+
```

bbcutil

NAME

`bbcutil` – a tool for debugging the communication component.

SYNOPSIS

```
bbcutil -h|-help

bbcutil -version

bbcutil -ovrg [<ovrg>]

bbcutil -reg|-registrations [<hostname>|<ip>] [-v|-verbose]

bbcutil -deregister {<path>|*} [-force] [-v|-verbose]
bbcutil -ping {[<hostname>|<ip>[:<port>]] | [<uri>]} [count] [-v|-verbose]

bbcutil -status {[<hostname>|<ip>[:<port>]] | [<uri>} [-v|-verbose]]

bbcutil -migrate {[<namespace>] [<appname>] [<filename>]} [-v|-verbose]

bbcutil -count|-size|-list [-p|-path <path>] [-t|-target <target>] [-v|-verbose]

bbcutil -getcbport [<hostname>|<ip>]

bbcutil -gettarget [<hostname>|<ip>]
```

DESCRIPTION

The `bbcutil` command helps you to debug a communication server. The `bbcutil` command can be used to list all applications registered to a Communication Broker, to check whether specified communication services are alive, and to display details about the current state of the server.

Parameters

The `bbcutil` command incorporates the options in the following list. The syntax for the `[<hostname>|<ip>][:<port>]]` string, for example; in the options `-registrations` or `-ping`, can be a hostname and a port separated by a colon (`:`) but can also be a full URL path (including protocol), such as:

`https://merlin.guilford.mycom.com:383/Hewlett-Packard/OpenView/Coda/`

`bbcutil` recognizes the following options:

`-h|-help`

Displays and describes the available options for the `bbcutil` command.

`-version`

Displays the version of the communication software in use.

`-ovrg <ovrg>`

Executes a `bbcutil` command option in the context of the resource group specified by <*ovrg*>. This is an optional command. It can be used with other `bbcutil` commands. For example, `bbcutil -ovrg testsrv -getcbport` command returns the Communications Broker port number of the resource group, testsrv.

`-reg|-registrations [<hostname>|<ip>>]`

Queries a Communications Broker on the node specified by *<hostname>* or *<ip>* and displays a list of all registered applications. If the hostname or IP address is not specified, localhost is assumed.

`-deregister {<path>|*} [-force]`

Deregisters the specified path from the Communications Broker on the localhost. You can use the asterisk character '*' to denote all paths. The specified path will not be deregistered if the application servicing the specified path is currently running. Use the `-force` option to override this behavior and force the path to be deregistered.

`-ping {[<`*`hostname`*`ip >][:<`*`port`*`>]] | [<`*`uri`*`>]} [count]`

Pings the specified server process. A hostname or IP address with an optional port number or a URL may be given to locate the server process to ping. If a URL is given with the path of a valid process registered with the Communications Broker, the Communications Broker will automatically forward the ping to the registered process. Count specifies the number of times to execute the ping. The node may be specified with a hostname or IP address. Default for the node is “localhost”. Default for the port is the Communications Broker port on the specified node. Default count is 1.

`-status {[<`*`hostnameip`*`>[:<`*`port`*`>]] | [<`*`uri`*`>]}`

Displays the status of the specified server process. A hostname or IP address with an optional port number or a URI may be given to locate the server process. The node may be specified with a hostname or IP address. Default for the node is localhost. Default for the port is the Communications Broker on the specified node.

`-migrate {[<`*namespace*`>] [<`*appname*`>] [<`*filename*`>]} [-v|-verbose]`

Migrates the specified communication configuration parameters. If no command parameters are specified, the BBC 2 LLB and the BBC 4 CB parameters will be migrated to the namespace `bbc.cb` in the configuration database. The BBC 2/3 DEFAULT parameters will be migrated to the namespaces `bbc.http`, `bbc.fx`, and `bbc.snf`. BBC 4 CB parameters will override BBC 2 LLB parameters. The *namespace* parameter specifies the BBC 2/3/4 namespace to migrate the parameters from. The <*appname*> parameter specifies the application name to use in determining the BBC 5 target namespace. Parameters are migrated to the `bbc.http.ext.<appname>`, `bbc.fx.ext.<appname>`, and `bbc.snf.ext.<appname>` namespaces. The *filename* parameter specifies the file to read the parameters from. Default file name is the BBC 2 standard `default.txt` file and the standard BBC 4 Communication Broker `settings.ini` file. The BBC 4 settings.ini parameters override the BBC 2 default.txt parameters.

`-count`

Displays the number of requests in a store-and-forward buffer for the specified target, or the entire buffer if no target is specified.

`-size`

The `-size` option displays the size of a store-and-forward buffer. If `-verbose` is specified as well, the size of each individual request is displayed. If a target is specified, only the size of the requests to this target are displayed.

`-list`

The `-list` option displays all requests in a store-and-forward buffer for the specified target or the entire buffer if no target is specified.

`-p|-path` *<path>*

The `-path` option defines the path to the store-and-forward buffer. This parameter is used to set the BUFFER_PATH parameter.

`-t|-target` *<target>*

The `-target` option specifies the target URI, whose information you want to display. If no target is specified, information for all targets in the buffer is displayed.

`-verbose`

Shows more detailed output.

`-getcbport` [<*hostname*>|<*ip*>]

Displays the configured Communication Broker port number of the node specified by *<hostname>* or *<ip>*. If the hostname or IP address is not specified, localhost is assumed. If no Communication Broker port number is configured for the node, the default value 383 is displayed.

`-gettarget` [<*hostname*>|<*ip*>]

Displays the IP address of the target node and the Communication Broker port number, or the HTTP Proxy and port number, if a proxy is configured for the specified <*hostname*> or <*ip*>.

EXIT STATUS

The following exit values are returned:

0	`bbcutil` exited normally with no error.
1	Command syntax error encountered. See command syntax for more details on possible values.
2	Command partially succeeded.

3	Command failed. See command output for more detailed information.
4	`bbcutil` could not complete the requested command due to an authorization error.
100	An exception was encountered causing the Communication Broker to exit.

Corresponding error messages are written to `stderror`.

EXAMPLES

The following examples show you how to use the `bbcutil` command:

- To show the status of Communication Broker on the local node:

  ```
  bbcutil -status
  ```

- To query the communication server located at `https://merlin.guilford.mycom.com:383/Hewlett-Packard/OpenView/Coda/` for details about the current state of the server:

  ```
  bbcutil -ping https://merlin.guilford.mycom.com:383/Hewlett-
  Packard/OpenView/Coda/
  ```

- To get the IP address and Communication Broker port number of a target node `node1`:

  ```
  ode>bbcutil -gettarget node1
  ```

OVC

NAME

`ovc` – perform actions on local components

SYNOPSIS

```
ovc -h|-help

ovc -start [><target> ... ] [-boot]{[-async]|[-verbose]}

ovc -stop [<target> ... ][-nostart]{[-async]| [-verbose]}

ovc -restart [<target> ... ]

ovc -kill [-verbose]

ovc -status [<target> ... ] [-level <level>]

ovc -notify <event> [<target> ...] [-value <value>]

ovc -version
```

DESCRIPTION

`ovc` controls the start and stop actions, event notification, and status reporting of all components registered with the Operations Control service.

A component can be a server process belonging to any of the products such as Operations Manager for Windows, Operations Agent (for example, the Performance Agent or the Discovery Agent), an event interceptor, or an application delivered by an integrator. Each component must have an associated registration file providing Operations Manager with configuration and process information about the component. For more information about registration, `ovcreg(1)`.

A target can be either a component or a group of components, defined as a category. The `ovc` command first tries to initiate action on the category specified in `target`. If the category called `target` is not found, `ovc` then tries the individual component called `target`. Note that a category name must not match any component name.

The Operations Control daemon or service automatically restarts any component that terminates unexpectedly if the `AutoRestart` option in the registration file of the component is set to TRUE. If the Operations Control daemon or service is stopped using the `-kill` option, all registered components are stopped, too.

Parameters

`ovc` recognizes the following options:

`-h|-help`

Displays `all` available options for the `ovc` command.

`-start [<target> ... ] [-boot]{[-async]|[verbose]}`

Starts the selected components. `<target>` specifies a component or category. If `<target>` is not used, all components are started. If `-boot` is used, only components that start at boot time are started.

The `-async` option starts the components asynchronously. If you use the `-verbose` option, `ovc` command displays the progress of the command execution. You can use the `-async` or the `-verbose` option, but you must not include these options together in a command.

`-stop [<target> ... ] [-nostart]{[-async]|[verbose]}`

Stops the selected components. `<target>` specifies a component or category. If `<target>` is not used, all components are stopped except components, which belong to the CORE component group. If you specify the `-nostart` option and if the control daemon is not running, the command does not perform any action. If you do not specify the `-nostart` option, the `ovc -stop` command starts the control daemon and ovbbccb components if these components are not running. The `-async` option starts the components asynchronously. If you use the `-verbose` option, the `ovc` command displays the progress of the command execution. You can use the `-async` or the `-verbose` option, but you must not include these options together in a command.

`-restart [<target> ... ]`

Stops the components before they are restarted. `<target>` specifies a component or category. If `<target>` is not used, all components are stopped and restarted.

`-kill [-verbose]`

Stops all components registered with the Operations Control service. If you use the `-verbose` option, the `ovc` command displays the progress of the command execution.

`-notify <event> [<target> ... ] [-value <value>]`

Sends notification of an event with the value of `<value>` to the component or category specified by `<target> ...` . You can specify the `<value>` to the component that generates the event (event generator) and sends the event-related information to all components that request the event information (event subscribers). If target is not used, the event notification is sent to all components. If `<value>` is not used, only the event notification is sent.

`-status [<target> ... ] [-level <level>]`

Reports the status of a component or category specified by `<target>`. The status report contains the component's label, description, category, process ID, and state. The state of the components can be: Stopped (0 in numeric format), Starting (1), Initializing (2), Running (3), Stopping (4), N/A (5) or Aborted (6). If `<target>` is not specified, the status of all components is returned. `<level>` specifies the type and quantity of information to display, as follows:

`Level 0`	Status of registered components monitored by Operations Manager.
`Level 1`	Status of registered components whether they are monitored by Operations Manager or not.
`Level 2`	Status of registered components and a dump of their registration information.
`Level 3`	ID of core processes. 0 (zero) indicates root, non-zero indicates non-root ownership.
`Level 4`	Similar to level 0, but the STATE is reported in numeric format.

`Level 5`	Similar to level 1, but the STATE is reported in numeric format.
`Level 6`	Similar to level 0, but the output is not formatted
`Level 7`	Similar to level 1, but the output is not formatted
`Level 8`	Shows the detailed status of the processes with the recent history of each process.

`-version`

Prints the version of `ovc`

EXIT STATUS

The following exit values are returned:

0	Success.
1	Not defined.
2	Ignored.
62	The UNIX daemon or Windows service is not running.
63	The Control daemon is being initialized.
64	Generic error.
65	Invalid target.
67	Operation aborted.
69	Missing prerequisite.
70	Authorization error.
71	Operation on prerequisite failed.
73	Invalid event.

EXAMPLES

The following examples show how to use the `ovc` command and some of its options to control and display important information about registered components.

- To start the component registered as `opcle`:

  ```
  ovc -start opcle
  ```

  Before `opcle` itself starts, all the components that `opcle` depends on are started.

- To start the component registered as `opcle` and display the progress of the command execution:

```
ovc -start opcle -verbose
```

Before opcle itself starts, all the components that opcle depends on are started.

- To print the status of all registered components:

  ```
  ovc -status
  ```

- To stop the component registered as opcle:

  ```
  ovc -stop opcle -verbose
  ```

  Before opcle itself stops, all the components that depend on opcle are stopped. This command starts the control daemon and ovbbccb components if these components are not running.

- To stop the component registered as opcle using the ovc -stop[<target>...] -nostart option:

  ```
  ovc -stop opcle -nostart
  ```

  Before opcle itself stops, all the components that depend on opcle are stopped. This command does not perform any action if the control daemon is not running.

- To send the event RECONFIGURE to all running components:

  ```
  ovc -notify RECONFIGURE
  ```

- To start all components (and their dependents) belonging to category SERVER and AGENT:

  ```
  ovc -start SERVER AGENT
  ```

- To print the status of the component opcle and display the registration details:

  ```
  ovc -status opcle -level 2
  ```

ovcreg

NAME

ovcreg – component registration tool

SYNOPSIS

```
ovcreg -h|-help

ovcreg -check [ <filename> ]

ovcreg -add [ <filename> ]

ovcreg -del [ <component> ]
```

```
ovcreg -version
```

DESCRIPTION

`ovcreg` is used to register a component with (and de-register the component from) the OvCtrl. The `ovcreg` command can also be used to check a component registration file for syntactical correctness.

If the OvCtrl daemon (`ovcd`) is running at the time of registration, it will be informed about the new component only if the `-add` option was applied and the component is not started. The OvCtrl shows the new component the next time the `ovc` command is called with the `-status` option.

If the OvCtrl daemon (`ovcd`) is running, the component will be stopped if the `-del`(ete) option was applied.

> **Note:** This option will not stop CORE components, which are denoted by the option `CoreProcess` in the registration file. CORE components should be stopped with `ovc` command and the `-kill` option.

Parameters

`ovcreg` recognizes the following options:

`-h|-help`

Displays `all` available options for the `ovcreg` command.

`-check [< filename >]`

Checks the syntax of `<filename>. <filename>` must not contain more than one component.

`-add [<filename>]`

Checks the syntax of `<filename>` and stores a copy in the configuration directory. Adding a component with a name which is already registered with the OvCtrl will overwrite the original registration with the new one. `<filename>` must not contain more than one component.

`-del [<component>]`

Stops and deregisters the specified `<component>` from the OvCtrl and deletes the specified `<component>` registration file.

> **Note:** The `delete` option does not stop CORE components.

`-version`

Displays the version of ovcreg

EXIT STATUS

The following exit values are returned:

`0` Success - The syntax of the file is correct and the registration file is successfully added or deleted.

`1` Wrong usage

`2` Parsing error

`3` Error deleting registration file

`5` Error writing XML file

`6` Component is not registered

`7` Error stopping component

`8` Error deleting component

FILES

Registration files for components registered with the OvCtrl for the supported platforms reside in the following locations:

On AIX/HP-UX/Linux/Solaris:

```
/var/opt/OV/conf/ctrl/*.xml
```

On Windows:

```
%OvDataDir%\conf\ctrl\*.xml
```

Note that the user can change the specified default location for the registration files on machines running on Windows.

EXAMPLES

The following examples show how to use the `ovcreg` command and some of its options to control and display important information about registered components.

- To check the syntax of the component registration file: `opcle.xml:`

  ```
  ovcreg -check opcle.xml
  ```

- To check the syntax of the component registration file, `opcle.xml`, and add the component defined in the component registration file: `opcle.xml` to the OvCtrl:

  ```
  ovcreg -add opcle.xml
  ```

- To stop and de-register the component registered as `opcle`:

  ```
  ovcreg -del opcle
  ```

ovcert

NAME

`ovcert` – Manages certificates with the Certificate Client on an HTTPS based node.

SYNOPSIS

```
ovcert -h|-help

ovcert -importcert -file <file> [-pass <passphrase>] [-ovrg <ov_resource_group>]

ovcert -exportcert -file <file> [-alias <alias>] [-pass <passphrase>] [-ovrg <ov_
resource_group>]

ovcert -importtrusted -file <file> [-ovrg <ov_resource_group>]

ovcert -exporttrusted -file <file> [-alias <alias>] [-ovrg <ov_resource_group>]

ovcert -certreq [-instkey <file> [-pass <passphrase>]]

ovcert -list [-ovrg <ov_resource_group>]

ovcert -remove <alias> [-f] [-ovrg <ov_resource_group>]

ovcert -certinfo <alias> [-ovrg <ov_resource_group>]

ovcert -check

ovcert -status

ovcert -updatetrusted

ovcert -version
```

DESCRIPTION

The `ovcert` command is used to manage certificates with the Certificate Client on an HTTPS-based node. You can execute tasks such as initiating a new certificate request to the Certificate Server, adding node certificates and importing the private keys, and adding certificates to the trusted root certificates.

Parameters

The `ovcert` command incorporates the following options:

`-h|-help`

Displays usage help for the `ovcert` command options.

`-importcert -file <file> [-pass <passphrase>] [-ovrg <ov_resource_group>]`

Adds the certificate located in the file `<file>` (in `PKCS12`format) as node certificate and imports the private key which must be located in the same file as the private key for the node. The pass phrase for protecting the exported data using encryption specified during creation of the data to import must be specified as parameter `<passphrase>`.

The optional `<ov_resource_group>` parameter can be specified to import an additional certificate on an HA system. As a result, the specified certificate will not be imported to the default location but to the HA default location for the specified package on the shared disk.

`-exportcert -file <file> [-alias <alias>] [-pass <passphrase>] [-ovrg <ov_resource_group>]`

Exports the currently installed node certificate together with its private key to the file system location specified as parameter `<file>` (in `PKCS12` format). The passphrase for protecting the exported data using encryption specified during creation of the data to import must be specified as parameter `<passphrase>`.

The optional `<ov_resource_group>` parameter can be specified to export an additional certificate on an HA system. As a result, not the default node certificate but the certificate installed for the specified HA package from the shared disk will be exported.

`-importtrusted -file <file> [-ovrg <ov_resource_group>]`

Adds the certificate located in the specified file (in PEM format) to the trusted root certificates.

The optional `<ov_resource_group>` parameter can be specified to import an additional root certificate on an HA system. As a result, the specified root certificates will not be imported to the default location but to the HA default location for the specified package on the shared disk.

`-exporttrusted -file <file> [-alias <alias>] [-ovrg <ov_resource_group>`

Exports the trusted certificate to the file system location specified as parameter `<file>` (in PEM format). The pass phrase for protecting the exported data using encryption specified during creation of the data to import must be specified as parameter `<passphrase>`.

The optional `<ov_resource_group>` parameter can be specified to export an additional certificate on an HA system. As a result, not the default node certificate but the certificate installed for the specified HA package from the shared disk will be exported.

`-certreq [-instkey <file> [-pass <passphrase>]]`

Initiates a new certificate request that is sent to the Certificate Server.

The optional parameters `<file>` and `<passphrase>` can be used to initiate a certificate request that will be based on the installation key that is contained in the specified file. Such an installation key file can be generated with the `ovcm` tool on the certificate server.

The installation key can be used to authenticate the node on the certificate server. Therefore, such a request may be granted automatically without human interaction.

`-list [-ovrg <ov_resource_group>]`

Displays the aliases of the installed certificates and trusted certificates.

`-certinfo <alias> [-ovrg <ov_resource_group>]`

Displays information such as serial number, issuer, subject, and fingerprint for the certificate specified by `<alias>`.

`-remove <alias> [-ovrg <ov_resource_group>]`

Removes the certificate specified by `<alias>`.

`-check`

Checks whether all prerequisites for SSL communication are fulfilled, such as assigned OvCoreId, installed and valid certificate and private key, and installed and valid trusted certificate.

On completion, the components checked and their status along with the final result are displayed.

`-status`

Contacts the Certificate Client and displays the current certificate status, which can be one of the following possible values:

- certificate installed

- no certificate

- pending certificate request

- certificate request denied

- undefined (if Certificate Client cannot be contacted)

`-updatetrusted`

Retrieves the currently trusted certificates from the Certificate Server and installs them as trusted certificates on the node.

`-version`

Returns the version of the tool (the component version).

EXIT STATUS

The following exit values are returned:

`0` All steps were successful.

`1` One or more steps were not successful.

Corresponding error messages are written to `stderror`.

EXAMPLES

The following examples show how to use the `ovcert` command:

- To import the certificate, private key, and trusted certificates located in the file <`file`> to the system's keystore:

  ```
  ovcert -importcert -file <file>
  ```

- To add the certificate(s) located in <`file`> to the trusted certificates:

  ```
  ovcert -importtrusted -file <file>
  ```

ovcm

NAME

`ovcm` – manages certificates with the Certificate Server in an HTTPS-based environment.

SYNOPSIS

```
ovcm -h|-help
```

```
ovcm -version
```

```
ovcm -newcacert [-ni]
```

```
ovcm -importcacert -file <file> [-pass <passphrase>]
```

```
ovcm -exportcacert -file <file> [-pass <passphrase>]
```

```
ovcm -listpending [-l]
```

```
ovcm -grant <reqid>
```

```
ovcm -deny <reqid>
```

```
ovcm -remove <reqid>
```

```
ovcm -issue -file <file> -name <nodename> [-pass <passphrase>] [-coreid <OvCoreId>]
[-ca]
```

```
ovcm -genInstKey -file <file> [-context <context>] [-pass <passphrase>]
```

DESCRIPTION

The `ovcm` command is used to manage certificates with the Certificate Server in an HTTPS based environment. You can execute tasks such as creating public/private key pairs for signing certificates, granting and issuing signed certificates and the corresponding private keys against certificate requests from HTTPS nodes.

Parameters

The `ovcm` command provides the following options:

`-h|-help`

Displays all the command-line options for the `ovcm` command.

`-version`

Returns the version of the tool (the component version).

`-newcacert [-ni]`

Creates a new public/private key pair for signing certificates. If there is already a public/private key pair in use by the certification authority, you are asked whether this should be replaced. Use this option with care! An initial public/private key pair is automatically created when the Certificate Management component is installed. The `-ni` non-interactive option creates a new public/private key pair without operator interaction. If a public/private key pair already exists, the request is canceled.

`-importcacert -file <file> [-pass <passphrase>]`

Imports a certificate for signing certificate requests together with its private key (both contained in one file in PKCS12 format). Use this option with care as the existing certificate and private key are replaced. This option helps in restoring a backup of the current private key/certificate, for example, if the originals are damaged or destroyed, or for setting up a backup system.

Use `<file>` to specify the name of the file (in PKCS12 format) to import from.

Use `<passphrase>` to specify the text string you use to protect the data. If the `-pass` option is not used, you are prompted to enter the value of the pass phrase.

`-exportcacert -file <file> [-pass <passphrase>]`

Exports the certificate and the corresponding private key of the current certification authority to a file. This option is used for creating backups. The certification authority private key must be handled very carefully because of its importance to the whole communication environment. It should never be transmitted over the network or stored in an insecure place.

Use `<file>` to specify the name of the file where the certificate data should be written to (in PKCS12 format).

Use `<passphrase>` to specify the text string that you use to protect the data. If the `-pass` option is not used, you are prompted to enter the value of the passphrase.

`-listPending [-l]`

Displays the request IDs of all pending certificate requests.

With the `-l` option, detailed information on every pending request is listed.

`-grant <reqid>`

The selected certificate request is granted and a signed certificate is sent to the requesting certificate client.

The state of the pending certificate request with the request ID `<reqid>` is changed to granted.

`-deny <reqid>`

The selected certificate request is denied and a message is sent to the requesting certificate client.

The state of the pending certificate request with the request ID `<reqid>` is changed to denied.

`-remove <reqid>`

The selected certificate request is removed from the pending pool. No message is sent to the requesting certificate client.

The state of the pending certificate request with the request ID `<reqid>` is changed to removed.

`-issue -file <file> -name <nodename> [-pass <passphrase>] [-coreid <OvCoreId>] [-ca]`

Issues a signed certificate and the associated private key for a node and writes both to the file `<file>` (in PKCS12 format). The file can then be moved to a portable medium and taken to the corresponding node.

The `<nodename>` must be specified as additional information.

The optional `<OvCoreId>` parameter can be used to specify the unique ID of the certificate. If this parameter is empty, a new `OvCoreId` value is generated for the certificate.

The `<passphrase>` parameter is required to protect the generated certificate data. The pass phrase entered is used to calculate an encryption key that then is used to encrypt the generated certificate data. If the `-pass` option is not used, you are prompted to enter the value of the pass phrase.

If you use the `-ca` option, you can use the issued certificate to sign other certificates. This may be necessary if you want to set up a second Certificate Server, which creates certificates that are trusted by all nodes that trust the root Certificate Server.

`-genInstKey -file <file> [-context <context> ] [-pass <passphrase> ]`

Creates a new installation key, which, together with some additional information is stored in the file `<file>`. The created file should then be securely transferred to the node system.

On the target node, it can then be used to initiate a new certificate request that will be encrypted with the installation key. The certificate server will accept only one request that is encrypted with this key.

This approach offers the advantage that the certificate request (including the private key) is generated on the node system and the system can be authenticated by using the installation key.

The optional parameter `<context>` can be used to add additional (application specific) information that is contained in the certificate request.

The `<passphrase>` parameter is required to protect the generated installation key. The pass phrase entered is used to calculate an encryption key that then is used to encrypt the generated installation key. If the -pass option is not used, you are prompted to enter the value of the pass phrase.

EXIT STATUS

The following exit values are returned:

0 All steps were successful.

1 One or more steps were not successful.

Corresponding error messages are written to stderror.

EXAMPLES

The following examples show how to use the ovcm command:

- To create a new public/private key pair for the signing of certificates on the management server system:

  ```
  ovcm -newcacert
  ```

- To grant the certificate request *reqid* and send a signed certificate to the requesting certificate client:

  ```
  ovcm -grant reqid
  ```

ovcoreid

NAME

ovcoreid – Manages the unique node identifier OvCoreId on the local node.

SYNOPSIS

```
ovcoreid -show [-ovrg < OV_Resource_Group >]
```

```
ovcoreid -create [-force] [-ovrg < OV_Resource_Group >]
```

```
ovcoreid -set < OvCoreId > [-force] [-ovrg < OV_Resource_Group >]
```

```
ovcoreid -version
```

```
ovcoreid -h|-help
```

DESCRIPTION

The ovcoreid command is used to display existing OvCoreId values and, in addition, create and set new OvCoreId values on the local node.

Parameters

The ovcoreid command accepts the following parameters and options:

`-show [-ovrg < OV_Resource_Group >]`

Displays the current OvCoreId of the system (configuration setting CORE_ID in namespace [sec.core]). This is the default if no parameters are specified. If the OvCoreId you want to show belongs to an OpenView Resource Group, use the -ovrg option to specify the name of the Resource Group. If a resource group is specified, the corresponding configuration settings will be read or modified as well.

If you specify a non-existent resource group, ovcoreid displays the local OvCoreId.

`-create [-force] [-ovrg < OV_Resource_Group >]`

Generates a new OvCoreId. If a CORE_ID value already exists, the existing OvCoreId is only overridden when -force is specified. If the OvCoreId you want to show belongs to an OpenView Resource Group, use the -ovrg option to specify the name of the Resource Group. If a resource group is specified, the corresponding configuration settings will be read or modified as well.

If you specify a non-existent resource group, ovcoreid displays an error.

`-set [-force] [-ovrg < OV_Resource_Group >]`

Sets a specific OvCoreId. The -force option must be used if an OvCoreId value has already been set. If the OvCoreId you want to show belongs to an OpenView resource group, use the -ovrg option to specify the name of the resource group. If a resource group is specified, the corresponding configuration settings will be read or modified as well.

`-version`

Returns the version of the tool (the component version).

`-h|-help`

Display all available command options.

EXIT STATUS

The following exit values are returned:

0 All steps were successful.

1 If -create or -set is used without -force and a value for OvCoreId already exists.

2 One or more steps were not successful.

Corresponding error messages are written to stderror.

Changing the `OvCoreId` of a system is analogous to giving the system a new identity and is an action that should only be executed if the consequences are fully understood. Changing the `OvCoreId` of a system requires a number of significant changes including the need for a new certificate, and having to do appropriate reconfiguration of the server(s).

EXAMPLES

The following examples show you how to use the `ovcoreid` command:

- To display the `OvCoreId` for the local node:

  ```
  ovcoreid -show
  ```

- To create and set a new `OvCoreId` on the local node:

  ```
  ovcoreid -create
  ```

- To set the specified `OvCoreId` on the local node:

  ```
  ovcoreid -set <OvCoreId>
  ```

ovconfchg

NAME

`ovconfchg` - manipulates setting files, updates the configuration database, and triggers notification scripts

SYNOPSIS

```
ovconfchg -h | -help
```

```
ovconfchg -version
```

```
ovconfchg [-ovrg <OVRG>] [-edit | -job {-ns namespace {-set <attr> <value> | -clear <attr> | -clear -all} ... } ... ]
```

DESCRIPTION

Installed Operations Manager components have associated configuration settings files that contain one or more namespaces. A namespace is a group of configuration settings that belong to a component.

`ovconfchg` manipulates the settings in either the system-wide configuration file or the configuration file for the specified resource group, `local_settings.ini`, updates the configuration database, `settings.dat`, and triggers notification scripts. If `ovconfchg` is called without options, or only with -

`ovrg`, no settings are changed but an update is triggered anyway. This is to allow updating after default settings files have been added, removed, or updated.

When `ovconfchg` runs, all configuration settings are read and merged in memory. Default definitions are used to make corresponding checks, as well as to emit and log warnings in the event of a violation. During this process, file locks are used to prevent parallel updates. A new configuration database is then created containing the merged data.

> **Note:** When a variable value is modified using the `ovconfchg` command and then tried to change using the nodeinfo policy, the new values set through the nodeinfo policy will not be reflected on the node. This is because the order of precedence for the config settings is as follows:
>
> **Default settings** < **Policy Settings** < **Local Settings**
>
> (ovconfchg on the node has highest priority).

Parameters

`ovconfchg` recognizes the following options:

`-h | -help`

Displays all the options for the `ovconfchg` command.

`-version`

Displays the version of the `ovconfchg` command.

`-ovrg <OVRG >`

If the parameter you want to change belongs to a resource group, use `-ovrg` to specify the name of the resource group. Otherwise, system-wide settings files are opened.

`-edit`

Starts a text editor to edit the settings file, local_settings.ini. The text editor used is determined by the $EDITOR environment variable. If $EDITOR is not set, vi starts on UNIX and Notepad starts on Windows.

A temporary copy of the file is created for editing. After the changes are made, the file is validated for syntax errors. The syntax rule for validation is that the namespace and attribute names should contain only letters (a-z, A-Z), digits (0-9), period (.) and underscore (_) characters.

If the validation fails, the line number of the error is reported and the user will be prompted to correct the file. If Yes, the file will be reopened for making the necessary changes. If No, the original settings file remains unchanged. If the validation is successful, the changes are saved into the original settings file.

Do not configure binary values using this option. This can corrupt the file. It is also recommended to restrict the data entered using this option to US-ASCII (7-bit only) subset.

Do not open the settings file directly in a text editor and change it. This can corrupt the file.

`-job`

Create and update the job file only and do not synchronize.

`-ns | -namespace <namespace>`

Sets a namespace for the `-set` and `-clear` options.

`-set <attr > < value >`

Sets an attribute value in the namespace specified by the `-namespace` option. The local or resource settings file is updated accordingly.

`-clear < attr >`

Clears the local setting for the attribute `attr` in the namespace specified by the `-namespace` option. The local settings file is updated accordingly.

`-clear -all`

Clears all local settings. The local settings file is updated accordingly.

FILES

The `ovconfchg` command uses the following files to store local settings:

`<DataDir>/conf/xpl/config/local_settings.ini`

`<ShareDir >/< OVRG >/conf/xpl/config/local_settings.ini`

The `ovconfchg` command uses the following files to store database configuration settings:

`<DataDir >/datafiles/xpl/config/settings.dat`

`<ShareDir >/< OVRG >/datafiles/xpl/settings.dat`

EXAMPLES

The following examples show how to use the `ovconfchg` command:

- To assign the value 12 to the attribute COUNT, and assign the value "red blue white" to the attribute COLORS in the namespace, tst.lib:

  ```
  ovconfchg -ns tst.lib -set COUNT 12 -set COLORS "red blue white"
  ```

- To clear the attribute COUNT in the namespace tst.lib:

  ```
  ovconfchg -ns tst.lib -clear COUNT
  ```

- To remove all locally configured attributes from the namespace tst.lib:

  ```
  ovconfchg -ns tst.lib -clear '*'
  ```

- For the resource group server, assign the value 50 to the attribute COUNT in the namespace tst.lib:

  ```
  ovconfchg -ovrg server -ns tst.lib -set COUNT 50
  ```

ovconfget

NAME

ovconfget – returns specified attributes from the configuration database.

SYNOPSIS

```
ovconfget -h | -help
```

```
ovconfget -version
```

```
ovconfget [-ovrg < OVRG >] [< > [ <attr >]]
```

DESCRIPTION

Installed HPE Software components have associated configuration settings files that contain one or more namespaces and apply system wide or for a specified resource group. A namespace is a group of configuration settings that belong to a component. All configurations specified in the settings files are duplicated in the settings.dat configuration database.

For each specified namespace, ovconfget returns the specified attribute or attributes and writes them to stdout. When used without arguments, ovconfget writes all attributes in all namespaces to stdout.

Parameters

ovconfget recognizes the following options:

```
-h | -help
```

Displays the options for the `ovconfget` command

`-version`

Displays the component version

`-ovrg <OVRG>`

Specifies the named resource group `<OVRG>`.

`<namespace> <attr>`

Obtains the specified attribute in the specified namespace for the named resource group `<OVRG>` and writes them to stdout. If `namespace` is used without specifying an attribute, `<attr>`, `ovconfget` writes the contents of the database for the specified namespace. If neither `<attr>` nor `<namespace>` is specified, `ovconfget` writes the complete contents of the configuration database to stdout.

FILES

The `ovconfget` command uses the following files to read configuration-database settings:

```
<DataDir>/datafiles/xpl/config/settings.dat
```

```
<ShareDir>/<OVRG>/datafiles/xpl/settings.dat
```

EXAMPLES

The following examples show how to use the `ovconfget` command:

- To return the value of the `Port` attribute in the `tst.settings` namespace, for example:

  ```
  9012
  ```

  ```
  ovconfget tst.settings Port
  ```

  ```
  9012
  ```

- To return all attributes in the `tst.settings` namespace as multiple lines in the form of `attr=value`, for example:

  ```
  ovconfget tst.settings
  ```

  ```
  Port=9012

  Protocols=HTTP FTP HTTPS

  MaxFileSize=128
  ```

- To return all attributes in all namespaces on multiple lines, for example:

  ```
  ovconfget
  ```

```
[tst.lib]

LibraryPath=/opt/OV/lib:/opt/OV/lbin/tst/var/opt/OV/tmp

[tst.settings]

Port=9012

Protocols=HTTP FTP HTTPS

MaxFileSize=128
```

ovlogdump

NAME

`ovlogdump` – dumps a specified binary log file as text in the current locale to the console

SYNOPSIS

```
ovlogdump -h|-help

ovlogdump -version

ovlogdump [< binary_logfile_name >]

ovlogdump -merge -tofile < binary_logfile_name > -fromfiles < binary_logfile1_name
> < binary_logfile2_name >...
```

DESCRIPTION

The `ovlogdump` command dumps a binary log file as text in the current locale to the console. To view the contents of a log file, specify its location and name; else, the `system.bin` file is dumped to the console by default.

By default, all the log files are stored in the following location:

On Windows:

```
C:\Documents and Settings\All Users\Application Data\HP\HP BTO Software\log
```

On UNIX:

```
/var/opt/OV/log
```

If permissions are inadequate for the default locations, the log files are stored in the `<OvDataDir>/log/public` directory.

During application logging, if multiple log files are created, you can use the `-merge` option to merge these files into a single binary log file.

Parameters

`ovlogdump` recognizes the following options:

`[ <binary_logfile_name>]`

The name and location of the binary log file to be dumped. If the log file name is not specified, `system.bin` file in the <`OVDataDir`>`/log/` directory is displayed on the console by default.

`-merge -tofile <binary_logfile_name> -fromfiles <binary_logfile1_name> <binary_logfile2_name>....`

Merges application log files specified by `<binary_logfile1_name>`.... into a single binary log file specified by <`binary_logfile_name`>. This option is not supported for merging system log files.

`-h|-help`

Displays all available options for the `ovlogdump` command.

`-version`

Displays the version of the `ovlogdump` command.

ovtrccfg

NAME

`ovtrccfg` – enables the tracing mechanism for supported applications on the local machine.

This utility is present in the following directories:

On Windows:

`%ovinstalldir%\support`

On HP-UX/Linux/Solaris:

`/opt/OV/support` and `/opt/perf/support`

On AIX:

`/usr/lpp/OV/support` and `/usr/lpp/perf/support`

> **Note:** For streaming agent the utility location is `ovtrccfg/ovtrcmon/ovtrcgui`.

SYNOPSIS

```
ovtrccfg -app|-application <application_name> [-cm|-component <component_name>] [-
sink <filename>] [-gc|-generate_configuration <filename>]
```

```
ovtrccfg -cf|-configuration <filename>
```

```
ovtrccfg -off
```

```
ovtrccfg -version
```

```
ovtrccfg -h|-help
```

```
ovtrccfg -vc
```

DESCRIPTION

The `ovtrccfg`> command helps you enable and configure the tracing mechanism to record the state of a supported application on the system where an HPE Software product is installed. By default, trace log files are placed into the application's home directory after you enable the tracing mechanism. When you configure the tracing mechanism with the `gc` option, all configuration details are directed to a trace configuration (`.tcf`) file. You can create and modify trace configuration files with the command or with a text editor.

In the trace configuration file, you can specify the location of trace log files with the `sink` option. When you start the tracing process without a configuration file, all available trace levels and categories are enabled. If you want to enable only select levels of tracing, you must use a trace configuration file.

The tracing mechanism provides the following different levels of tracing:

`Info`

Enable traces marked as information.

`Warn`

Enable traces marked as warning.

`Error`

Enable traces marked as error.

`Support`

Enable the normal tracing. The trace output includes informational notifications, warnings, and error messages. This option is recommended for troubleshooting problems. This level of tracing can be enabled for a long duration as the overhead to capture the trace output is minimal with this option.

In addition, you can use the location, stack, developer, and verbose levels when detailed trace messages are requested by HPE Support.

Parameters

The `ovtrccfg` command accepts the following parameters and options:

`-app|-application <application_name>`

This option helps you enable the tracing mechanism for selecting HPE Software applications. These applications are essentially programs, daemons, processes, and services that are used by different HPE Software products.

`-cm|-component <component_name>`

You can enable tracing of select components of an application with the `cm` option. By default, all components of an application are traced by the tracing mechanism. You can use the wildcard character (`*`) with this option. For example, the `ovtrccfg -app ovconfd -cm xpl*` command starts tracing for all the components, which belong to the `ovconfd` application, with the names that begin with `xpl`.

`-cf|-configuration <filename>`

You can enable the tracing mechanism according to the rules specified in a configuration file. The configuration files are stored on the same system with the extension `.tcf`.

`-sink <filename>`

The `sink` option helps you direct the trace log files to a location of your choice on the local system. All trace log files generated with the command are placed into the location specified with the `sink` option.

`-gc|-generate_configuration <filename>`

The `gc` option creates a trace configuration file (`.tcf`) that can be edited to set the desired tracing configuration.

`-off`

The `off` option helps you disable the tracing process. If you use the `off` option without any other options, the entire tracing mechanism stops. You can use the `app` and `cm` options with the `off` option to conditionally exclude select applications and components when you enable tracing. For example, the "`ovtrccfg -app o* -off ovc*`" command enables tracing for all applications with the names that begin with "o," but excludes the applications with the names that begin with "ovc". Similarly, the "ovtrccfg -app ovoadif -cm e* -off eaagt.misc" command enables the tracing mechanism for all components with the names that begin with "e," which belong to the application "ovoadif," except the component eaagt.misc.

-vc

This option displays the current tracing status of all the supported applications available on the system.

-version

This option displays the version of this command.

-h|-help

Displays all available command options.

EXAMPLE

The following examples show how to use the ovtrccfg command:

- Enable the tracing mechanism for all applications with the names that begin with **o**:

  ```
  ovtrccfg -app “o*”
  ```

- Enable the tracing mechanism for the coda application and direct the trace log files to the /opt/OV/support directory:

  ```
  ovtrccfg -app coda -sink /opt/OV/support/output.trc
  ```

- Enable the tracing mechanism on the local system based on the rules set in the trace configuration file config.tcf:

  ```
  ovtrccfg -cf config.tcf
  ```

ovtrcmon

NAME

ovtrcmon – helps you view the trace messages from trace files and enables you to store the trace messages into another file on the same system.

This utility is present in the following directories:

On Windows:

```
%ovinstalldir%\support
```

On HP-UX/Linux/Solaris:

`/opt/OV/support` and `/opt/perf/support`

On AIX:

`/usr/lpp/OV/support` and `/usr/lpp/perf/support`

SYNOPSIS

```
ovtrcmon [-h|-help] -fromfile <source_file> -tofile <target_file>] -short|-long|-
verbose|[-fmt <format_name>]
```

DESCRIPTION

The `ovtrcmon` command helps you view the contents of a trace file and lets you store the file content into another file on the same machine. When you start the tracing mechanism with the `ovtrccfg` command, trace messages get captured into trace files in the binary format. To read the contents of a trace file, you can use the "`ovtrcmon -fromfile <source_file> -fmt <format>`" command. Alternatively, you can store the contents of a trace file into a new file in a readable format with the "`ovtrcmon -fromfile <source_file> -tofile <target_file> -fmt <format>`" command. With the help of the configuration file `$OvDataDir/conf/xpl/trc/ovtrcmon.cfg`, you can specify a customized format of your choice that you want to use while viewing and storing the contents of trace files. You can use the following keywords while configuring this file:

`Severity`

The trace file captures trace messages with different severity levels. This keyword helps you filter the trace messages based on the severity level. Available severity levels are: Info, Warn, Error, Support, Location, Stack, Developer, and Verbose.

`Count`

The serial number for a particular trace message.

`Tic`

A high-resolution elapsed time value.

`LocalTime`

The local equivalent date and time of the trace message.

`UTCTime`

The UTC time of the trace message.

`Pid`

The process ID of the traced application.

`Tid`

The thread ID of the traced application.

`Component`

The name of the component issuing the trace message.

`Category`

An arbitrary name assigned by the traced application or one of several categories provided by the tracing mechanism.

`Source`

The line number and file name of the source generating the trace.

`Stack`

A description of the calling stack in the traced application.

`TrcMsg`

Trace message description.

`Attribute`

Attribute of the trace message.

`Application`

Name of the traced application.

`Machine`

Name of the machine where the traced application resides.

`Formatting`

You can use one of four types of formatting on the trace output.

The Formatting keyword helps you generate the output in the following formats:

CSV

Comma separated values. This keyword presents the output in a standard delimited format with double quotes (") around the text.

formatted

A *printf*-like output format.

fixed

This keyword presents the output with fixed-width fields and white space padding. Field widths are specified after the keyword fixed with commas. For example, `fixed,w1,w2,..wn]`.

xml

Presents the trace output in the XML format.

Parameters

The `ovtrcmon` command accepts the following parameters:

`-fromfile <source_file >`

With this parameter, you can specify the name of the binary trace file.

`-tofile <target_file>`

With this parameter, you can specify the name of the file where you want to direct the contents of the trace file.

`-long`

Displays or stores the following details from the trace file: Severity, Component, Category, and trace description.

`-short`

Displays or stores only the trace description from the trace file.

`-verbose`

Displays or stores all details available in the trace file.

`-fmt`

With this parameter, you can view the contents of the trace file in a pre-configured format. You must specify the format definitions in the `$OvDataDir/conf/xpl/trc/ovtrcmon.cfg` file. You must declare *<format_name>* in this configuration file.

`-h|-help`

Displays all available command options.

EXAMPLE

The following examples show how to use the `ovtrcmon` command:

- View the trace messages in the `$OvDataDir/log/example1.trc` file in the format `format1`, which is defined in the `$OvDataDir/conf/xpl/trc/ovtrcmon.cfg` file:
  ```
  ovtrcmon -fromfile $OvDataDir/log/example1.trc -fmt format1
  ```
- View only the descriptions of the trace messages in the `$OvDataDir/log/example1.trc` file:
  ```
  ovtrcmon -fromfile $OvDataDir/log/example1.trc -short
  ```
- Store the available trace messages in the `$OvDataDir/log/example1.trc` file into the `$OvDataDir/log/trace.txt` file in the format `format1`, which is defined in the `$OvDataDir/conf/xpl/trc/ovtrcmon.cfg` file:
  ```
  ovtrcmon -fromfile $OvDataDir/log/example1.trc -tofile $OvDataDir/log/trace.txt
  -fmt format1
  ```

ovdeploy

NAME

`ovdeploy` – performs tasks related to software installation on local and remote hosts.

SYNOPSIS

`ovdeploy -install -pkg` *<package_descriptor>* `|-file` *<filename>*`... |-dir` *<dirname>* `[[-sourcerootdir` *<symbolic_name>*`] [-sourcedir` *<dirname>*`] [-targetrootdir` *<symbolic_name>*`] [-targetdir` *<dirname>*`] [-force] [-perm` *<file_permissions>*`] [-host` *<name_or_ip>*`] [-instserv` *<name_or_ip>*`] [-targetid` *<id>*`] [-cmd_timeout` *<time_in_milli_sec>*`]]`

`ovdeploy -remove -pkg` *<package_name>* `|-file` *<filename>* `|-dir` *<dirname>* `[[-targetrootdir` *<symbolic_name>*`] [-targetdir` *<dirname>*`] [-force] [-host` *<name_or_ip>*`] [-instserv` *<name_or_ip>*`] [targetid` *<id>*`] [-ovrg` *<id>*`] [-cmd_timeout` *<time_in_milli_sec>*`]]`

`ovdeploy -upload -pkg` *<package_descriptor>* `|-file` *<filename>* `|-dir` *<dirname>* `[[-sourcerootdir` *<symbolic_name>*`] [-sourcedir` *<dirname>*`] [-targetrootdir` *<symbolic_name>*`] [-targetdir` *<dirname>*`] [-force] [-perm` *<file_permissions>*`] [-host` *<name_or_ip>*`] [-instserv` *<name_or_ip>*`] [-targetid` *<id>*`] [-cmd_timeout` *<time_in_milli_sec>*`]]`

`ovdeploy -download -pkg` *<package_descriptor>* `|-file` *<filename>* `|-dir` *<dirname>* `[[-sourcerootdir` *<symbolic_name>*`] [-sourcedir` *<dirname>*`] [-targetrootdir` *<symbolic_name>*`] [-targetdir` *<dirname>*`] [-force] [-perm` *<file_permissions>*`] [-host` *<name_or_ip>*`] [-instserv` *<name_or_ip>*`] [-targetid` *<id>*`] [-ovrg` *<id>*`][-cmd_timeout` *<time_in_milli_sec>*`]]`

`ovdeploy -inv [-host` *<name_or_ip>*`] [-invtype` *<inventory_type>*`] [-all]`

```
ovdeploy -reg -pkg <package_descriptor>[ [-sourcerootdir <symbolic_name>] [-sourcedir <dirname>] [-force] [-host <name_or_ip>] [-targetid <id>] [-ovrg <id>]]
```

```
ovdeploy -unreg -pkgname <package_name> [[-force] [-host <name_or_ip>] [-targetid <id>] [-ovrg <id>]]
```

```
ovdeploy -exec -file <filename> [[-targetrootdir <symbolic_name>] [-targetdir <dirname>] [-shell] [-host <name_or_ip>] [-targetid <id>] [-ovrg <id>] [-cmd_timeout <time_in_milli_sec>]]
```

```
ovdeploy -cmd -file <filename>[-host <name_or_ip>] [-par <params>] [-cmd_timeout <time_in_milli_sec>][-targetrootdir <symbolic_name>] [-targetdir <dirname>]]
```

```
ovdeploy -get <node_attribute> [-node <name_or_ip>]
```

```
ovdeploy -env <environment_var> [-node <name_or_ip>]
```

DESCRIPTION

`ovdeploy` manages objects on local and remote hosts. An object can be a file, a directory, or a package. A package can be a file, a group of files, a directory or group of directories or a combination of all.

You use `ovdeploy` to install, remove, upload or download, and register or unregister objects on the local and remote hosts you are managing. You can also use the `ovdeploy` command to list package inventories and execute commands on specified files.

Parameters

`ovdeploy` recognizes the following options:

Options	Description
`-install` *<options>*	Installs the specified object
`-remove` *<options>*	Removes one or more objects
`-upload` *<options>*	Uploads one or more objects from a target node.
`-download` *<options>*	Downloads one or more objects from a target node.
`-inv` *<options>*	Returns a list of installed objects on a target node either to stdout or in the form of an XML file written to the local directory.
`-reg` *<options>*	Adds a specified package name to the list of registered package names on a target node.

-unreg *<options>*	Removes a specified package name from the list of registered package names on a target node.
-exec *<options>*	Executes a specified file on a specified host and returns the result of the action to stdout.
-cmd *<options>*	Executes a specified command or file on a different shell on the specified host and returns the results of the action on stdout.
-get *<options>*	Displays the value of an environment variable on a specified node. The environment variable can be, for example, `PATH` or `OvInstallDir`. The value of the environment variable can be obtained only if that variable is set on the specified node.
-env *<options>*	The `-list` option displays all requests in a store-and-forward buffer for the specified target or the entire buffer if no target is specified.

Options

The following options may be used in conjunction with the command parameters indicated in the Parameters section:

`-all`

Returns the complete inventory in XML format. The complete inventory comprises the package descriptors of all packages that have been installed on the node. If not used, only the name and version of each registered package is returned.

`-dir <dirname>`

The name of a directory to be installed, removed, up or downloaded, registered or unregistered. The contents of the directory will also be installed, removed, uploaded/downloaded.

`-file <filename>...`

The name of a file to be installed, removed, uploaded or downloaded, registered or unregistered, or executed.

With the `-install` parameter, multiple files can be specified.

`-force`

With the `-install` parameter, it installs the specified object even if an object already exists on the target node with the same or higher version. With the `-remove` parameter, it must be used with -dir, and removes all subdirectories.

`-host <target_host>`

The name or IP address of the target host. If not used, the local host is assumed.

`-instserv <name_or_ip>`

Name or IP address of an installation server. If an installation server is specified, the specified file is not copied from the local host to the target host. Instead, it is copied from the installation server to the target host.

`-invtype [depl|native]`

`depl` returns to `stdout` the inventory of objects installed using the `deploy` command. `native` returns to stdout the inventory of objects installed using the operating system's native installation program.

`-ovrg <id>`

The ID of an resource group, if applicable.

`-perm <nnn>`

Sets file permissions of an installed, uploaded/downloaded, unregistered/registered file. This parameter takes three digits. The first digit specifies the owner permission, the second digit specifies the group permission, and the third digit specifies public permissions. The allowable digits are:

`0`	No permissions
`1`	Execute
`2`	Write
`3`	Execute and write
`4`	Read
`5`	Execute and read
`6`	Read and write
`7`	Read, write, and execute

`-pkg <package_descriptor>`

The full path and name of the package descriptor file. The package descriptor file contains a list of all the files that are to be installed, removed, uploaded or downloaded, registered or unregistered and their locations.

`-shell`

Used with the `-exec` parameter to execute in a shell the file specified by `<file>`. For UNIX systems, `/bin/sh -c` is used. For a Windows system, `%ComSpec%/cmd.exe /c` is used.

`-sourcerootdir <symbolic_name>`

The symbolic path name used to create an absolute file path for the source file.

`-sourcedir <dirname>`

Used to create an absolute file path for the source file. If a source root directory is specified, the source directory is appended to it. If no source root directory is specified, the source directory is appended to the default directory, `$OvDataDir/installation/incoming/files/`.

`-cmd_timeout <time_in_milli_sec>`

Sets the timeout (in milliseconds) for individual commands executed from `ovdeploy` command. If this option is not set, the value specified for COMMAND_TIMEOUT under depl namespace of configuration settings of the target system will be used for individual commands (which has a default value of 10 minutes). This option applies to package install and remove commands: `-exec`, and `-cmd` commands.

`-targetrootdir <symbolic_name>`

A symbolic path name used to create an absolute file path for the target root directory.

`-targetdir <dirname>`

The name of the directory on the target node where the object is to be installed, removed, uploaded or downloaded, registered or unregistered.

`-targetid <id>`

The target ID of the target node.

`-pw <node_password>`

Use the `-pw` option to specify a login password for a node.

`-pw_prompt`

Use the `-pw_prompt` option to prompt for a login password.

EXAMPLES

The following examples show how to use the `ovdeploy` command:

- To install the package `testpackage.xml`, located in the local `/tmp` directory, to the remote node `test.com`.
  ```
  ovdeploy -install -pkg /tmp/testpackage.xml -node test.com
  ```

- To install the file `testfile`, located in the local `/tmp` directory, to the remote node `test.com`.

  ```
  ovdeploy -install -file /tmp/testfile -node test.com
  ```

- To deploy the file `testfile`, located in the local `/tmp` directory, to the `/opt/OV/bin` directory on the remote node `test.com`.

  ```
  ovdeploy deploy -file /tmp/testfile -targetdir /opt/OV/bin -node test.com
  ```

- To delete the file `/opt/OV/bin/testfile` from the host `test.com`.

  ```
  ovdeploy -remove -file testfile -targetdir /opt/OV/bin -node test.com
  ```

- To delete the file `$OvDataDir/installation/incoming/files/test/testfile` from the host `test.com`. The absolute path of the file is created from the specified target directory. No target root directory was specified, so the default target root directory is used.

  ```
  ovdeploy -remove -file testfile -targetdir test -host test.com
  ```

- To delete the package `testpkg1` from the host `test.com`.

  ```
  ovdeploy -remove -pkg testpkg1 -host test.com
  ```

- To copy the file `testfile` to the default target directory on the local host. The default target directory is `$OvDataDir/installation/incoming/files/`.

  ```
  ovdeploy -upload -file /tmp/testfile
  ```

- To copy all files that are specified in the package descriptor `package1.xml` to the default package upload directory on the host `test.com`. The directory containing the specified package descriptor file and all files specified in it is the bin directory.

  ```
  ovdeploy -upload -pkg package1.xml -sourcerootdir bin -host test.com
  ```

- To copy the files in the `/tmp/testdir` directory on the host `test.com` to the directory `/opt/OV/bin` on the local host.

  ```
  ovdeploy -download -dir /tmp/testdir -targetdir /opt/OV/bin -node test.com
  ```

- If `testpackage1` and `testpackage2` have been installed on the local machine, the following is returned:

  ```
  ovdeploy -inv

  NAME          VERSION    TYPE     ARCHITECTURE

  testpackage1  05.00.050  package  windows 4.0

  testpackage2   01.00.050   package  windows 4.0
  ```

- To display the native package inventory of the local host. For example:

  ```
  ovdeploy -inv -invtype native

  OpenView BBC Package 5.0.50
  ```

```
OpenView Performance Access Package 10.00.123
```

- To register the package `package1.xml` by copying the package descriptor to the inventory directory. If the package descriptor already exists, an error is returned to stdout.
  ```
  ovdeploy -reg -pkg /tmp/package1.xml
  ```
- To unegister the package `testpack2` from the host `test.com`.
  ```
  ovdeploy -unreg -pkgname testpack2 -host test.com
  ```
- To execute the file `run` on the local host and returns the output to stdout.
  ```
  ovdeploy -exec -file /tmp/run
  ```
- To execute the file `run.sh` with the shell on the host `test.com`.
  ```
  ovdeploy -exec -shell -file run.sh -targetrootdir bin -node test.com
  ```
- To set the timeout value for the `-exec` command for `my_exe.exe` file on host `node1.`
  ```
  ovdeploy -exec -file C:\my_exe.exe -node node1 -cmd_timeout 9000000
  ```
- To find out which operating system is running on the host `node1.`
  ```
  ovdeploy -get ostype -nodenode1
  ```
- To find the value set for the environment variable `OvInstallDir` on host `node1.`
  ```
  ovdeploy -env OvInstallDir -node node1
  ```
- To specify a login password for a node when configuring Operations Agent remotely.
  ```
  ovdeploy -cmd “%ovinstalldir%bin\win64\OpC\install\oasetup -configure -
  management_server mgmt_srv -certificate_server cert_srv” -node node1 -fem
  winservice -ostype Windows -user user1 -pw password
  ```
- To prompt for a login password when configuring the Operations Agent remotely.
  ```
  ovdeploy -cmd “%ovinstalldir%bin\win64\OpC\install\oasetup -configure -
  management_server mgmt_srv -certificate_server cert_srv” -node node1 -fem
  winservice -ostype Windows -user user1 -pw_prompt
  ```

ovconfpar

NAME

`ovconfpar` – set and return configuration parameters remotely

SYNOPSIS

```
ovconfpar -get [-host <hostname> [-targetid [<id>]...] -ovrg <OVRG> -ns <namespace>
]
```

```
ovconfpar -change [-host <hostname> [-targetid [<id>]...] -ovrg <OVRG>] -ns
<namespace> [ [-set <attr> <value>]... | [-clear [<attr>] ]... ]
```

```
ovconfpar -help
```

```
ovconfpar -version
```

DESCRIPTION

`ovconfpar` reads and sets configuration parameters for installed HPE Software components. For information about the parameters you can use with the `ovconfpar` command, see "Parameters". For information about the options you can use with the `ovconfpar` command parameters, see "Options".

Parameters

The `ovconfpar` command recognizes the following parameters:

`-get <options>`

Returns the value or values of one or more keys for the specified namespaces.

`-change <options>`

Sets different key-value pairs for multiple namespaces.

`-version`

Displays the version of the command.

`-help`

Displays the help information.

Options

You can use the following options with the `ovconfpar` command parameters:

`-host <hostname> [-targetid <id>]`

The host name and target ID of the remote machine.

`-ovrg <OVRG>`

If the parameter you want to get or change belongs to a resource group, use `-ovrg` to specify the name of the resource group.

`-ns <namespace >`

The name of the namespace, whose configuration parameters you want to get or change.

`-set <attr> <value> ...`

Set the named attribute to the specified value for the specified namespace.

`-clear [<attr>] ...`

Clear the named attribute(s) from the specified namespace. If no attribute is specified, all attributes are cleared for the specified namespace.

Return Codes

`ovconfpar` issues the following return codes:

`0` All steps were successful.

`-1` One or more steps failed.

EXAMPLES

The following examples show how to use the `ovconfpar` command.

- To set the key `ovo_port_range` to `12345` in the `ovo.server` namespace:

  ```
  ovconfpar -set -ns ovo.svr01 -set ovo_port_range 12345
  ```

- To set the key `ovo_port_range` to `12345` in the `ovo.svr01` and `ovo.svr02` namespaces:

  ```
  ovconfpar -set -ns ovo.svr01 -set ovo_port_range 12345 -ns ovo.svr02 -set ovo_
  port_range 12345
  ```

- To set the key `MaxFileSize` to `128` and the key `Protocol` to `HTTP` in the `ovo.svr01` namespace:

  ```
  ovconfpar -set -ns ovo.svr01 -set MaxFileSize 128 -ns ovo.svr01 -set Protocol
  HTTP
  ```

- To display all keys and their values for all namespaces:

  ```
  ovconfpar -g
  ```

- To display the value for `MaxFileSize` in the `ovo.svr01` namespace:

  ```
  ovconfpar -g -ns ovo.svr01 MaxFileSize
  ```

- To display the values in the `ovo.svr01` namespace:

  ```
  ovconfpar -g -ns ovo.svr01
  ```

ovappinstance

NAME

`ovappinstance` – return configuration parameters for application instances.

SYNOPSIS

```
ovappinstance -h | -help
```

```
ovappinstance -v | -version
```

```
ovappinstance -i | -instance <instance > {-st | -state} | {-h | -host} [-an | -
appNamespace < appNamespace >]
```

```
ovappinstance -is | -instances [-an | -appNamespace < appNamespace>]
```

```
ovappinstance -ai | -activeInstances [-an | -appNamespace < appNamespace> ]
```

```
ovappinstance -vc | -verifyConfig
```

DESCRIPTION

The ovappinstance command reads and displays the information contained in the APM XML configuration files. For information about the parameters you can use with the ovappinstance command, see "Parameters". For information about the options you can use with the ovappinstance command parameters, see "Options".

Parameters

The ovappinstance command recognizes the following parameters:

`-h | -help`

Displays the command parameters and options.

`-v | -version`

Displays the version of the command.

`i | -instance < instance >`

Returns information about the specified application instance.

`-is | -instances`

Returns information about all application instances found.

`- ai | -activeInstances`

Returns information about all application instances found to be up and running.

`- vc | -verifyConfig`

Checks and reports on the validity of the APM XML configuration file(s).

Options

You can use the following options with the ovappinstance command parameters:

`-st | -state`

Displays the outage state of the instance specified in `<instance>`.

`-h | -host`

Get either the virtual IP address of the instance `<instance>`. Alternatively, if the command is executed on a node, which is not configured as part of a high-availability cluster, get the FQDN or IP address of the local host.

`-an | -appNamespace`

Specify the name of the application namespace, whose information you want to display.

Return Codes

`ovappinstance` issues the following return codes:

`0` All steps were successful.

`1` One or more steps failed.

EXAMPLES

The following examples show how to use the `ovappinstance` command:

- To display a list of all application instances for a given application namespace:

  ```
  ovappinstance -instances -appNamespace < appNamespace >
  ```

- To display a list of all application instances which are active (or running) in a given application namespace:

  ```
  ovappinstance -activeInstances -appNamespace < appNamespace >
  ```

ovpolicy

NAME

`ovpolicy` – installs, manages, and removes both local and remote policies.

SYNOPSIS

```
ovpolicy -help
```

```
ovpolicy -version
```

```
ovpolicy -install [-host <hostname> [-targetid [< id >]...] {-enabled|-disabled} -
chkvers -add-category [<cat1>]... {-remove-category [<cat>]...
|-remove-all-categories} -force-cat -add-attribute [<name> <value>]... -remove-
attribute [<name> <value>]...| -remove-all-attributes
-force-attr -set-owner <owner> -force-owner -no-notify]
{-file [<file>]...|-dir [<dir>]...} [-ovrg <ov_res_group>]
```

```
ovpolicy -remove [-no-notify -host <hostname> [-targetid [<id>]...] [-ovrg <ov_res_
group>] <SELECTION>
```

```
ovpolicy [-enable |-disable] [-no-notify -host <hostname> [-targetid [<id>]...] [-
ovrg <ov_res_group>] <SELECTION>
```

```
ovpolicy [-addcategory |-removecategory] <cat>... [-no-notify -host <hostname> [-
targetid [<id>] [-ovrg <ov_res_group>]] <SELECTION>
```

```
ovpolicy -removeallcategories [<cat>]... [-no-notify -host <hostname> [-targetid
[<id>]...] [-ovrg <ov_res_group>]] <SELECTION>
```

```
ovpolicy [-addattribute |-removeattribute] <name> <value>... [-no-notify -host
<hostname> [-targetid [<id>]...][-ovrg <ov_res_group>]] <SELECTION>
```

```
ovpolicy -removeallattributes [-no-notify -host <hostname> [-targetid [<id>]...][-
ovrg <ov_res_group>]] <SELECTION>
```

```
ovpolicy [-setowner | -removeowner <owner>] [-no-notify -host <hostname> [-targetid
[<id>]...][-ovrg <ov_res_group>]] <SELECTION>
```

```
ovpolicy -notify [-host <hostname> [-targetid [<id>]...][-ovrg <ov_res_group>]]
```

```
ovpolicy -list [-level <0|1|2|3|4> -host <hostname> [-targetid [<id>]...][-ovrg
<ov_res_group>]]
```

DESCRIPTION

`ovpolicy` installs, manages, and removes local and remote policies. A policy is a set of one or more specifications rules and other information that help automate network, system, service, and process management. Policies can be deployed to managed systems, providing consistent, automated administration across the network. Policies can be grouped into categories, for example; to assign policies to a special policy group for simple enable and disable actions. Each category can have one or more policies. Policies can also have one or more attributes, an attribute being a name value pair.

You use `ovpolicy` command among other functions. This command is used to install, remove, enable, and disable local policies. For information about the parameters supported by the `ovpolicy` command, see "Parameters". For information about parameter options, see "Options".

Parameters

`ovpolicy` recognizes the following parameters:

`install`

Installs one or more policies using a single policy file specified with `-file` or multiple policy files specified with `-dir`.

`-remove`

Removes one or more policies.

`- enable`

Enables one or more policies.

`-disable`

Disables one or more policies. Note that the `-disable` option only disables a policy, it does not remove a policy from the file system.

`-addcategory`

Adds all category strings to the policy. You can add multiple categories using a blank-separated list.

`-removecategory`

Removes the specified category strings from the policy. You can remove multiple categories using a blank-separated list.

`-removeallcategories`

Deletes `all` categories.

`-addattribute`

Adds a category attribute to the policy. You can add multiple attribute names using a blank-separated list.

`-removeattribute`

Removes category attribute from the policy. You can remove multiple attribute names using a blank-separated list.

`-removeallattributes`

Deletes `all` category attributes.

`-setowner`

Sets the owner of a policy.

`-removeowner`

Removes the owner of a policy.

`-list`

Lists the installed policies.

`-notify`

Triggers any notifications to the control service, if there are any outstanding or suppressed notifications from previous policy operations.

`-version`

Displays the version number of the command.

`-h | -help`

Displays the help information.

Options

You can use the following options with the allowed `ovpolicy` command parameters:

`-add-attribute`

Add an attribute `<name>` with the value defined in `<value>` to the specified installed policy.

`-add-category` *`<cat1>`* `[`*`<cat2>`* `...` *`<catN>`*`]`

Adds all category strings to the policy. This is a blank-separated list.

`-chkvers`

Checks and compares the version of the already installed policy and the policy you want to install. If `-chkvers` is used, the new policy is not installed if the current installed version is the same or higher. If `-chkvers` is not used, the new policy overwrites the current policy with the same policy_id, regardless of the version number. `-chkvers` does not overwrite the categories, owner, or status of a current policy. To overwrite the categories, owner, and status associated with a policy owner, use `-forcecat`, and `-forceowner` respectively.

`-dir` *`<dirname>`*

If you specify a directory name, all policy files from that directory are used. A line is printed to stdout for each successfully installed policy.

`-enabled|-disabled`

If either `-enabled` or `-disabled` is used, the new policy acquires the status that is defined in the policy header. If neither `-enabled` nor `-disabled` is used, the new policy acquires the status of the currently installed policy (if any).

Note that this option overwrites the status defined in the policy-header installation file. So, if the new policy is already installed on the target system, the new version assumes the status of the installed version.

`-file <`*`filename`*`>`

Specifies a policy file name to be used. A line is printed to stdout for the successfully installed policy.

`-force-attr`

Allows you to remove category attributes that are set on a current installed policy. By default, the attributes from current installed policies are used. If there is no current installed policy, the attributes set in the header file of the new policy are used.

`-force-cat`

Allows you to remove categories that are set on a current installed policy. By default, the categories from current installed policies are used. If there is no current installed policy, the categories set in the header file of the new policy are used.

`-force-owner`

Overwrites the policy owner regardless of the settings for the installed policy.

`-host <`*`hostname`*`> [-targetid <`*`ids`*`>]`

This option specifies the hostname of the managed node. If no hostname is specified, the local host is assumed. `-targetid` specifies one or more target IDs.

`-level`

Specifies the type of information to be returned with the `-list` parameter, as follows:

0

Policy type, policy name, status, policy version. This is the default setting.

1

Policy type, policy name, status, policy version, policy_ID.

2

Policy type, policy name, status, policy version, policy_ID, category.

3

Policy type, policy name, status, policy version, policy_ID, category, owner.

4

Policy type, policy name, status, policy version, policy_ID, category, owner, attributes.

`-no-notify`

When `-no-notify` is used, `ovpolicy` does not trigger any notifications.

`-remove-category` *`<cat1>`* `[`*`<cat2>`*` ... `*`<catN>`*`]`

Removes the specified category strings from the policy. Using the `-remove-category` option with an empty string deletes all categories. This is a blank separated list.

`-remove-all-categories`

Removes the specified category strings from the policy.

`-remove-attribute`

Removes the category attribute `<name>` with the value defined in `<value>` from the specified installed policy.

`-remove-all-attributes`

Allows you to remove all category attributes that are set on a current installed policy. If there is no current installed policy, the attributes set in the header file of the new policy are used.

`-set-owner` *`<owner>`*

Sets the owner of a policy. `-set-owner` with an empty string deletes the owner.

`-ovrg` *`<ovrg_res_group>`*

Sets the name of the resource group.

`The <SELECTION>` option is one of the following:

`<SELECTION>-all|-owner <owner>|-owner <owner> -polname <name>|-polid <uuid> |-polname <[type:]name>|-poltype <typename>|-category <category> |-attribute <name> [value]`

`-all`

All installed policies.

`-owner <owner>`

The policy owner `<owner >`

`-owner <owner> -polname <name>`

The policy owner `<owner>` and the policy name `-owner <name>`

`-polid <id>`

The ID of the policy.

`-polname [<policy_type_name>:]<policy name>`

The name of the policy. If `policy_type_name` is used, the section applies to all policies of the specified type.

`-poltype <policy_type_name>`

The name of the type of policy.

`-category <category_name>`

The name of the category to be used.

`-attribute <name><value>`

The name of the policy attribute and value to be used.

`-targetovrg <ovrg_res_group>`

Sets the name of the resource group.

Return Codes

`ovpolicy` recognizes the following return codes:

`0` All steps were successful.

`1` One or more steps were not successful.

EXAMPLES

The following examples show you how to use the `ovpolicy` command:

- To list all policies on a node:
  `ovpolicy -list`
- To disable the HP-UX `syslog` policy:
  `ovpolicy -disable -polname "HPUX ovsyslog"`
- To enable all trap policies:
  `ovpolicy -enable -poltype ovsnmptrap`
- To install all policies located in the current working directory:
  `ovpolicy -install -dir`
- To install all policies located in the `/tmp/sap_policies` directory with a status of disabled:
  `ovpolicy -install -disable -dir /tmp/sap_policies`
- To reinstall all policies located in the `/tmp/xyz` directory, independent of the former owner:
  `ovpolicy -install -forceowner -dir /tmp/xyz`
- To remove all policies from the local host:
  `ovpolicy -remove -all`
- To remove all installed policies that are owned by the management server:
  `ovpolicy -remove -owner mgtsvr`

ovclusterinfo

NAME

`ovclusterinfo` – obtain information about clusters, cluster nodes, or high-availability (HA) resource groups.

SYNOPSIS

```
ovclusterinfo -h | -help
```

```
ovclusterinfo -v | -version
```

```
ovclusterinfo -a | -all
```

```
ovclusterinfo -c | -cluster {-ty | -type} | {-nm | -name} | {-st | -state} | {-nds
| -nodes} | {-rgs | -groups}
```

```
ovclusterinfo -n | -node < node > {-id} | {-st | -state}
```

```
ovclusterinfo -g | -group < group > {-id} | {-st | -state} | {-ls | -localState} |
{-nds | -nodes} | {-vip | -virtualIPAddress} | {-an | -activeNode}
```

DESCRIPTION

The `ovclusterinfo` command obtains information about high-availability clusters, and cluster nodes, and resource groups including; the name, status, and type of the cluster and, in addition, the nodes configured in the cluster. The `ovclusterinfo` command also obtains information about high availability (HA) Resource Groups, including; the status, IP address, and the nodes, which the Resource Group contains. An HA Resource Group is a collection of resources, such as files and processes that are available on one node in a cluster and can be switched to another cluster node as a single entity.

Parameters

The `ovclusterinfo` command accepts the following parameters:

`-h | -help`

Displays all options for the `ovclusterinfo` command.

`-v | -version`

Displays the version of the installed command.

`-c | -cluster`

Displays information about the named cluster.

`-a | -all`

Displays all available information about the named cluster, nodes, and resource groups.

`-n | -node`

Displays all available information about the named node in the cluster.

`-g | -group`

Displays information about the named high-availability resource group.

Options

You can use the following options with the appropriate command parameters:

`-ty | -type`

Displays the type of cluster which is installed. Possible values are:

- Microsoft Clustering Services (Windows),

- MC/ServiceGuard (HP-UX),

- VERITAS Cluster Server (Solaris),

- Sun Cluster (Solaris),

- Red Hat Advanced Server (RHAS),

- HACMP (AIX),

- Unknown.

`-nm |-name`

The name of the cluster.

`-st | -state`

The status of the cluster on the local node. This can be one of:

- Cluster is up

- Cluster is down

- State unknown

`-nds | -nodes`

Displays the names of the nodes in the cluster on separate lines. Cluster configuration determines how the node information is displayed, for example; short or long hostnames, IP address, and so on.

`-rgs | -groups`

All resource groups in the cluster.

`-status`

The status of the HA resource group, defined by `<rgname>`, on the local node.

`-virtualIPaddress`

The virtual IP address of the HA resource group, defined by `<rgname>`.

`-nodes`

The list of all nodes to which the HA resource group, defined by `<rgname>`, can fail over.

`-activeNode`

The node that currently hosts the HA resource group, defined by `<rgname>`.

EXAMPLES

The following examples show how to use the `ovclusterinfo` command:

- To display the name of the cluster:

  ```
  ovclusterinfo -cluster -name
  ```

- To display the names of all HA resource groups in the cluster:

  ```
  ovclusterinfo -cluster -groups
  ```

- To display the virtual IP address that is configured for the HA resource group haRG:

  ```
  ovclusterinfo -group haRG -virtualIPaddress
  ```

- To display the name of the node where the HA resource group haRG is currently running:

  ```
  ovclusterinfo -group haRG -activeNode
  ```

ovagtrep

NAME

`ovagtrep` – Enables configuration and control of the discovery agent and agent repository.

SYNOPSIS

```
            ovagtrep           [-clearAll] |

                        [-run <policy name>] |

                        [-publish]
```

DESCRIPTION

The discovery agent is an extension to the HTTPS agent, which runs service discovery policies that have been deployed from a management server. It stores the services that it discovers in the agent repository, which is a local datastore of services that exist on the node.

The agent synchronizes the services in the agent repository with the management server. The management server receives details of new, changed, and removed services only. Details of unchanged services are not resent.

The `ovagtrep` command enables you to configure and control the discovery agent and agent repository. It has the following options:

`-clearAll`

Clears all services from the agent repository. The next time that the discovery agent runs service discovery policies, it will recreate the services. The agent then synchronizes the services with the management server. This enables you to force the agent to synchronize unchanged services with the management server.

`-run <policy name>`

Runs a service discovery policy. Use this to run a policy at an unscheduled time, to discover any changes immediately. The agent sends details of changes to the management server. You can find the names of installed policies using `ovpolicy`.

`-publish`

Resends details of all the services that are currently in the agent repository to the management server. Use this for troubleshooting if services fail to appear on the management server.

The discovery agent and agent repository are part of a component that is registered with the control service. You can start and stop the component with the commands `ovc -start agtrep` and `ovc -stop agtrep`.

You can use the command `ovconfchg` to modify the following settings in the `agtrep` name space:

`ACTION_TIMEOUT <minutes>`

Sets the maximum number of minutes that a service discovery policy can run. If the policy runs any longer, the discovery agent stops running the policy and logs an error in the system log (`<data_dir>/log/System.txt`).

`INSTANCE_DELETION_THRESHOLD <value>`

Sets the number of times that service discovery policies must fail to discover existing services before the agent deletes the services from the agent repository.

If a service discovery policy can no longer discover a service that exists in the agent repository, the discovery agent deletes the service from the agent repository only after the service discovery policy has run the number of times that you specify with this setting.

For example, to set the action timeout to five minutes with the command `ovconfchg -ns agtrep -set ACTION_TIMEOUT 5`.

After you change the action timeout or instance deletion threshold, restart the component with the command `ovc -restart agtrep`.

opcmon

NAME

opcmon - forwards the current value of the monitored object to the monitor agent of the Operations Agent running on a local managed node.

SYNOPSIS

```
opcmon [ -help ] <object_name>[-<shortname>]=<value> [ -object <msg_object> ] [ -
option <variable>=<var_value> ]*
```

DESCRIPTION

The opcmon command forwards the current value of the monitored object to the monitor agent of the Operations Agent running on a managed node. The monitor agent checks this value against the configured threshold. According to the monitor configuration, the event is locally logged, suppressed, or forwarded to the message agent running on the managed node if the threshold is exceeded. The message agent forwards the message to the OM management server, where the message can be reviewed in the message browser.

The monitor agent checks the values received from monitored objects once in every two seconds. When the opcmon command forwards multiple values to the monitor agent between two successive checks, the monitor agent accepts only the last value for further processing.

If a local automatic command is set up to run when the threshold is exceeded, this command is immediately started by the local Operations Agent. The monitor agent must be configured and operating on the managed node, otherwise the opcmon command will fail.

Options

`-help`

Print usage message of opcmon. All other parameters are ignored.

<object_name>[-*<shortname>*]=*<value>*

Object name is the name of the measurement threshold policy. When the measurement threshold policy has been configured for multiple instance data, the short name is used to uniquely identify each instance within the policy.

-object*<msg_object>*

Value of the object text box which is part of an message. Setting the object with the opcmon can be used for the object monitoring.

`-option` *<variable>=<var_value>*

Sets the variable `$OPTION(<variable>)` to *`<var_value>`*. Within the message conditions this variable can be used to access the value passed with the `opcmon` call. Special characters must be escaped with a backslash "\".

Exit Values

This command exits with value zero (0) after successful execution. If something is wrong regarding the passed parameters, opcmon exits with value 2 and explains the problem on standard error. For other errors, the exit value is set to 1 and an appropriate error message is returned on standard error.

opcmsg

NAME

`opcmsg` - Generates messages that are forwarded to OM.

SYNOPSIS

```
opcmsg [ -help ] [ -id ] application=<application> object=<object name> msg_
text="<message_text>" [ severity=<severity label> ] [ msg_grp=<message_group> ] [
node=<node Name> ] [ service_id=<service name> ] [ -option variable=<value> ]*
```

DESCRIPTION

The `opcmsg` command generates a message for Operations Manager. Before the message is submitted, it is interpreted by the Message Interceptor on the local managed node where the command is executed. Depending on how you configure the message, the message can be:

- Discarded
- Locally logged
- Forwarded to the management server
- Forwarded to the management server, with local logging.

The behavior of messages depends on the configuration of interceptors (or opcmsg policies). A message may be created, or may be suppressed. For example, you might have a suppress condition in the `opcmsg` policy, which for example suppresses all messages with `application=Test`.

For example, the message for the following call will not appear in the message browser (the generated message will be suppressed):

```
opcmsg application=Test msg_text="Test message"
```

The message interceptor must be configured with at least one Open Message interface policy and be running on the managed node, otherwise the `opcmsg` command will fail.

Options

`-help`

Print usage message of `opcmsg`. All other options are ignored and no message is submitted.

`-id`

Return the message ID of the submitted message to stdout. This option also sets the `OPCDATA_REMARK_FOR_ACK` flag of the message, so that the manager information of the message is held by the message agent.

`severity=`*<severity label>*

Specifies the severity of the message. Following severities are supported: normal, warning, minor, major, critical. By default, normal severity is applied.

`application=`*<application name>*

Name of application (or script/program) that is affected by or has detected the event/problem.

`msg_grp=`*<message group>*

Default message group to which the message belongs. By default, no message group is assigned.

`object=`*<object name>*

Object which is affected by or has detected the event/problem.

`msg_text=`*<message text>*

Descriptive text explaining the event/problem in more detail.

`node=`*<node Name>*

System on which the event/problem is detected. By default the node name of the current system is applied.

`service_id=`*<service name>*

Name of the service (as defined in the Service Editor) to which the message is mapped.

`-option variable=`*<value>*

Sets the variable `$OPTION(variable)` to value. Within the message conditions this variable can be used to access the value passed with the `opcmsg` call.

Special characters must be escaped.

Exit Values

This command exits with value zero after a message is successfully generated; in case of an internal error, 1 is returned and an error message displays. If a syntax or usage error is detected, 2 is returned and an error message displays.

Restrictions

This command can be run by any user. The message group (`msg_grp`) parameter should not be longer than 32 bytes, because this is the maximum size OM can handle with this parameter.

Example

To submit a normal message issued when a user logs onto the system, you could set up the following scheduled task:

```
opcmsg appl=ScheduledTask obj=login severity=normal msg_g=Security msg_
t="%USERNAME% logged onto system %COMPUTERNAME%"
```

opcmack

NAME

`opcmack` - acknowledges messages created by the Operations Agent.

SYNOPSIS

```
opcmack [ -help ] message_id
```

DESCRIPTION

The `opcmack` command acknowledges messages created by the message agent of the Operations Agent.

The message commands received from the message agent on the local managed node are forwarded to the corresponding manager. If a message was previously sent to the management server, the message agent holds the information of the responsible management server in the memory.

Other operations on the message with this ID will be sent directly to this management server. After a specified time (default 1 hour), the message agent deletes this information to save memory and disk space; message operations will then be sent to all management servers.

On Windows, you must run the `opcmack` command with a user, who is a member of the Local Administrators group.

Options

`-help`

Print usage message of `opcmack`. All other options are ignored and no message will be acknowledged.

`-msg_id`

Specifies the message to acknowledge.

Utilities Provided by the Performance Collection Component

This section provides information on command-line utilities presented by the Performance Collection Component of the Operations Agent.

agsysdb

NAME

`agsysdb` - Performance Collection Component alarm generator system database manipulation program

SYNOPSIS

`agsysdb`

DESCRIPTION

`agsysdb` is a program that is used to list the contents of the Performance Collection Component alarm generator system database. The database contains information concerning all systems to which the alarm generator will send alert notifications. For more information, see "*Performance Alarms*" in the *Operations Agent User Guide*.

Options

`-ovo off |on`	Updates the option of whether or not to send alert notifications to OM. If this option is set to on, and if the Operations Monitoring Component is running on the node, all alert notifications will be submitted as a message to the Operations Monitoring Component. If this option is set to off, the alert notifications will not be sent to the Operations Monitoring Component. DEFAULT: on
`-add hostname`	Adds the SNMP management node to the alarm generator database. The hostname can be either a name or IP address.

`-delete hostname`	Deletes the SNMP management node from the alarm generator database. The hostname can be either a name or IP address.
`-delpv hostname`	Removes the Performance Manager 3.X system from the alarmgen generator database. The hostname can be either a name or IP address.

`-actions off | always | on`	Updates the option of whether or not to execute local actions. If this option is set to on, then the local actions defined in the EXEC statements in the `alarmdef` file will be executed if: 1) The Operations Monitoring Component is not running on the node. 2) The Operations Monitoring Component is running on the node, but the `ovo` option has been set to off. If this option is set to always, local actions will always be executed even if the Operations Monitoring Component is running. If this option is set to off, local actions will not be executed. If the Operations Monitoring Component is running, the local action will be submitted as a message to the Operations Monitoring Component. DEFAULT: on
`-l`	Lists where alert notifications will be sent. The "Last Error" fields will contain the last error that occurred when an alert notification was sent. If there was an error, see the `status.perfalarm` file for more details about the error.

Files

`/var/opt/perf/datafiles/agdb.*`

dsilog

NAME

`dsilog` - a program that logs incoming data

SYNOPSIS

`dsilog <logfile name> <class> [options]`

DESCRIPTION

`dsilog` is a program that logs incoming data. A separate logging process must be used for each class you have defined. The `dsilog` program expects to receive data from stdin.

Options

`logfile name`	Is used as a datasource name. If it is not in the current directory, the name must be fully qualified.
`class`	Is the name of the class to be logged.
`-i fifo`	Indicates that the input should come from the fifo named. If fifo is not used, input comes from stdin. If you use this method, start dsilog before starting your collection process. See man page mkfifo for more information about using a fifo.
`-timestamp`	Indicates that the logging process should not provide the timestamp, but use the one already provided in the input data. The timestamp in the incoming data must be in UNIX timestamp format (seconds since 1/1/70 00:00:00) and represents the local time (not Greenwich Mean Time).
`-c <char>`	Use the character as a string delimiter or separator.
`dsilog -vers`	Displays the version of this program.
`dsilog -?`	Displays options for this program. If your system interprets ? as a wildcard character, use an invalid option such as -xxx instead of -?.

Extract

NAME

Extract program is used to retrieve and analyze the historical data logged in the Operations Agent's datastore. The Extract program performs the export function. It reads data from datastore and exports the results to output files in the ASCII format.

SYNOPSIS

To export data:

```
extract -xp [d|w|m|y] [-gapdzcnituyhxGADZNITUYHX] [-f
<outputfilename>] [-r <filename>] [-b <date> <time>] [-e <date>
<time>]
```

To select DSI data to export:

```
-C classname [DETAIL|SUMMARY|BOTH]
```

> **Note:** The SUMMARY and BOTH options are only functional when performing a data export. The extract functionality does not support data summarization.

To display more details about the extract program parameters:

```
man extract
```

Or

```
extract ?
```

DESCRIPTION

To retrieve data from the datastore run the Extract program using the command line mode. The syntax for the command line interface is similar to standard UNIX command line interfaces on other programs.

> **Note:** During an upgrade from Operations Agent version 11.xx to Operations Agent 12.05 the older data that is the datastored in the CODA database files, SCOPE log files, and the DSI log files will be retained in read-only mode. Extract program will be able to read data from both old and new datastore.

Options

`-b <date> <time>`	Sets starting date and time
`-B <EPOCH time>`	Sets starting date and time in UNIX/EPOCH format
`-e <date> <time>`	Sets ending date and time
`-E <EPOCH time>`	Sets ending date and time in UNIX/EPOCH format
`-r <reportfile>`	Specifies export template file for export formats
`-f <file> <fopt>`	Sends extracted data to a specific output file. If not specified, export data goes to default files `xfr*logfilename.ext`
`-C <classname>`	Selects DSI (data source integration) data to export.

`-gapdzcnituyhxGADZNITUYHX`	Selects types of data to extract/export
	g = global detail
	a = application detail
	p = process detail
	d = disk device detail
	z = logical volume detail
	c = configuration detail
	n = netif detail
	t = transaction detail
	u = CPU detail
	y = filesystem detail
	h = Host Bus Adapter (HBA) detail

	i = logical system detail
	Note: logical systems detail is not supported on Windows and Linux platforms.
	x = core detail
	G = global summary (Export only)
	A = application summary (Export only)
	D = disk device summary (Export only)
	Z = logical volume summary (Export only)
	N = netif summary (Export only)
	`I` = logical systems summary
	Note: logical systems summary is not supported on Windows and Linux platforms.
	T = transaction summary (Export only)
	U = CPU summary (Export only)
	Y = filesystem summary (Export only)
	`H` = HBA summary
	X = core summary

`-xp <xopt>`	Exports data
`?`	Displays command line syntax

where:

`<date>`	Specifies a date in native language syntax. (The default format is MM/DD/YY, as in 12/31/03.)
`<time>`	Specifies a time in native language syntax. (The default format is hh:mm AM or hh:mm PM, where hh is 12-hour hours, mm is minutes)
`<reportfile>`	Names an ASCII template file that defines output data fields and format for the EXPORT command. The default template file is `/var/opt/perf/reptfile`.
`<file>`	Specifies an output file name for EXPORT. (See the OUTPUT command for default values.)

For Example:

To export detail global data, run the following command:

```
extract -g -r /var/opt/perf/myrept -f myout -xp
```

The global data is exported to the output file named myout. The output file contains all global metrics specified in the myrept export template file.

For Example:

1. To export the detail global data starting from June 5, 2014 8:00 am and ending at June 5, 2014 5:00 pm, run the following command:

   ```
   extract -g -b "6/5/14 8:00" -e "6/5/14 17:00" -f myout -xp
   ```

   The global data is exported to the output file named myout. Since no export template file is specified, the default export template file, reptfile, is used. All global metrics specified in reptfile are included in the output file.

2. To export the detail global data starting from June 5, 2014 8:30:20 am and ending at June 5, 2014 5:30:20 pm, run the following command:

   ```
   extract -g -b "6/5/14 8:30:20" -e "6/5/14 17:30:20" -f myout -xp
   ```

   The global data is exported to the output file named `myout`. Since no export template file is specified, the default export template file, `reptfile`, is used. All global metrics specified in `reptfile` are included in the output file.

For more details on Extract program see, the *Operations Agent User Guide.*

glance

NAME

`glance` - GlancePlus system performance monitor for UNIX/Linux

SYNOPSIS

`glance` [`-j` interval] [`-p` [dest]] [`-f` dest] [-command]

[`-maxpages numpages`] [`-nice nicevalue`] [`-nosort`] [`-lock`]

[`-adviser_off`] [`-adviser_only`] [`-bootup`]

[`-iterations` count][`-align`] [`-syntax` filename]

[`-aos` filename [-noscaling]]

[`-all_trans`] [`-all_instances`] [`-no_fkeys`]

DESCRIPTION

GlancePlus is a powerful and easy-to-use online performance diagnostic tool for UNIX/Linux systems. It is distributed in two forms: "xglance", a Motif-based program, and "glance", a character mode program. The combination allows you to use the appropriate tool for your job. With `xglance` you get the power and ease-of-use a Motif-based tool. With `glance` you can run on almost any terminal or workstation, over a serial interface and relatively slow data communication links, and with lower resource requirements. Either component provides the same rich set of performance information.

The default Process List screen provides general data on system resources and active processes. More specific data is presented via the CPU, Memory, Disk IO, Network, NFS, Swap, and System Table screens. Process workload groups, or applications can be viewed in the Application List screen. Specific per-process detail is also available via the individual process screens. Running in a terminal environment, Glance aids performance problem resolution on all Linux systems.

Definitions and descriptions of each metric are provided in GlancePlus online help.

Options

`-j interval`	This option allows you to preset the number of seconds between screen refreshes in place of the default value of 5 seconds. For example, if -j 60 is passed in, the screen update interval would be preset to 60 seconds.
`-p [dest]`	This option indicates that the continuous `print` option should be enabled when the tool starts. This can be useful for printing screens automatically over a long interval. The output will be directed to the default lp device, unless the dest parameter is supplied. Once GlancePlus is running, another `p` command toggles the continuous print off.
`-f dest`	This option indicates that the continuous `print` option should be enabled when the tool starts. This can be useful for printing screens automatically over a long interval. The output will be directed to the specified destination file. Once GlancePlus is running, another `p` command toggles the continuous print off.
`-maxpages numpages`	This option changes the maximum number of pages that can be printed with the `p` command. The default maximum is 200 pages.
`-command`	This option can be used to request a different initial screen than the Global Summary screen. This startup option corresponds to the keystroke commands that display different detail screens once the tool is run. Only one of the commands in the first section of the COMMAND SUMMARY (below) is allowed for this option.
`-nice nicevalue`	This option allows you to set the nice priority value for the GlancePlus process. The default nice value is -10.
`-nosort`	This option tells GlancePlus not to sort interesting processes listed on the Global Summary screen. This reduces Glances CPU overhead.
`-lock`	This option allows Glance to lock itself into memory. Note that response

	time may be enhanced by using this option, but there is the possibility that you will receive the error "Unable to allocate memory/swap space". If this happens, you must run GlancePlus without using this option.
`-adviser_off`	Allows you to run Glance without the Adviser.
`-adviser_only`	This option allows Glance to run with no screen displays to the terminal. Only the Adviser will run, sending its output to stdout. With this option the GlancePlus Adviser can be run in the background, with stdout output optionally redirected to a file. If you want to run GlancePlus in "Adviser only" mode at startup time, you must include the `-bootup` option as well.
`-bootup`	This option allows Glance to ignore the SIGHUP signal. Use this option along with -adviser_only or -aos when you want to run GlancePlus in "Adviser only" mode at startup time.
`-iterations count`	This option allows you to limit the number of intervals that Glance will run. This can be used in conjunction with the `-adviser_only` option which enables GlancePlus to run in the background with no terminal screen displays. Glance will execute for the number of iterations specified and then terminate.
`-align`	This option aligns the screen update interval to 1 minute if the glance update interval is set to more than or equal to 60 seconds. If the glance update interval is less than 60 seconds, aligns the screen update interval to the interval boundary. This option can be used only with the -adviser_only mode.
`-syntax filename`	Use this option to specify a file name which contains the syntax to be used by the Adviser. If no syntax file is specified, the Adviser will search for a user default file, `~/adviser.syntax`. If no user syntax file is found, the system default syntax file, `/var/opt/perf/adviser.syntax`, will be used.
`-aos filename`	Use this option as an alternative to `-adviser_only -syntax` filename option.
`-noscaling`	This option is used to turn off scaling of metric value to appropriate units like kb/mb/gb. This can only be used in conjunction with adviser_only mode.
`-all_trans`	This option allows GlancePlus to display all transactions registered on your system. If not specified, GlancePlus only displays transactions filtered by the values specified in your threshold file.
`-all_instances`	This option allows GlancePlus to display the most recent 2048 instances in a transaction. If not specified, GlancePlus only displays active instances with no stop time.
`-no_fkeys`	This option disables the display of function key labels.

COMMAND SUMMARY

The following commands are grouped into three sections; top level screens, secondary screens, and miscellaneous commands. The top level screen commands are the only ones allowed on the command line.

Command	Screen Displayed / Description
a	CPU by Processor
c	CPU Report
d	Disk Report
g	Process List
i	File System Capacity
l	Network by Interface
m	Memory Report
t	System Tables Report
u	IO by Disk
w	Swap Space
A	Application List
F	Process Open Files
N	NFS Global Activity
R	Process Resources
M	Process Memory Regions
Z	Global Thread List
I	Thread Resources
G	Process Thread List
T	Transaction Tracking
H	Alarm History
?	Commands Menu
S	Select an Application / Transaction / Logical
V	Select a Logical System List
K	Select a Logical System Report
s	Select a single process

b	Scroll page backward
f	Scroll page forward
h	Online help
j	Adjust refresh interval
o	Adjust process threshold
p	Print toggle
q	Quit GlancePlus
r	Refresh the current screen
<cr>	Update current screen
y	Renice a process
z	Reset statistics to zero
>	Display next logical screen
<	Display previous screen
!	Invoke a shell

EXAMPLES

- To choose the default start up options, which start glance in the Global Summary screen, updating statistics every five seconds, enter:

  ```
  glance
  ```

- To monitor system resource usage overnight, run glance, and print the screen once every hour until you return and exit the program, enter:

  ```
  glance -j 3600 -p
  ```

- To monitor and print swap utilization once every two hours, specifying the lp2 destination printer, enter:

  ```
  glance -j 7200 -p lp2 -w
  ```

- To execute GlancePlus at a higher than normal priority, limit the maximum number of pages output by the print command to 10, and request that the interesting processes be unsorted, enter:

  ```
  glance -nice -19 -maxpages 10 -nosort
  ```

midaemon

NAME

midaemon - Performance Measurement Interface daemon.

SYNOPSIS

midaemon [options}

DESCRIPTION

The Measurement Interface daemon, midaemon, provides an interface between ARM Transaction Tracking and the Performance Collectors. This program translates trace data into Measurement Interface counter data using a memory based MI Performance Database to hold the counters. This database is accessed by collector programs such as glance, xglance, gpm, and oacore.

The Measurement Interface daemon, midaemon, must be executed as root or with the set-user-id bit set to root. Attempting to run the **midaemon** process without user-id of root results in an immediate termination.

The midaemon backgrounds itself when started. Status and errors are written to the file:

```
/var/opt/perf/status.mi
```

COMMAND LINE OPTIONS

midaemon recognizes the following command line options:

-?	Displays the mode dependent available options on standard error.
-bufsize <value>	The **midaemon** process uses bufsize buffers to communicate with ARM Transaction Tracking. This option changes the buffer default value. The default value is defined by experience and validation tests. Avoid changing this value unless the performance tools or the daemon process itself report losing buffers. If the passed value is less than 4096, the value is reset to the default value for performance reasons. DEFAULT: 131072 bytes.
-debug <level>	Enables or disables the midaemon debugging mode. Possible debugging level values are: 0 - disables all debugging levels 1 - enables the first (lowest) level of debugging 2 - enables the second (medium) level of debugging

	3 - enables the third (highest) level of debugging. The debugging information is related to the midaemon activity and to the collectors requests and is printed in the `status.mi` file. DEFAULT: 0 [off].
`-fg`	Allows the **midaemon** process to execute in foreground. This option should only be used when debugging. DEFAULT: off.
`-ignore_mt` **Note:** This option is present only on HP-UX. On other platforms, core-based accounting is achieved by configuring the `ignore_mt` parameter in the **parm** file. See *the Operations Agent User Guide* for more information.	Enables the kernel instrumentation to provide core-based accounting. This option is useful on a system where hyper-threading is enabled. Providing this option enables CPU-related metrics to be normalized against the number of active cores on the system. If this option is not provided, CPU-related metrics are normalized against the number of hardware threads (logical CPUs) on the system.
`-k`	Sends a termination request to an active **midaemon** process. This will cause the running midaemon to release the MI Performance Database and to exit. If performance collectors are still active and attached to the MI Performance Database the active midaemon will ignore the termination flag and will continue to run. If this daemon process was in the debugging level 1, a message describing the attempt to terminate is written to the `status.mi` file. If a **midaemon** process has been killed because of a SIGKILL signal, the `-k` option can be used to delete the existing MI Performance Database. DEFAULT: off.
`-K`	Sends the no_permanent and termination requests to an active **midaemon** process. This will cause the running **midaemon** process to become no_permanent, to release the MI Performance Database and to exit. This option is as an alias of the `no_pk` options. DEFAULT: off.
`-mlock`	Specifies the locking of the MI Shared Memory Performance Database in memory. By default, MI does not lock the database and pages in only the active memory pages, creating less intrusiveness on the system memory utilization. DEFAULT: off.
`-no_mlock`	Tells the **midaemon** process not to lock the MI Shared Memory Performance Database in physical memory. In control mode, this request

	can be sent to the active **midaemon** process to unlock the database. DEFAULT: on.
`-no_p`	Sends the no_permanent request to the **midaemon** process. This means that when the last performance tools will exit, the **midaemon** process will exit by releasing the MI Performance Database. DEFAULT: off.
`-normal_prio`	Specifies that the **midaemon** process starts in the normal scheduling priority. DEFAULT: off. WARNING: This option must not be used if ARM instrumented data is collected. If this option is used when ARM instrumented data is collected, the transaction throughput handled by **midaemon** is affected. The local zones runs in normal priority. On Solaris local zones, the default value is ON and it cannot be overridden. **midaemon** cannot be moved to real time priority inside local zones as priocntl does not work due to privilege issues inside minimum configured local zones.
`-p`	Specifies that the **midaemon** process runs permanently, even if no performance tools are attached to the Measurement Interface. To stop this behavior, the control mode -no_p request has to be sent to the active daemon. To stop a permanent daemon, the requests -no_p, -k or -K should be used. DEFAULT: on.
`-rtprio <priority>`	Specifies the process real-time priority to be used for the **midaemon** process. The default value was chosen by experience and tests with other system daemons. DEFAULT: Real time priority. On Solaris local zones, real-time priority for **midaemon** is disabled.
`-sizes`	Specifies that the **midaemon** process writes the sizes of the enabled Classes of the MI Shared Memory Performance Database in the `status.mi` file. DEFAULT: off.
`-smdvss <value>`	Specifies the maximum virtual set size of the MI Shared Memory Performance Database. This option limits the amount of memory used by the database and should be used to limit the dynamic expansion of the performance Classes. The default size is kernel dependent. The `-sizes` option can be used to determine the MI Performance Database sizing values. DEFAULT: kernel dependent.
`-timeout <value>`	Specifies that the **midaemon** process sets a specific timeout value for the kernel instrumentation interface. Do not modify the default value, unless in a

	debug situation. DEFAULT: 300 milliseconds
`-T`	Specifies that the active **midaemon** process terminates immediately without any regard to attached performance tools. This option is for use only during software installation or removal procedures. DEFAULT: off.
`-udts <value>`	Specifies the maximum number of UDT entries for the ARM Transaction Tracking data class in the MI Performance Database. DEFAULT: 20.
`-V`	Prints the midaemon version to standard out.

MI ERROR MESSAGES

In case of errors, midaemon was designed to use well defined exit values and to write explicit error messages in the `/var/opt/perf/status.mi` file. The error messages are formatted:

– name of the running midaemon program - timestamp,

– name of the routine generating the error - error message,

– error message generated by a perror(3C) call if a system call has failed. This information may be of use when reporting problems.

EXAMPLES

- The standard **midaemon** process execution is:

```
% midaemon
```

- To specify the MI Shared Memory Database size up to 1 Megabyte at the MI initialization:

```
% midaemon -smdvss 1M
```

WARNINGS

The midaemon program is automatically executed by performance tools such as GlancePlus (glance or xglance) or Performance Collection Component (oacore). However, it can be executed manually to customize the MI Performance Database or to send to the active **midaemon** process specific requests.

If the creation of the `status.mi` file failed, midaemon will use the error file `/tmp/status.mi`.

ovpa

NAME

ovpa - Performance Collection Component script for starting and stopping data collection and alarms

SYNOPSIS

ovpa [action] [subsystem] [parms]

DESCRIPTION

ovpa is a script that is used to start, stop, and re-initialize Performance Collection Component processes.

ACTION

`-?`	Lists all ovpa options. If your shell interprets ? as a wildcard character, use an invalid option such as -xxx instead of -?.
`start`	Start all or part of Performance Collection Component. (default)
`stop`	Stop all or part of Performance Collection Component.
`restart`	Reinitialize all or part of Performance Collection Component. This option causes some processes to be stopped and restarted.
`status`	List the status of all or part of Performance Collection Component processes.
`version`	List the version of all or part of the Performance Collection Component files.

SUBSYSTEM

all	Perform the selected action on all Performance Collection Component. (default)
scope	Perform the selected action on the **oacore** collector. The restart operation causes the **oacore** collector to stop, then restart. This causes the **parm** and **ttd.conf** files to be re-read.
server	Perform the selected action on the Performance Collection Component. This affects the **oacore** daemon as well as the alarm generation subsystem. The restart operation causes coda to terminate and restart. This causes the datasources and alarmdef files to be re-read.
alarm	Perform the selected action on the Performance Collection Component. Restart is the only valid option and causes the alarmdef file to be reprocessed.

PARMS

-midaemon *<miparms>*	Provide the midaemon with parameters to initiate it with other than default parameters. If the -midaemon parameter is used, it must be the last parameter in the list. All remaining parameters are passed to the **midaemon** process.

ovtrap

NAME

ovtrap - Script which will issue an SNMP trap to a node

SYNOPSIS

ovtrap [-s severity] host alarm_message

DESCRIPTION

ovtrap generates an SNMP V1 trap to the Event Browser. This capability may be used through a local action within the alarm syntax, but normally if you would like all alarms to automatically generate SNMP traps, you would use the appropriate option to agsysdb.

If you are not using the automatic trap option, you may call ovtrap directly from your alarmdef syntax via an EXEC statement. In this case, you must supply the options. Host is either an Internet address or a host name. The alarm_message must be no more than 128 characters. And the optional -s parameter can be one of the following: Normal, Minor, Warning, Major, or Critical.

EXAMPLE

ovtrap -s Warning monitoring_system "This is the message"

oacore

NAME

oacore - Performance collection daemon

DESCRIPTION

oacore is a daemon process that runs on a Performance Collection Component-monitored system. It is invoked by the ovpa script. oacore logs performance data that is read by the Performance Collection Component. oacore can serve as a data collector. The user-controlled configuration file **parm** is used to control oacore’s logging. The perfstat script can be used to check the status of performance collection daemon processes.

SDLCOMP

NAME

`sdlcomp` - a program that checks and compiles the class specification file

SYNOPSIS

```
sdlcomp specification_file [logfile_set [logfile-name]] sdlcomp -max- class number
specification_file logfile_set [logfile-name] sdlcomp [options]
```

DESCRIPTION

`sdlcomp` checks the class specification file for errors. If no errors are found, it adds the class and metric descriptions to the description file in the logfile set you name.

Options

`specification_file`	The name of the file containing the class specification. If it is not in the current directory, it must be fully qualified.
`logfile-name`	The logfile in the set that will contain the data for this class. If the datafile named does not exist, it is created. If a datafile with this name exists to hold other classes, the new class is added to it. If no datafile is named, a new datafile is created for the class and automatically named. Only classes with unlimited CAPACITY must be in separate datafiles.
`sdlcomp -vers`	Displays the version of this program.
`sdlcomp -?`	Displays options for this program. If your system interprets ? as a wildcard character, use an invalid option such as -xxx instead of -?.

SDLUTIL

NAME

`sdlutil` - a program you use to manage DSI data and class information

SYNOPSIS

`sdlutil logfile name` [options]

DESCRIPTION

`sdlutil` is a program which allows you to list or view class or metric information, statistics for classes, and version information. You can also remove classes and data from a datasource and recreate a class specification from the information in the datasource using this utility.

Options

`logfile name`	Is the name of the datasource created by compiling a class specification.
`-rm all`	Removes datasource, classes, metrics and data from the Metrics Datastore.
`sdlutil -vers`	Displays version information.
`sdlutil -?`	Displays options for this program. If your system interprets ? as a wildcard character, use an invalid option such as -xxx instead of -?.

UTILITY

NAME

`utility` - is a program for managing and reporting information on the collection parameters (**parm**) file and the alarm definitions (`alarmdef`) file.

SYNOPSIS

To scan the datastore file and produce a report on its contents:

```
utility -xs [-v] [-dD] [-f <filename>]
```

To check the syntax of the parm file:

```
utility -xp <parmfile> [-v] [-f <filename>]
```

To check the syntax of an alarm definitions file:

```
utility -xc <alarmdef> [-f <filename>]
```

> **Note:** The resize command `-xr` is available only for backward compatibility when you upgrade from Operations Agent 11.xx to Operations Agent 12.05.

DESCRIPTION

`-f <file>`	Sends output to a specific output file.
`-D`	Enables detail for scan, analyze, and **parm** file checking
`-d`	Disables detail for scan, analyze, and **parm** file checking

`-v`	Selects verbose output
`-xp <parmfile>`	Syntax checks a parameter file.
`-xc <alarmdef>`	Syntax checks an alarm definitions file and sets the alarmdef file name.
`-xa`	Analyzes the data stored in the datastore against the `alarmdef` file.
`-xs <logfile>`	Scans the datastore and produces a report.
`?`	Displays command line syntax

where:

`<parmfile>`	Specifies a parameter file; can be fully qualified with a path name. (The default parm file is **parm**.)
`<alarmdef>`	Specifies an alarm definitions file; can be fully qualified with a path name. (The default alarm definitions file is `alarmdef`.)

For Example

To scan datastore, run the following command:

```
utility -D -xs
```

For more information on utility program see, the *Operations Agent User Guide*.

xglance

NAME

`xglance` - GlancePlus system performance monitor for UNIX/Linux

SYNOPSIS

```
xglance [-nosave] [-rpt [reportname]] [-sharedclr] [-nice nicevalue] [-lock]
[Xoptions]
```

DESCRIPTION

GlancePlus is a powerful and easy-to-use online performance diagnostic tool for UNIX/Linux systems. The xglance program provides both graphical and textual information for system administrators and others who need help troubleshooting a performance problem. Advanced alarm and adviser capabilities make it an effective monitoring tool as well.

Options

<table>
<tr><td><code>-nosave</code></td><td>This option overrides <code>xglances</code> default of saving a users configuration at the next exit. If you would like to ensure that a particular user enters xglance in the same state at each start up, include the <code>-nosave</code> option in the xglance startup script.</td></tr>
<tr><td><code>-rpt reportname</code></td><td>This option allows you to specify one or more additional report windows to be displayed when you start xglance. By default, GlancePlus displays the windows that were open at the last xglance exit. Following are the various report names that may be used for reportname:
<table>
<tr><td>AlarmHistory</td><td>ApplicationCPUGraphs</td><td>ApplicationList</td></tr>
<tr><td>CPUByProcessor</td><td>CPUGraph</td><td>CPUReport</td></tr>
<tr><td>DiskGraph</td><td>DiskQueueGraphs</td><td>DiskReport</td></tr>
<tr><td>FileSystemCapacity</td><td>IOByDisk</td><td>Main</td></tr>
<tr><td>MemoryGraph</td><td>MemoryReport</td><td>MemoryUsageGraph</td></tr>
<tr><td>NetworkByCardGraph</td><td>NetworkByInterface</td><td>NetworkGraph</td></tr>
<tr><td>NfsByOperation</td><td>NfsGlobalActivity</td><td>ProcessList</td></tr>
<tr><td>ResourceHistory</td><td>SwapSpace</td><td>SymptomHistory</td></tr>
<tr><td>SymptomStatus</td><td>SystemAttributes</td><td>SystemTablesGraph</td></tr>
<tr><td>SystemTablesReport</td><td>TransactionTracking</td><td>ThreadList</td></tr>
</table></td></tr>
<tr><td><code>-sharedclr</code></td><td>This option causes xglance to use a shared color scheme. While it disables the ability to configure colors within xglance, it allows xglances colors to be configured along with other applications from a central point and leaves private color cells available to other applications. Even without this option, xglance will use a shared color scheme if it fails to obtain private color cells.</td></tr>
<tr><td><code>-nice nicevalue</code></td><td>This option allows you to set the nice priority value for the <b>xglance</b> process. The default nice value is -10.</td></tr>
<tr><td><code>-lock</code></td><td>This option causes xglance to lock both its text and data segments into memory. Note that response time may be enhanced with this option, but there is also the possibility that you will receive the error "Unable to allocate memory/swap space". If this happens, you must run GlancePlus without using this option.</td></tr>
<tr><td><code>Xoptions</code></td><td>The xglance program accepts the standard X Toolkit options. Examples of commonly used options are "-iconic" (start up iconified), "-bg color" (use specified background color), and "-display xdisplay" (display on specified X server). Note that "-fg color" is not accepted, since the window foreground color is computed by xglance from the background color. Also note that since help text defaults to black via a resource set in <code>/var/opt/perf/Gpm</code>, a dark</td></tr>
</table>

	background should not be used unless the help text resource is also set to a light color so it will be readable.

EXAMPLES

- To run xglance on a display named "sparc10a", enter:

  ```
  xglance -display sparc10a:0.0
  ```

- To run xglance so that it does not automatically save configuration changes on exit, enter:

  ```
  xglance -nosave
  ```

- To require shared color usage and add the DiskReport window to those displayed by xglance at startup, enter:

  ```
  xglance -sharedclr -rpt DiskReport
  ```

ttd

NAME

`ttd` - transaction tracking registration daemon.

SYNOPSIS

`ttd` [options]

DESCRIPTION

The transaction tracking daemon, ttd, reads and registers transaction definitions from the configuration file `/var/opt/perf/ttd.conf`. The ttd also assigns IDs to the transaction names passed to it via arm_getid calls that come from the ARM library. The ttd synchronizes these transaction definitions with the Measurement Interface daemon process, **midaemon**.

You must execute the registration daemon, ttd, as root or with the set-user-id bit set to root. ttd runs in background mode when dispatched, and errors are written to the error file: `/var/opt/perf/status.ttd`.

COMMAND LINE OPTIONS

`ttd` recognizes the following command line options:

-?	Displays the available options on standard error.
-hup	Tells a running **ttd** process to reread the configuration file without explicitly killing and

	restarting. The `-hup` option should be used in conjunction with the `-mi` option to synchronize the midaemon with the ttd. To have the changes synchronized with the midaemon, specify `-mi` after `-hup`.
`-fc`	Disables the processing of the `ttd.conf` configuration file during the start of **ttd** daemon.
`-fg`	Starts **ttd** as a foreground process instead of running in background.
`-k`	Kills the **ttd** process. The **ttd** process should only be killed if the **midaemon** process is likewise killed. Killing and restarting the **ttd** process without stopping the midaemon can result in unsynchronized TT data in the **midaemon** process.
`-mi`	Tells a running **ttd** process to synchronize its entries with the midaemon without explicitly killing and restarting. This option is typically used after ttd -hup to reread the `ttd.conf` file and synchronize any changes with the midaemon.

ERROR MESSAGES

`ttd` was designed to use exit values and to write explicit error messages in the `status.ttd` file. The error messages are formatted using these conventions:

- name of the running `ttd` program - timestamp,

- name of the routine generating the error - error message,

- error message generated by a perror(3C) call if a system call has failed.

DEPENDENCIES

The ARM library registration calls arm_init() and arm_getid(). The control call arm_stop (...,ARM_ ABORT,...) fails if the ttd daemon is not running. However, ARM operations arm_start() or arm_stop() can execute successfully without the **ttd** daemon process if arm_getid was successfully called before ttd was stopped.

The Measurement Interface processing daemon midaemon must also be running to process the user-defined transactions and to measure performance metrics associated with these transactions.

The localhost loopback interface must be configured in order for **ttd** to receive the client RPC connection requests. Client applications connect to ttd via RPC connections when the arm_getid() function is called.

The number of active client processes that can register transactions with ttd via the arm_getid() call is limited to the maxfiles kernel parameter. This parameter controls the number of open files per process. Each client registration request results in **ttd** opening a socket (an open file) for the RPC connection. The socket is closed when the client application terminates; therefore, this limit affects only the number of active clients that have registered a transaction via the arm_getid call. Once this limit is reached, **ttd** will return TT_TTD- NOTRUNNING to a clients arm_getid() request. The maxfiles kernel parameter can be increased to raise this limit above the number of active applications that will register transactions with **ttd.**

EXAMPLES

- The standard **ttd** process execution is:
  ```
  % ttd
  ```
- To signal an active **ttd** daemon process to reread the configuration file and synchronize with the **midaemon** process:
  ```
  % ttd -hup -mi
  ```
- To stop an active **ttd** daemon process:
  ```
  % ttd -k
  ```

EXTENSIONS

An active **ttd** daemon process stores its pid in the file: `ttd.pid` located in the same directory as the `status.ttd` file.

Utilities Provided by the RTMA Component

This section provides information on command-line utilities presented by the RTMA component of the Operations Agent. You can access real-time system performance data from the monitored system by using these commands.

perfd

NAME

`perfd` - Multi-platform system performance metric server

SYNOPSIS

```
perfd [options]
```

DESCRIPTION

`perfd` is a system performance daemon that provides real-time access to system performance metrics, locally or remotely. For all but the global metric class, `perfd` provides data for the last interval only. For global data, `perfd` can provide average, minimum, maximum values as well as standard deviation, based on a configurable collection depth.

COMMAND LINE OPTIONS

The options are:

`-c directory`	This option specifies an alternate configuration directory. `perfd` will change its working directory to the specified location and attempt to load additional configuration options from perfd.ini. If a **parm** file is not present in the specified directory, `perfd` will use the system-wide file in the official configuration directory. The default configuration directory is platform dependent.
`-C`	Check the configuration file and exit. If the `-c` option is specified, the configuration in that directory is checked.
`-d depth`	This option specifies the number of intervals for which global metric values are kept. By default, `perfd` keeps 5-minute historical data for all global (single-instance) metrics.
`-f`	This option helps to debug and will cause `perfd` to run in the foreground instead of running it in the background.
`-i interval`	This option specifies the data collection frequency. The default is 10 seconds.
`-l`	If this option is present, `perfd` will not collect data for processes, application, NFS operations, logical systems, or ARM. Additionally, on HP-UX, HBA and LVM data will not be collected either.
`-p port`	This option specifies an alternate port. The default registered port number for `perfd` is 5227.
`-r maxrps`	This option specifies the maximum number of requests that a given thread is allowed to send per second. If the limit is exceeded, the server will pause for one second and `perfd` will log this information in the log file. The default limit is 20.
`-s`	This option causes the server to deny all requests except those coming from the host system through the loopback interface (localhost). Denied connection requests are logged.
`-t maxtpc`	This option specifies the maximum number of threads per client system. The default number is 30. If the number is exceeded, the connection request is denied. Denied connection requests are logged.
`-x maxcps`	This option specifies the maximum number of connections the server will handle per second. The default is 2. If the number of connection requests exceeds this value, the server will pause for 3 seconds before establishing the connection.
`-4`	This option causes `perfd` to accept only IPv4 connections. Note that, by default, `perfd` will automatically switch to IPv4-only if it is unable to create

	an IPv6 socket. Thus this option should only be used if you want to explicitly disable IPv6 .
`-?`	Print option list.

FILES

All options that can be specified at run-time can also be placed in the configuration file:

On Windows:

```
%ovdatadir%\perfd.ini
```

On UNIX/Linux:

```
/var/opt/perf/perfd.ini
```

Options specified on the command line take precedence over the options specified in the configuration file. Each line in the file specifies a run-time option. Characters after the pound sign (#) are ignored.

cpsh

NAME

`cpsh` - Cross-platform performance shell

SYNOPSIS

`cpsh` [options]

DESCRIPTION

This program allows a performance expert to display glance metrics from any system where the perfd daemon is running. If no options (with exceptions mentioned below) are present, cpsh will run in the interactive mode, otherwise it will run in the batch mode.

COMMAND LINE OPTIONS

The options are:

`-c class`	This option specifies the metric class (category) being requested. The default class is `gbl`, also known as GLOBAL in the Performance Collection Component. Short names or Performance Collection Component class names (like DISK, APPLICATION, and so on) are allowed. See the classes command in the EXAMPLES of the interactive mode below.
`-C subclass`	This option must be used with the `-c` option above and specifies the metric

	subclass (subcategory) being requested with that class.
`-d`	This option causes the metric class tree and the full metrics dictionary (all available metric classes and the metric names within the classes) to be printed.
`-f filter`	This option is used to specify a filter for a given metric class. A filter has the form "<metric> <operator> <value>". See EXAMPLES below.
`-h header`	This option specifies the header type. The argument can be 0 (no header), 1 (two-line header) or 2 (two-line header with interval spacing). The default is one. See the `-s` and `-t` options also. This option will not turn off the interactive mode.
`-H`	This option causes `cpsh` to print the metric help text for the specified metric(s) in the specified class and optional subclass, and then exit. If no metric/class/subclass are specified, the default global metric list is used. If a class and optional subclass are specified, the default metric list for that class/subclass is used (see FILES below). Options other than metric class (es) and lists do not affect the output. These options, if present, are validated but ignored.
`–i iterations`	This option specifies the number of iterations to be performed. The default is one iteration. A zero iteration count can be used to request an unlimited number of iterations.
`–I instance`	This option is used together with the `-c` and `-C` options to specify an instance when requesting data for a metric subclass.
`–m metrics`	This option specifies the list of metrics being displayed. If this option is not present, the default list is taken from configuration files.
`–n system`	This specifies which system to get performance data from. The system name may be in the form `system:port` where port is the port `perfd` is listening on. This is one of the options that will not cause `cpsh` to start in the batch mode; it can be used to start an interactive session on a remote system.
`-N`	This option is only meaningful when used with one of the summarization options. By default, when requesting summarized data, the interval is aligned on the appropriate minute/second boundary. If this option is specified, the alignment will not be performed.
`–o optfile`	This option enables you to specify an option file. All the options mentioned in this topic can be entered in an option file to facilitate simpler preset batch runs.
`–r`	This option is used to request the raw data. This only applies to a small subset of metrics.
`–s`	This option enables you to specify field separators. By default, fields are horizontally separated by a space, vertically with spaces (if a header type one or two is specified using the `-h` option above). Special characters must be quoted or escaped (with a backslash). If the argument is the default CSV separator, the output will be CSV (Comma-Separated Values) with the metric names as headers (single-line headers). If CSV is requested, -h 2 is ignored.

`-t`	This option specifies that the output should be printed using a table format combining multi-line header and special separators. It is equivalent to specifying a header type 2 and |+- field separators (vertical separator is |, horizontal separator is - and cross-line separator is +). This option will not turn off the interactive mode.
`–v`	Verbose mode. This option will cause `cpsh` to print the system type and `perfd` server information when running in the batch mode.
`–W`	Normally, when running in the batch mode, `cpsh` will remove trailing spaces. If this option is present, spaces at the end of the lines will not be removed.
`-z summinterval`	This option specifies the summarization interval to be used when displaying summarized data. By default, the maximum summarization configured on the `perfd` server is used. The `-v` option can be used to obtain information about the server configuration. Note that summarization is only available for the global (single-instance) data.
`–Z summtype`	This option specifies the summarization type desired. The argument may be one of AVG (average), MIN (minimum), MAX (maximum), STDDEV (standard deviation), or ALL (all values). The argument is not case sensitive.
`–?`	Print the option list and defaults.

OPTION FILE

The following list of options can be specified in a `cpsh` option file:

```
class=<valid class string>

subclass=<valid subclass string>

filter=<valid filter string>

header=<valid numerical header value>

iterations=<valid numerical iterations value>

instance=<valid numerical instance value>

metrics=<valid metrics string>

system=<valid system string>

noalign=<true/false>

optfile=<valid optfile string>

raw=<true/false>

nostrip=<true/false>

separator=<valid separator string>
```

```
fancy=<true/false>

ipv4=<true/false>

verbose=<true/false>

summinterval=<valid numerical summinterval value>

summtype=<valid summtype string>
```

This list will be printed if an invalid option is specified in the option file and if the -v flag is present.

FILES

Default metrics can be specified for each metric class. If a metric list is not specified at the command line (or in the option file), cpsh will attempt to find a default using the following search order:

<InstallDir>`/perfd/system/`*<system name>/<metric class>*

<InstallDir>`/perfd/os/`*<os type>/<metric class>*

<InstallDir>`/perfd/default/`*<metric class>*

<*DataDir*>`/perfd/os/`<os type>/<*metric class*>

<*DataDir*>`//perfd/default/`*<metric class>*

On HP-UX, Linux, Solaris, and AIX, <*DataDir*> is `/var/opt/perf`.

On Windows, the default <*DataDir*> is `%ovdatadir%`.

<*system name*> is the name of the system, <*os type*> is the operating system type as returned by glance (available only on UNIX/Linux) in the metric `GBL_OSNAME`.

METRIC LISTS

The `perfd` server enables clients to combine metrics when appropriate: global (gbl) and table (tbl) metrics may be added to any metric list, and parent class metrics may be added to subclass metrics. To add metrics from a different class, the appropriate metric prefix must be used. For example, the following command will print the global node and operating system name metrics, all application-level utilizations for application index number 3, then process IDs and all the process-level 'name' metrics for each process that is currently alive in application number 3:

```
cpsh -c app -C proc -I 3 -m 'gbl*name app*util proc_proc_id proc*name'
```

The following command will print the time, the application name for application index number 3, then the process IDs, names, and CPU utilization for all processes currently alive in application number 3:

```
cpsh -n itill -c app -C proc -I 3 -m "gbl_stattime app_name proc_proc_id proc_proc_
name proc_cpu_total_util"
```

The metric lists for base classes and subclasses are different. For instance, the list of process metrics is different for a full process list (such as in the 'proc' command) and a process list for an application (such as in the 'app 1 proc' command). The same configuration file is used in both cases, and unavailable metrics are silently ignored. For example, specifying the list `proc*name proc_proc_id app*util` in <*InstallDir*>`/perfd/default/proc` will cause the 'proc' command to display all process names and the process ID, but the 'app 1 proc' command will additionally display the application's utilization metrics. When running in the interactive mode, modifying a class metric list does not affect the subclass, and modifying a subclass metric list does not affect the class metric list.

RAW METRICS

By default, all metrics are displayed in the same format as in glance: time stamps show date/time, rates can show Kb, Mb, and so on. If the `-r` option is specified, some metrics, like the ones just mentioned, will be displayed in the raw format. Interpreting the raw data requires knowledge about the metric types and is not recommended for general use.

EXAMPLES

The following are examples of the batch mode invocations.

Print all global metrics whose names match the pattern:

```
cpsh -c gbl -m "gbl_nodename *cpu*util" -n test123

Node CPU Idle Intrpt Nice Phys System User Wait

 Name Entl % CPU % CPU % CPU % CPU % CPU % CPU % CPU % CPU %

 test123 4.8 95.4 0.2 0.0 4.8 2.2 4.8 2.6 0.0
```

Print open files for **init** process on the system test123:

```
cpsh -n test123 -c proc -C pfile -I 1

 Open File File

 PID Mode Type Name

 1 rd/wr fifo /dev/initctl
```

Print default global metrics on current system in the table format:

```
cpsh -c gbl -t

+----------------+--------+--------+------+------+------+------+------+

|Node | Time| CSwitch| | Load| Peak|Pg Req| |

|Name | Stamp| Rate| CPU %| Avg|Disk %| Rate|Swap %|

+----------------+--------+--------+------+------+------+------+------+
|system1 |09:18:15| 260.7| 5.3| 0.0| 0.9| 30.1| 40.0|
+---------------+-------+-------+-----+-----+-----+-----+------+
```

INTERACTIVE MODE

If no arguments are present (except for the exceptions noted above), `cpsh` will run in the interactive mode. In that mode, the user can connect to any system where `perfd` is running and request metrics from that system.

Typing a question mark and a carriage return will print a list of the available commands as well as a few examples.

Note that all interactive commands may be piped through any UNIX command (for instance, more, grep, and so on).

The main commands are:

```
system [name]
```

Without arguments, this command displays information about the current system. If an argument is provided, it is the name of a system that `cpsh` will attempt to connect to or switch to if a connection has already been established. Note that just entering the system name will switch to an already active system as well.

Note that any given system can be connected to in multiple ways, using its simple name, its fully qualified name, its IP address, the loopback, and so on. `cpsh` will attempt to create a single connection for a unique system:port combination. If an alternate name is provided, that name will be shown in parenthesis. For example, if system1 was first used, then localhost, the systems command shows:

```
system1 (localhost) - 1-way 9000/800 64-bit HP-UX B.11.11, up 29 days 07:04
```

```
systems
```

This command takes no arguments. It will print a list of the systems `cpsh` is currently connected to with an asterisk to the left of the system currently active.

```
server
```

This command is similar to the system command above, but it prints configuration information for the perfd server instead of system information.

```
servers
```

Also similar to systems, but prints information about the perfd servers.

class *[metric class] [metric subclass]*

Without argument, this command prints the current metric class. If valid class/subclass names are provided as argument, `cpsh` will switch to that class/subclass.

`classes`

This command takes no arguments. It will display the full list of classes and subclasses available from the `perfd` server currently active.

<metric class> [instance ID] [<metric subclass>]

Typing any base metric class (for instance, gbl or bydsk or proc) will display the values for the list of chosen (or default) metrics. If the metric class is multi-instance (like bydsk or proc), data for all the instances will be displayed.

If the class is multi-instance, you can use an optional instance ID and, in this case, only data for the specified instance is printed. For instance, `proc 1` will print data for process 1 only. It will also select instance 1 as the default instance.

If a metric subclass is specified, the data for that subclass will be printed. For instance, `proc 1 pfile` will print the open files for process 1. If an instance has been established as the default, the instance ID can be skipped in further requests. So, after typing `proc 1`, the `proc byregion` command will print the memory regions for process 1.

`init, add, del[ete]`

These commands initialize, add to, or delete from the metric list for the selected class and the selected system. The arguments are a list of metric names or matching patterns. The modified metric list is preserved until you exit the `cpsh` prompt.

`list [all]`

This command prints a list of the metrics currently selected. If the optional `all` argument is provided, all the available metrics for the current class are printed.

`push`

Once the system, metric class, and metric list are all established, this command will instruct `cpsh` to print new metric values as soon as they become available in `perfd`. The push process can be interrupted using the configured interrupt key (typically control-C).

`help`

If no arguments are present, this command is equivalent to the `?` command and prints a summary of the commands and some examples. If you specify an argument (which is a metric name in the current metric class or in the global class), `cpsh` will print the help text associated with the metric. perfd servers can only provide help text for the metrics that are enabled on the server.

`mdict`

Print the entire metric dictionary. Similar to the batched `cpsh -d` command.

```
filter [<filter value> or "disable"]
```

Without argument, this command will print the current filter for the current metric class on the current system. If the argument is the keyword `disable`, the current filter is disabled; otherwise, the arguments should be a valid filter expression.

```
summ [seconds]
```

If the current metric class supports summarization, this command will print the summarized data for the established metric set. If an additional argument is present, it is a number of seconds to be used instead of the default summarization configured in the `perfd` server. If a non-numeric `seconds` argument is provided, it is silently ignored.

```
wait
```

This command will cause `cpsh` to pause until new data is available from the currently selected `perfd` server.

```
exit
```

You can exit the `cpsh` prompt by typing this command.

padv

NAME

`padv` - perfd adviser

SYNOPSIS

padv [options]

DESCRIPTION

This program allows a performance expert to run glance adviser scripts on remote systems where the perfd daemon is running.

COMMAND LINE OPTIONS

The options are:

-h	This option tells `padv` to run the adviser script on the historical data before using the real time data (by default, the `perfd` server keeps 5 minutes of historical data for global metrics). Messages resulting from exceeded threshold or alerts in the historical data will be terminated with "minus N seconds" where N is the number of seconds before

	the current time stamp. If the script issues print statements, `padv` will print a delimiter after the print statements resulting from historical data, to indicate that it is switching to real time data. For all other statements, historical data can be easily identified by the "minus" terminators. Adviser scripts referencing non-global data are not allowed and will result in an error.
`-i iterations`	This option limits the number of iterations to be performed. By default, that value is 0 (zero), which means a continuous run. This count specifies the number of iterations on the real time data (iterations based on historical data, if -h is present, are not included).
`-n system`	This specifies which system to run the adviser script on. The system name may be in the form `system:port` where port is the port `perfd` is listening on.
`-s script`	This option specifies which script to run.
`-S`	If this option is present, each symptom defined in the script is displayed along with its value (probability) at each pass.
`-?`	Print option list and defaults.

FILES

If a script is not specified, `padv` will use the appropriate default script if one is present. The default script name is adv and the search order is:

<Install_Dir>`/perfd/system/`*<system name>*`/adv`

<Install_Dir>`/perfd/os/`*<os type>*`/adv`

<Install_Dir>`/perfd/default/adv`

<Data_Dir>`/perfd/os/`*<os type>*`/adv`

<Data_Dir>`/perfd/default/adv`

<system name> is the name of the system, *<os type>* is the OS type as returned by `glance` in the metric GBL_OSNAME (currently one of the following: AIX, HP-UX, Linux, NT, or SunOS).

EXAMPLES

The `padv -S -i1` command will return the current values for the default 4 bottlenecks on the current system:

```
Symptom 0: CPU bottleneck = 0.00%

Symptom 1: Disk bottleneck = 0.60%

Symptom 2: Memory bottleneck = 0.00%

Symptom 3: Network bottleneck = 0.00%
```

The `padv -S -i 1 -n system1` command will return the current values for the same default 4 bottlenecks on the system system1:

```
Symptom 0: CPU bottleneck = 0.00%

Symptom 1: Disk bottleneck = 0.00%

Symptom 2: Memory bottleneck = 0.00%

Symptom 3: Network bottleneck = 0.00%
```

CAVEATS

Running complex scripts involving nested loops (for instance summing up all the virtual sizes of all memory regions in all processes) may provide inconsistent data if the requests cannot be completed in a single `perfd` interval and may even result in script termination. For such scripts, using glance is recommended. Scripts with one level of loop (or no loops) are guaranteed to return data for the same interval.

Process-level system calls are not supported in remote adviser scripts. For such scripts, glance should also be used.

mpadv

NAME

`mpadv` - perfd multi-system adviser

SYNOPSIS

`mpadv` [options]

DESCRIPTION

This program allows a performance expert to run glance adviser scripts on multiple systems where the `perfd` daemon is running at the same time. Note that, in order to limit the potentially high amount of output, `mpadv` ignores all print statements from the adviser syntax file. It is recommended that remote scripts that rely on print statements be run on a single system using `padv`.

COMMAND LINE OPTIONS

The options are:

`-c`	If this option is present, time stamps will reflect the time on the system where `mpadv` is running, not the remote systems.

-h	This option tells mpadv to run the adviser script on the historical data before using the real time data (by default, the perfd server keeps 5 minutes of historical data for global metrics). Messages resulting from exceeded threshold or alerts in the historical data will be terminated with "minus N seconds" where N is the number of seconds before the current time stamp. Adviser scripts referencing non-global data are not allowed and will result in an error.
-i iterations	This option limits the number of iterations to be performed. By default that value is 0 (zero), meaning a continuous run. This count specifies the number of iterations on the real time data (iterations based on historical data, if -h is present, are not included).
-l list	This specifies a file name containing a list of systems to run the adviser script(s) on. The system names may be in the form system:port where port is the port perfd is listening on, one system per line with optional comments after a pound sign.
-r	This option tells mpadv to keep trying if a system is not available when the program is invoked or if a system goes down as the program is running.
-s script	This option specifies which script to run. See the FILES section in the "padv" on page 114 topic for default script and locations. If the list of system include multiple platforms (operating systems) and if a script is specified, the script must contain metrics that are common to all platforms.
-t threshold	This option specifies a threshold value above which bottleneck probabilities will be printed even if the values do not result in an alert. The default threshold is 70, meaning that any bottleneck probability equal or above 70% will be printed. If a threshold above 100 is provided, only alerts contained in the adviser script(s) will be printed.
-v	This option causes mpadv to produce more verbose output, such as connection information.
-?	Print option list.

FILES

See the "padv" on page 114 topic for default file name and locations. If a script is not provided (with the -s option) and if the list of systems contains multiple platforms, the appropriate OS default file applies to each system.

EXAMPLE

The `mpadv -l ~/stage/config/systems -v -r -t 101` command will print all alerts on all the systems in the 'systems' list, connection and dropped connection messages.

```
Starting to monitor system1, Fri Feb 8 10:21:48 2008

Starting to monitor system2, Fri Feb 8 10:21:48 2008

Starting to monitor system3, Fri Feb 8 10:21:48 2008

Starting to monitor system4, Fri Feb 8 10:21:48 2008
Starting to monitor system6, Fri Feb 8 10:21:48 2008
Starting to monitor system7, Fri Feb 8 10:21:48 2008
Starting to monitor test-system2, Fri Feb 8 10:21:49 2008
Starting to monitor test-system3, Fri Feb 8 10:21:49 2008
Starting to monitor test-system4, Fri Feb 8 10:21:49 2008
Starting to monitor test-system1, Fri Feb 8 10:21:49 2008
Starting to monitor test-system5, Fri Feb 8 10:21:49 2008
Starting to monitor test-system6, Fri Feb 8 10:21:49 2008
Starting to monitor test-system7, Fri Feb 8 10:21:49 2008
Starting to monitor test124, Fri Feb 8 10:21:50 2008
Connection to system7 lost: Connection reset by peer, Fri Feb 8 10:43:18 2008
Starting to monitor system7, Fri Feb 8 10:43:29 2008
Connection to system1 lost: Connection reset by peer, Fri Feb 8 11:49:52 2008
Connection to system4 lost: Connection reset by peer, Fri Feb 8 11:50:06 2008
Starting to monitor system1, Fri Feb 8 11:50:53 2008
Starting to monitor system4, Fri Feb 8 11:50:57 2008
test124 : YELLOW Disk Bottleneck probability= 78.60%, 02/09/08 01:15:55
test124 : END End of Disk Bottleneck Alert, 02/09/08 01:17:55
test124 : YELLOW Disk Bottleneck probability= 71.40%, 02/09/08 01:30:30
test124 : END End of Disk Bottleneck Alert, 02/09/08 01:30:50
system7 : RED Disk Bottleneck probability= 95.20%, 02/11/08 01:02:05
system7 : END End of Disk Bottleneck Alert, 02/11/08 01:03:15
system7 : YELLOW Memory Bottleneck probability= 85.00%, 02/12/08 05:39:25
system7 : END End of Memory Bottleneck Alert, 02/12/08 05:39:45
system3 : YELLOW CPU Bottleneck probability= 82.00%, 02/12/08 14:08:35
system3 : END End of CPU Bottleneck Alert, 02/12/08 14:10:45
```

Utilities Provided by the RTM Component

This section provides information about configuration settings for HPCS using the `hpcs.conf` file.

hpsensor

NAME

hpsensor - Compute Sensor, a light weight system performance viewer component

DESCRIPTION

hpsensor is a system performance daemon that provides real-time access to system performance metrics, locally or remotely through secure channels. For all metrics class, **hpsensor** provides data for the last interval only.

> **Note:** The **rtmd** process is replaced with the **hpsensor** process. The XPL configurations for `rtmd` are not backward compatible and will not work after you upgrade from Operations Agent 11.xx toOperations Agent 12.05. The **hpsensor** process provides similar XPL configurations to enforce security (SSL).

CONFIGURATION OPTIONS

You can edit the following internal configuration settings of HPCS using the `hpcs.conf` file:

> **Note:** The `hpcs.conf` file is located at:
>
> **On Windows:**
>
> `%OvDataDir%\hpcs\`
>
> **On UNIX/Linux:**
>
> `/var/opt/OV/hpcs/`

Configuration Options for hpsensor

Variable	Namespace	Description	Restart Required	Default Value
port	`hpcs.runtime`	This parameter enables you to modify the default port number.	YES	Any available port

Configuration Options for hpsensor, continued

collection_ Interval	`hpcs.runtime`	This parameter enables you to modify the default frequency of the collection interval. Specify the value in seconds.	YES	10
num_threads	`hpcs.runtime`	This parameter enables you to modify the default number of threads.	YES	5
connnection_ backlog	`hpcs.runtime`	This parameter enables you to modify the default connection backlog.	YES	On Windows: 16384 On Linux: 128
regBBC	`hpcs.runtime`	To register hpcs component with BBC, set this variable to TRUE.	YES	TRUE

Debug_Level	`hpcs.trace`	This parameter enables you to modify debug level value. You can use one of the following debug level values:
		INFO/WARN/ERROR/DEBUG/ALL
		The `hpcstrace.log` file are created at:
		On Windows:
		`%OvDataDir%\hpcs\`
		On UNIX/Linux:
		`/var/opt/OV/hpcs/`
			YES	INFO

<Target Server>	`hpcs.hbp`	This parameter enables you to push heart beat information to a Target Server. **Syntax:** Add the following to send heart beat information to the <Target Server> http://<Target Server IP>:<Target Server Port>/hpcsrv/hbphandler=/lwiregistry/up In this instance: <Target Server IP> Is the IP address of the Target Server. You can also use the host name. <Target Server Port> The port where	YES	

Configuration Options for hpsensor, continued

		hpcsrvd (HPSC Server) is listening.		
interval	`hpcs.hbp`	This parameter enables you to modify the default HBP interval. Specify the value in seconds.	YES	30

> **Note:** You must restart `hpsensor` after changing the configurations.
>
> Run the following command to restart `hpsensor`:
>
> ```
> ovc -restart hpsensor
> ```

Watchdog Mechanism for hpsensor

The **Watchdog** mechanism monitors the **hpsensor** process. Watchdog runs once in every hour and logs the status of the **hpsensor** process in the `hpcswatch.log` file located at:

On Windows:

```
ovdatadir/hpcs/hpcswatch.log
```

On UNIX and Linux:

```
/var/opt/OV/hpcs/hpcswatch.log
```

While installing Operations Agent, the Watchdog mechanism adds a crontab entry on UNIX and Linux systems and a schedule task entry on Windows systems.

If the **hpsensor** process is not running, it executes the following command to start **hpsensor**:

```
ovc –start hpsensor
```

The `hpcswatch.log` file is rolled over when it reaches a maximum size of 1MB. During roll over the existing file is replaced with a new `hpcswatch.log` file.

Chapter 4: Configuration Variables of the Operations Agent

Your use of the *Configuration Variables of the Operations Agent* section is subject to the license terms for the Operations Agent software.

You can modify the default behavior of the Operations Agent by configuring different variables available with the components of the Operations Agent. You must perform the configuration steps to modify the default settings of these variable only with the `ovconfchg` command.

To modify the default setting of a variable, follow these steps:

1. Log on to the Operations Agent node with necessary privileges.
2. Run the following command:

   `ovconfchg -ns` <*namespace*> `-set` <*variable*> <*value*>

   In this instance:

   <*namespace*>: The namespace information of the variable (see Table 4).

   <*variable*>: The name of the variable.

   <*value*>: The value that you want to assign to the variable.

3. If necessary, restart the agent processes for the changes to take effect. See Table 4 to identify the variables that do not need a manual restart of agent processes after modification. To restart the agent processes, run the following commands:

   a. `ovc -kill`

   b. `ovc -start`

To switch back to the default setting of a variable, follow these steps:

1. Log on to the Operations Agent node with necessary privileges.
2. Run the following command:

   `ovconfchg -ns` <*namespace*> `-clear` <*variable*>

   In this instance:

   <*namespace*>: The namespace information of the variable.

<*variable*>: The name of the variable.

Alternatively, to bring all the variables back to their default settings, run the following command:

```
ovconfchg -ns<namespace> -clear -all
```

Configuration Variables for the Operations Monitoring Component

The Operations Agent presents a vast set of variables that you can configure using the `ovconfchg` command to change the default behavior.

The following table lists configuration variables provided by the Operations Monitoring Component of the Operations Agent:

Table 4a List of Configuration Variables for the Operations Monitoring Component

Variable	Description	Restart Required	Default Value	Type
FAILED_ COLLECTION_ RETRIES	Specifies whether startup of a failed collection should be restarted for an Advanced Monitor Policy. Possible values: integers. Special values: **0:** No retries **-1**: The failure is ignored by the agent and the policy does not go into the failed state	NO	3	Integer
FAILED_ POLICY_TIME_ TO_ REACTIVATE	The wait time before the policy restarts its operation after a failure can be specified with this variable. The time is specified in hours. Use 0, if no policy restart is required.	YES	24	Integer
IPADDR_ CHECK_ INTERVAL	Time (in seconds) between two consecutive checks for IP address change (for DHCP).	YES	1800 (30 minutes)	Integer
LIMIT_NBR_ PARALLEL_	If the value of LIMIT_NBR_PARALLEL_	YES	FALSE	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
ACTIONS	ACTIONS is TRUE, the action agent will consider the value specified for MAX_NBR_PARALLEL_ACTIONS. If the number of running actions reaches the value specified for MAX_NBR_PARALLEL_ACTIONS, the action agent will wait for any of those running actions to complete before scheduling the remaining actions. Note that you have to set a suitable value for MAX_NBR_PARALLEL_ACTIONS depending on the number and interval of policies and execution time of scripts if you are using this variable.			
MAX_NBR_PARALLEL_ACTIONS	The maximum number of automatic actions that can run on the node simultaneously. The Windows operating system restricts the number simultaneous actions (any type of actions: automatic or manual) on a system. Therefore, on a Windows agent node, the number of simultaneous automatic actions is either the number of available free slots for actions, or MAX_NBR_PARALLEL_ACTIONS, whichever is smaller.	YES	25	Integer
MAX_RETRIES_UNTIL_POLICY_FAILED	This number specifies how often a policy should try to collect data. This is important for usage with external program sources. If the external program has a problem, the policy should not stop operating immediately. Therefore, a policy can stop and retry the external data collection process when it fails to collect data from the external source. With this variable, you can specify the number of retries to be attempted by a policy. Use 1 if no retries should be made.	YES	3	Integer
OPC_ACTAGT_LOGGING	Enable the action agent to log data on the agent node. By default, output is written into the file `opcaalog` in the agent log	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	directory.			
OPCMONA_ERRORMSG_ONLY_OPCERROR	If set to TRUE, error messages ranging from OpC30-3400 to OpC30-3409 are not sent to the OM console but logged in the agent trace.	YES	FALSE	Boolean
OPC_ACTION_CHARSET	To let the **opcacta** automatically get the system charset, set this variable to SYSTEM. To set to a specific charset, say acp1252, set this variable to acp1252.	YES	_	String
OPC_AGENT_ID	Managed node identifier, which is known on agent and server; used to identify messages and action requests. Introduced to support DHCP environments.	YES		String
OPC_AGTKILL_TIMEOUT	*UNIX only.* Time for a complete agent shutdown (`opcagt -kill`); after the specified time, agent processes are killed with -9.	YES	120	Integer
OPC_AGTSTOP_TIMEOUT	Time (in seconds) for an interceptor process to shut down.	YES	4	Integer
OPC_AGT_PROCESS_PRIORITY	*Windows only.* Change the priority of the agent processes. This now defaults to "Below Normal" (Windows 2000). Possible settings: • ABOVE • NORMAL • BELOW • IDLE	YES	BELOW	String
OPC_AGTMSI_ALLOW_AA	Allows an MSI instance to create or modify messages with automatic actions.	YES	FALSE	Boolean
OPC_AGTMSI_ALLOW_OA	Allows an MSI instance to create or modify messages with operator-initiated actions.	YES	FALSE	Boolean
OPC_AGTMSI_ENABLE	Allows an MSI instance to access OM data streams.	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_AVOID_SEGMENT_NAMES	If the variable is set, segment names (any name ending with .Segment<*number*>) will not be resolved, but the name service cache returns NULL.	YES	FALSE	Boolean
OPC_BUFLIMIT_ENABLE	Enable/disable checking of the buffer file limit on the agent node. Checks are applied on the `msgagtdf` file.	YES	FALSE	Boolean
OPC_COMPRESSION_DISABLE	Enable/disable OM data compression for network transfer.	YES	FALSE	Boolean
OPC_COND_FIELD_ICASE	Comparison for the object, application and message group fields are not case sensitive when this variable is set to TRUE.	YES	TRUE	Boolean
OPC_CONNECT_SRV_ONLY_IF_DATA	If this variable is set to TRUE, the managed node connects to its management server(s) after agent startup only if data is present.	YES	FALSE	Boolean
OPC_DISABLE_MSGGRP_OVERRIDE	If this variable is set to TRUE, the category/message group in the message will not get replaced with that of the SNMP/CMIP events received.	YES	FALSE	Boolean
OPC_DISABLE_NODE_OVERRIDE	If TRUE, the node override variable of a trap is not evaluated. This may avoid name service accesses for unresolvable names.	YES	FALSE	Boolean
OPC_DISABLE_SEVERITY_OVERRIDE	If this variable is set to TRUE, the severity level in the message will not be replaced with that of the SNMP/CMIP events received.	YES	FALSE	Boolean
OPC_DYNAMIC_LOGFILE_ONCE	If this is set to TRUE, dynamic evaluation of logpaths in the logfile encapsulator is done only after startup or policy distribution.	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_EVENT_RETRY_OLDEST	Set the number of retries attempted by the logfile encapsulator to read an event when EventLog fills up very fast. When EventLog fills up very fast, the recent Events can be overwritten before the **opcle** can process them. To keep up with the fill rate of EventLog, **opcle** must skip the overwritten Events and start at the current end of EventLog. **opcle** does retry this for the configured number of times.	YES	30	Integer
OPC_EC_STREAM_POLICY	ECS event handling policy. The ECS engine creates output if ALL or if ANY circuits create output for an event. Values: OUTPUT,UNSPECIFIED, DISCARD	YES	OUTPUT	String
OPC_ENFORCE_PASSWORD_CHECK	Enforces a switch user for each user running a command on the managed node through the action agent. This covers tools and scheduled task policies.	YES	FALSE	Boolean
OPC_EVENT_RUNTIME_ONLY	Set this to TRUE to configure the Windows Event Log monitoring to read only events that come in during the agent runtime. All events that come in during the system restart or when the agent is stopped are ignored.	YES	FALSE	Boolean
OPC_INCLUDE_VIRTUAL_IP_ADDRS_FOR_LOCAL_NODE_MAPPING	Before 8.16, messages from a node in an HA cluster were marked as local (not-proxied), an OPC_ADVMON_POLICY_VERSION_CHECKd were added to the physical node in the DB. As a result, the messages were ignored for the service tree of the virtual host, which used to result in incorrect status calculation. Since 8.16, the OPC_SET_PROXY_FLAG_FOR_IP_ADDRESSES setting is no longer needed. The OPC_INCLUDE_VIRTUAL_IP_ADDRS_FOR_LOCAL_NODE_MAPPING switches back to the old behavior if set to TRUE.	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_INT_MSG_FLT_AWS	With this variable set to TRUE, OM internal messages will always be filtered by the message interceptor even when the process does not run. Internal messages will arrive, in this case, at the next startup of the message interceptor. If you set this variable to TRUE, no internal messages will be sent unfiltered to the management server anymore. **Note:** OPC_INT_MSG_FLT must be set to TRUE in order to use OPC_INT_MSG_FLT_AWS.	NO	FALSE	Boolean
OPC_INT_MSG_FLT	If set to TRUE, OM-internal messages (message-group OpC or OpenView; mainly OM-internal status- and error-messages) are passed to the agent and can be filtered through message interceptor templates. **Note:** This is also possible on the OM management server. However, the local OM management server's agent must run and it must use the same character set as the server.	YES	FALSE	Boolean
OPC_KEEP_PERL_PATH	If set to TRUE, the monitor agent removes the Perl directory from the PATH environment variable before running a program.	YES	FALSE	Boolean
OPC_KILL_AUTO_ACTION	When set to TRUE, the kill operation for automatic actions is enabled. Every time an action should be started the action agent checks if its action queue contains already 10 actions. If so, it checks if the action running for the longest time runs longer than the predefined time out. If this is the case it kills this action. This works only when the OPC_NO_SHELL_TO_EXEC_ACTION variable is set to TRUE on UNIX platforms.	YES	TRUE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_KILL_AUTO_ACTION_TIMEOUT	This variable defines the timeout value of an action in the action queue. If an action, present in the action queue, fails to start within the time period specified with this variable, the action agent assumes that the action is hanging and proceeds to kill it. (See also OPC_KILL_AUTO_ACTION)	YES	590	Integer
EXT_INTERVAL	The frequency of checking the monitor queue if external monitors are configured.	NO	15 (sec)	Integer
ECA_ANNO_NODE	To add ECS annotate nodes on OM managed nodes.	YES		String
ECA_INSTANCE	Instance number of the ECS subagent on the managed node (**opceca**).	YES	12	Integer
ECA_PERLFILE	Name of Perl script that should be loaded into the ECS engine for the managed node and should reside in the AGENT_CONFIG_DIR directory.	YES	Empty	String
ECENG_CLOCK_INTERVAL	The global setting for the ECS engine clocking time (in milliseconds).	YES	1000	Integer
ECENG_LOG_LEVEL	The ECS tracing level setting. Possible settings: NONE, SEVERE, ERROR, WARN, and FULL.	YES	FULL	String
ECENG_TRACEFILE	Name of the ECS trace file.	YES	ecengtr	String
ECENG_TRACE_LEVEL	ECS tracing level setting. Possible values: NONE, FULL.	YES		String
ECENG_TRACE_RSIZE	The 'relative' file size for the ECS trace file configuration.	YES	100	Integer
ECEVI_LOG_RSIZE	'Relative' file size of ECS event-input log configuration.	YES	100	Integer
ECEVO_LOG_RSIZE	'Relative' file size of ECS event-output log configuration.	YES	100	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
EC_MAX_AS_WAIT	The total wait time of the ECS engine to connect to the annotate server: a) at startup, b) at the time of re-configuration.	YES	10 (sec)	Integer
EC_MAX_ESOK_TRY	Maximum number of retries done by the ECS engine to connect to the EC annotate server socket stack.	YES	20	Integer
OPC_KILL_SCHEDULE	On Windows, processes started for scheduled actions from the action agent sometimes hang. The older versions of the agent used to wait until a process finishes. The default behavior was changed to do the following: If a new request to start a process for a scheduled action arrives at the action agent, the agent first checks that the process is started from the same policy. If yes, then the agent checks that the process has already run longer than a configured time-out (default: 55 seconds). If yes, the old process is killed, and then the new process is started. If the new process is not started, a message is sent to the management server. The OPC_KILL_SCHEDULE variable can be used to disable the new functionality. If this variable is set to FALSE the action agent behaves like it did before.	YES	TRUE	Boolean
OPC_KILL_SCHEDULE_TIMEOUT	Defines the time-out that is used to check that the old process is killed or the new one is not started. (See also OPC_KILL_SCHEDULE).	YES	55	Integer
OPC_IP_ADDRESS	The value of this variable specifies the IP addresses of the managed node. If the agent is installed on a node in a Network Address Translation (NAT) environment, make sure the value of this variable on the node is identical with the IP address used in the OM console while adding the node.	YES		String
OPC_LE_CHECK_INODE	Sets the interval for the logfile encapsulator to register a file name	YES	20	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	change.			
OPC_LE_ CLOSE_MSG_ DLL	If this is set to TRUE, the NT msg DLL for EventLog messages gets closed after every read. This might cause a higher CPU usage of the logfile encapsulator process, but it does not lock DLLs.	YES	FALSE	Boolean
OPC_LE_IGN_ TEMP_UNAVAIL	If set to TRUE, temporarily unavailable log files (for example, NFS mounted) will not be treated as recreated or truncated log files. Once the log file is available, it is read from the last read position.	YES	FALSE	Boolean
OPC_LE_KEEP_ DSCONNECTIO N	On the Windows node, **opcle** locks the 1025 and 1026 ports toward the primary domain controller (PDC) emulator. If set to TRUE, the connection is kept open as long as **opcle** runs. If set to FALSE, the connection is closed after each request to unlock the port. **Note:** The logfile encapsulator will consider the file `le_state` as obsolete if it is older than 24 hours.	YES	FALSE	Boolean
OPC_LE_MAX_ LINES_READ	Specifies the number of lines read by the logfile encapsulator at every interval. If you do not want to set a limit, set this variable to **0**.	NO	50	Integer
OPC_LE_SAVE_ STATE	With this flag set to TRUE, **opcle** can save the information about the monitored files into the file `/var/opt/OV/tmp/OpC/le_state`. If the logfile encapsulator resumes its operation after an interruption and if the `le_state` file exists, the logfile encapsulator will start operating after gathering the information preserved in this file. Therefore, you can monitor the messages that were written into the monitored log files even when the logfile encapsulator was down.	NO	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_LE_STATE_FILE	Configure this property to set a non-default location of the `le_state` file.	NO	`var/opt/OV/tmp/OpC/`	String
OPC_MONA_MSG_PER_STATE	A message is sent each time a state/threshold is reached (rather than just the first time). This applies only to Advanced Monitoring in **opcmona.**	YES	TRUE	Boolean
OPC_NAMESRV_BUFFER_SIZE *Obsolete from Operations Agent 11.14.*	Provides the initial value for the buffer size to retrieve the host data from an IP address. An insufficient buffer size is incremented repeatedly by the specified value until the host data is retrieved successfully.	YES	512	Integer
OPC_NEW_LOGFILE_FROM_BEGIN	If set to TRUE, the logfile encapsulator reads the newly discovered log files from the beginning of the files. A script is used to dynamically list the log files to be monitored and the logfile policies that have already run once. If set to FALSE (default), the newly discovered log files will be read from the last file position.	YES	FALSE	Boolean
OPC_NODE_CHARSET	Character set of managed node.	YES	UTF-8	String
OPC_NO_MSG_FLT_FOR_BUFFER_MSG	When the internal message filtering is enabled, all internal messages are passed on to the message interceptor by default. If this flag is set to TRUE, messages regarding the message agent's buffering (OpC40-1410 and OpC40-1411) are excluded. They will be forwarded to the OM console directly.	NO	FALSE	Boolean
OPC_MSI_CREATE_NEW_MSGID	Controls the behavior of MSI-users in terms of message-id Creation You can use the following values: 1: Create a new message-id each time a message attribute is changed or the copy	NO	2	Integer , 1 <= n <= 4

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Require d	Default Value	Type
	operator is called. 2: Set no new message-id when attributes are changed and if this message is sent to only one instance. To set no new message-id, the message must be 'diverted' and not 'copied' so that the OM management server (an MSI API user group also retains a copy of it). If you apply the API copy-operator to a message, the copied message is no longer 'diverted.' Subsequently, attribute changes result in a new message id. Note that the message->orig_msgid attribute, which is accessible for API-users, contains the original message-id if it was changed (otherwise it contains a null-id). 3: As 2, except that the copy-operator immediately creates a new message-id for the copy. 4: Message-ids are not modified at all. API-user is responsible for it.			
OPC_OPC MON_ OVERRIDE_ THRESHOLD	If set to TRUE, the threshold/reset override feature for monitor policies in the monitor agent is enabled.	YES	FALSE	Integer
OPC_OPCMSG_ API_CHECK_ MSGI_ RUNNING	If set to TRUE, opcmsg() and opcagtmsg_ send() will not write messages to queue if the message interceptor does not run and the APIs will return an error.	YES	FALSE	Boolean
OPC_OPCMSG_ CLI_CHECK_ MSGI_ RUNNING	If set to FALSE, the opcmsg CLI will write messages to queue even if the message interceptor is not running.	YES	TRUE	Boolean
OPC_ PRIMARY_ MGR	Defines the primary manager for messages. For example: [eaagt] OPC_MGMT_ SERVER=servername.hp.com	YES	_	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_BACKUP_ MGRS	Defines the backup servers for messages when the primary server is down. **For example:** OPC_BACKUP_ MGRS=servername.hp.com, servername1.hp.com	NO	_	String
OPC_BACKUP_ MGRS_ FAILOVER_ ONLY	If set to TRUE, then the message agent sends the messages to the backup servers, only if the primary server is down. If FALSE, the message agent sends the messages to backup server irrespective of the status of the primary server.	NO	FALSE	Boolean
OPC_Q_SYNC_ WRITES	If set to TRUE, updates to the queue will be synchronized to the disk at critical places. This drastically reduces the queue performance, but also reduces the chance of queue file corruption.	YES	FALSE	Boolean
OPC_ RESOLVE_IP	Specifies the IP address that should be used for a managed node to contact its primary manager.	YES	_	String , a.b.c.d (e.g. 15. 136.120 .1)
OPC_ RESOLVE_ TRAP_ LOCALHOST	If set to TRUE, the event interceptor will replace the source address 127.0.0.1 in a trap with the agent IP address.	YES	FALSE	Boolean
OPC_ RESTART_ COUNT	Defines how many times an aborted subagent process should be restarted. (See OPC_RESTART_SUBAGENT)	YES	5	Integer
OPC_RPC_ SHORT_ TIMEOUT	Specifies the communication time-out for local RPCs (called and executed on the same machine). LOCAL_ONLY sets the time-out to 5 seconds, ALWAYS sets the time-out to 5 seconds for local and remote, NEVER sets the time-out to 30 seconds.	YES	LOCAL _ONLY	String , LOCAL_ ONLY, ALWAYS, NEVER
OPC_SEND_ MAX_ONE_ MSG_PER_ TRAP	Can disable processing of additional trap templates if there was already a message generated from one template. This speeds up the processing, but may prevent some	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	templates from accepting the trap. As a result, fewer messages might reach the server and the behavior of duplicate message suppression for 'Suppress Messages Matching Condition' may change.			
OPC_SET_PROXY_FLAG_FOR_IP_ADDRESSES	Specifies a list of local IP addresses that will not be replaced by the node name as known on the server while sending messages. This is useful for agents with the OM for Windows management server. When messages are sent with the sender address of the list, the is_proxied flag of the message will be set as if the message was sent for a different node - even if that address is local (currently). You must not include the OPC_IP_ADDRESS in this list.	NO		String (comma separated list of IP address-es, e.g.)
OPC_TEMPL_STATUS	If set to TRUE, the logfile encapsulator and monitor agent will maintain the `le.status` and `monitor.status` files in the agent's `tmp` directory containing the current (internal) status of their sources.	YES	FALSE	Boolean
OPC_TRACE_CHILD	If set to TURE, tracing will be enabled between fork and exec for the child process (which can lead to mutex deadlock on multiprocessor machines)	YES	FALSE	Boolean
OPC_TRUNCATE_ORIG_TEXT	Truncates the original message text to a maximum length. -1: no truncation done (Default) 0: no original message sent at all <n> : truncation done after the <n>-th character.	NO	-1	Integer
OPC_USE_PROTECTTOOLS	The ProtectTools offer enhanced security features on Windows. One of them is password tagging/salting/preprocessing. If those tools are active, the agent needs to preprocess passwords to be able to	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	perform a switch user operation. Setting this variable to TRUE enables the agent to preprocess passwords.			
OPC_USE_ UDP_AS_ TRAP_SOURCE	To match the post master daemon's `-u` option, this variable can be set to TRUE. As a result, agent_addr (source of the trap) will be overwritten by the UDP packet header's source IP address (works only when pmd is running with `-u`).	YES	FALSE	Boolean
OPC_WBEMI_ BUF_SIZE	The WMI/WBEM interceptor uses an internal queue to buffer incoming objects until they can be checked. The default size of this queue is 10000 objects, but this size can be changed by using the variable OPC_WBEMI_BUF_SIZE. If too many objects arrive at the buffer queue, older objects are deleted from the queue and not handled. There are some possibilities to avoid such a buffer overflow: Check why so many objects arrived. Check that it is possible to limit the number of objects the WMI/WBEM interceptor has to handle by using a global WQL filter for the policy. Check that it is possible to use rules to suppress objects.	YES	_	_
OPC_WIN_ UAC_ENABLE	If the value of OPC_WIN_UAC_ENABLE is TRUE, agent will elevate the user and assign the administrative token with complete access if the user is part of administrative group. This will enable the user to launch processes with administrative privileges like launching a process on a node using OM tools. This flag should be set to TRUE only if UAC is enabled on the system.	YES	FALSE	Boolean
OPC_WIN_ DONT_USE_ PATH_	By setting this variable to TRUE, you can force the Windows agent to scan the currently used environment PATH to	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
NWDRIVE	remove all mapped network paths referenced in the PATH to avoid log-on errors on the systems providing the network share. This also removes relative PATH entries like ""."" or "".."" ."			
POLICY_MIN_ INTERVALS_ WAIT	The minimum number of wait intervals before stopping a policy when the policy does not receive any data. This is important while using the agent with program sources where the execution time of an external program depends on the current system performance. If the system is very busy, the execution might take longer than the configured interval. To configure how long the monitor agent should wait for the external programs to complete, this variable can be used. Use -1 if POLICY_MIN_TIME_WAIT should be used. Use 0 if the policy should not wait.	YES	-1	Integer
SNMP_ COMMUNITY	Standard SNMP community to be used in monitoring MIB objects with the Measurement Threshold policies.	NO	public	String
SNMP_ COMMUNITY_ LIST	List of SNMP communities to be used in monitoring MIB objects with the Measurement Threshold policies. By using this variable, specify a list of community strings separated by commas. The Operations Agent attempts to collect the MIB objects by using the first community string specified in the string. If the operation fails, the Operations Agent performs the same operation with the second community string in the list, and so on. If all the community strings fail to aid the Operations Agent to gather data, the community string specified with the variable SNMP_COMMUNITY takes effect.	NO	String specified with the variable SNMP_ COMMUNITY	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
SNMP_ REFUSE_ FORWD	Specifies whether the SNMP trap interceptor accepts events forwarded from another `pmd` on a remote NNM management station.	YES	FALSE	Boolean
SNMP_ REMOTE_PMD	Host where the SNMP trap interceptor tries to connect to `pmd` on a remote NNM management station.	YES	"" (local host)	String , any host name
SNMP_ REMOTE_PORT	Port number to which **opcmona** establishes a connection while monitoring SNMP variables.	YES	161	Integer, values: > 0
SNMP_ SESSION_TRY_ FOREVER	If this variable is set to TRUE, the **opctrapi** process will try to connect to `pmd` every 300 second until the connection is established.	YES	FALSE	Boolean
SNMP_TRAP_ FORWARD_ ENABLE	By setting this property to TRUE, you can enable the SNMP trap interceptor to forward the SNMP traps available on the node to remote systems or management stations.	YES	FALSE	Boolean
SNMP_TRAP_ FORWARD_ DEST_LIST	With this property, you can set the address of the remote management station where you want to forward all the available SNMP traps. You can specify multiple system names separated by commas.	YES		String
SNMP_TRAP_ FORWARD_ COMMUNITY	With this property, you can specify the necessary community string for the target system where you want to forward the SNMP traps. If you want to configure multiple target systems, specify the appropriate community strings separated by commas.	YES		String
SNMP_TRAP_ FORWARD_ FILTER	With this property, you can filter the available SNMP traps by their OIDs and forward only selected traps to the remote system. The filtering mechanism takes effect with the wildcard (*) character. For example, if you set this property to 1.2.3.*.*.*, the SNMP trap interceptor will	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	forward all the SNMP traps with the OIDs that begin with 1.2.3. By default, all the available traps are forwarded when you enable the SNMP trap interceptor to forward traps.			
OPC_LIMIT_MSG_WAIT_FOR_AA	Wait time for a message to receive an action response.	YES	3600	Integer
OPC_MAX_ERROR_HANDLING	If set to TRUE, errors are always sent as OM messages to the management server. **Note:** Only the **opcmon** command uses this feature.	YES	FALSE	Boolean
OPC_MAX_IP_PER_INTERFACE	Buffer size for storing IP addresses per interface.	YES	128	Integer
OPC_MON_MSGOBJ_COND_FIRST	<$MSG_OBJECT> of a monitor policy is replaced with the object that was set in the policy condition window. <$MSG_OBJECT> in a threshold monitor policy is replaced according to the following priority if OPC_MON_MSGOBJ_COND_FIRST is TRUE: 1. Object from set attributes 2. Object from **opcmon** 3. Template Default 4. Empty String	YES	FALSE	Boolean
OPC_MON_SAVE_STATE	If you set this variable to FALSE, the monitor agent stops preserving the values of monitored objects and session variables. If you set this variable to TRUE, the monitor agent starts preserving the values of monitored objects and session	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	variables. When you disable or deploy a newer version of measurement threshold policy, the policies are removed from the **opcmona** memory and the values of monitored objects and session variables of the existing running policies are also removed. The new policy version is then loaded and the **opcmona** will store new values for the monitored objects and session variables.			
OPC_MSGA_ PING_ SERVER_ INTERVAL	Interval (in seconds) in which unreachable OM servers are pinged by the communication component of the agent node.	YES	60	Integer
OPC_ NAMESRV_ BUFFER_SIZE	Provides initial value for the buffer size to get the host data from IP address; if buffer size is insufficient, then the size is incremented repeatedly by the provided value until host data is retrieved successfully.	YES	512	Integer
OPC_ NAMESRV_ LOCAL_NAME	Fully-qualified long hostname of a node. Will be used if OPC_IP_ADDRESS is unresolvable. (Usually used in NAT environments) The same can be achieved by adding an entry to the hosts file: <NAT IP> <name> on UNIX: `/etc/hosts`; on Windows: `%SYSTEMROOT%/system32/drivers/etc/hosts`	YES		String
OPC_NO_ PORTS_DELAY	When all ports of the port range specified for an RPC client (for example: `ovoareqsdr, opcragt, opcmsga`) are occupied, the next communication attempt is made after the time (in seconds) delay specified with this variable.	YES	1	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_NODE_TYPE	Node type of managed node; Values: CONTROLLED,MONITORED,MESSAGES_ALLOWED,UNMANAGED	YES	CONTR-OLLED	String
OPC_NO_SHELL_TO_EXEC_ACTION	By default, OM runs automatic and operator-initiated actions on UNIX in a shell. With this flag set, all actions are performed directly with the fork() or exec() system calls. Alternatively, you can prefix any action- or application call with the string _NO_SHELL:. This allows execution of single tasks without using a shell.	YES	FALSE	Boolean
OPC_OPCLE_POLICY_EXCLUDE_INFO	Comma-separated list of policy-name prefixes. All policies, with names that begin with such prefixes, are not processed by the normal **opcle**. Example: "abc,bcd" -> policies "abcd 500" and "bcde 600" are not processed. This variable is only useful in conjunction with the "multiple parallel **opcle**" feature.	YES		String; comma-separated list, no blanks in between.
OPC_PERL_INCLUDE_INSTR_DIR	If set to TRUE, Perl modules available in the instrumentation directory will be made available for embedded Perl policies processed by the monitor agent.	YES	TRUE	Boolean
OPC_PERL_PROG_BIN	Path to the Perl executable installed by OM.	YES	platform dependent directory under CSM_OVBIN_DIR()	String
OPC_STORE_TIME_FOR_MGR_INFO	Maximum time a message operation info block is held in the message agent.	YES	24	Integer , hour

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_TEMPL_ STATUS	Log file encapsulator and monitor agent maintains the `le.status` and `monitor.status` files in the tmp directory of the managed node. These files contain the current (internal) status of their sources if this variable is set to TRUE.	YES	FALSE	Boolean
POLICY_MIN_ TIME_WAIT	Minimum time to wait before stopping a policy if it does not receive any data. The time is specified in minutes. Important for program sources where the execution time of an external program depends on the current system performance. If the system is very busy, it is possible that the execution takes longer than the configured interval. Reconfiguring the time interval that the monitor agent waits for external programs to finish can be helpful.	YES	2	Integer
POLICY_MIN_ INTERVALS_ WAIT	Minimum number of wait intervals before stopping a policy if it does not receive any data. Important for program sources where the execution time of an external program depends on the current system performance. If the system is very busy, it is possible that the execution takes longer than the configured interval. Reconfiguring the time interval that the monitor agent waits for the external programs to finish can be helpful. Use -1 if POLICY_MIN_TIME_WAIT should be used. Use 0 if the policy should not wait.	YES	-1	Integer
SNMP_CONFIG	Name of the SNMP trap interceptor configuration file	YES	trapi	String
SNMP_EVENT_	Specify the events that will be forwarded	YES	CORR	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
FLOW	from NNM pmd to **opctrapi.** Values: CORR - NNM's correlated events. RAW - As if no event correlation took place ALL - Correlated events and raw events.			
SNMP_EVENT_LIST	Specify a filter which is given to NNM pmd. The filter defines which events are forwarded to **opctrapi**.	YES	.*	String
SNMP_SESSION_TRY_FOREVER	If this variable is set to TRUE, the SNMP trap interceptor process will try to connect to NNM pmd every 300 second until connection is established.	YES	FALSE	Boolean
SNMP_STREAM_NAME	Specify which event streams are forwarded from NNM pmd to **opctrapi**.	YES	(not set; registers to default-stream)	String
SNMP_V3_ENABLE	Enables **opctrapi** to intercept SNMPv3 traps. To disable **opctrapi** from intercepting SNMPV3 traps, set the variable to **FALSE**. If the variable is set to **FALSE**, **opctrapi** intercepts only SNMPv1 and SNMPv2 traps. Namespace - `eaagt`	NO	TRUE	Boolean

SNMP_V3_USERSDATA	Use the variable to configure SNMPv3 users.
	SNMP_V3_USERSDATA = `<Parameters for User 1>;<Parameters for User 2>;<Parameters for User n>`
	In this instance, `<Parameters for User>` includes the following:
	`<user name>:<authentication`
		NO		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	`method>:<authentication password>:<encryption method>:<encryption key>:<Engine ID>` In this instance: `<user name>` - Specifies the SNMPv3 user name created by the administrator. `<authentication method>` - Specifies the protocols used to encrypt the password. You can either use Message Digest Algorithm 5 (MD5) or Secure Hash Algorithm (SHA) protocols to encrypt the password. `<authentication password>` - Specifies the password encrypted using the **opcpwcrpt** utility. `<encryption method>` - Specifies the protocols used to encrypt the Protocol Data Unit (PDU). You can either use Data Encryption Standard (DES) or Advanced Encryption Standard (AES) protocols to encrypt the PDU. `<encryption key>` - Specifies the key used by DES and AES to encrypt the PDU. **Note:** Encryption key created by the administrator is encrypted with the **opcpwcrpt** utility. `<Engine ID>` - Specifies the SNMPEngineID that is used to uniquely identify SNMP entities. Namespace - `eaagt`			
MSGSRC_ WITH_POLICY_	If the value of MSGSRC_WITH_ POLICY_VERSION is TRUE, then policy	YES	TRUE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
VERSION	version will be appended to MSGSRC variable. If the value of MSGSRC_WITH_POLICY_VERSION is FALSE, then policy version will not be appended to MSGSRC variable.			
OPC_LE_CMD_WAIT_TIME	Maximum wait time for **opcle** to wait for the child processes to complete the task and get the status back. By default, **opcle** waits for 15 seconds. If the child doesn't return by this time, **opcle** kills the child process. This variable is applicable only in the UNIX/Linux environments.	YES	15	Integer
OPC_LE_CALC_HASH	If the value of OPC_LE_CALC_HASH is TRUE, **opcle** detects the file is appended or overwritten by calculating hash of last line and verifying some random check points.If **opcle** detects the file is overwritten, then it reads the file from beginning.	YES	FALSE	Boolean
OPC_IGNORE_DEFAULT_MSG_CORRELATION	If the value of OPC_IGNORE_DEFAULT_MSG_CORRELATION is TRUE, when the agent converts the loaded configuration to the internal data structure, default message correlation values are not merged to the condition correlation values. If the value of OPC_IGNORE_DEFAULT_MSG_CORRELATION is FALSE, when the agent converts the loaded configuration to the internal data structure, default message correlation values are merged to the condition correlation values.	YES	FALSE	Boolean
OPC_ADVMON_POLICY_VERSION_CHECK	This variable helps the Operations Agent verify and compare the versions of monitor policies that are run by **opcmona** sub agent. These policies are brought to effect on the node only when the Operations Agent determines that the versions of	YES	TRUE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	redeployed policies are higher than the existing versions. If this variable is set to TRUE, the **opcmona** policy types such as Measurement Threshold, Service Process Monitoring and Scheduled Task are affected. If this variable is set to FALSE, **opcmona** reloads all the monitoring policies, even if one of the monitoring policies is disabled or enabled. Therefore, it is best to set this variable to TRUE.			
OPC_INSTALLED_VERSION	The version of the installed agent.	YES	Not set	String , values: A.VV.FF
OPC_MSG_FLT_EXCLUDE_SVC	If set to TRUE, the Service Name field is not carried forward from the original OM-internal messages. The Service Name field for the message in the message browser is then left empty.	YES	FALSE	Boolean
OPC_MAX_MSG_LEN	Limit for the size of a message. Some sanity checks are made on incoming messages. OPC_MAX_MSG_LEN defines the accepted limit in bytes. If a message arrives that is bigger than the specified limit, the message is discarded or truncated.	YES	1048576 (1 MB)	Integer
OPC_RESOLVE_MAC_ADDRESS	If set to TRUE, MAC addresses (0x<6 hex bytes>) are resolved. Otherwise, the name service cache returns NULL. This can be set on both the agent and server.	YES	FALSE	Boolean
OPC_Q_MAX_SIZE	Newly created queue files have this size limit imposed.	YES	0 (unlimited)	Integer , kbytes

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	If the queue is within the limit, a queue element may be appended, resulting in a queue that is possibly slightly beyond the limit. If the queue file is larger than this limit, writing to the queue is not possible, and the same actions as if the disk was full are performed (shift queue and/or sleep up to OPC_Q_MAX_RETRY_TIME seconds, then fail). Has no effect on the reader of a queue.			
OPC_MGMT_SERVER	Fully qualified hostname of the OM management server.	YES	unknown (set at installation time)	String
OPC_MSI_CONF	Name of config-file for serial MSIs (message stream interface)	YES	msiconf	String
OPC_NAMESRV_RETRIES	Number of retries of gethostbyname and gethostbyaddr calls.	YES	3	Integer
OPC_SUPPRESS_ERROR_LIST	A comma-separated list of <msgset>-<msgnbr> values to suppress error messages (OpC<msgset>-<msgnbr>) output to all error message output targets. Set the setting on a process basis as some binaries may output the same error message on error condition and suppressing the message would alter the normal output. Do not insert any whitespaces. **For example:** To suppress error messages OpC50-10 and OpC50-202 for opcmsgm process: `ovconfchg -ovrg server -ns opc.opcmsgm \ -set OPC_SUPPRESS_ERROR_LIST \ "50-10,50-202"`	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_ NAMESRV_ CACHE_SIZE *Obsolete from Operations Agent 11.14.*	OM uses a name-resolution cache in all processes to improve performance. If the cache is full, least frequently used entries are replaced by new ones. For large environments, it is recommended to enlarge the cache size. (See also OPC_ NAMESRV_*).	YES	100	Integer
OPC_ NAMESRV_ DISABLE_ CACHE *Obsolete from Operations Agent 11.14.*	Enable and disable OM name-service cache.	YES	FALSE	Boolean
ECEVO_LOG_ RSIZE	'Relative' file size of ECS event-output log configuration.	YES	100	String
OPC_ NAMESRV_ MAX_TIME *Obsolete from Operations Agent 11.14.*	Allowed resolution time for one node in milliseconds. You get a WARNING in the trace and `opcerror` file if the configured limit is exceeded: Name resolution for node 'hostname.at.domain' took xxx milliseconds (exceeded the configured threshold of yyy) (OpC20-2212) Note, that this is only for reporting, the name service call will NOT be terminated if the max time is reached. You will need to use name service client settings for that (e.g. via the retry and retrans keywords in `/etc/resolv.conf` for DNS). The details depend on OS and used name service.	YES	200	Integer (milliseconds)
ALERT_LTU_ EXPIRY_DAYS	With the help of this variable, you can configure the Operations Agent to send alert messages to the OM message	YES	7,3,1	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	browser to notify you about the expiry details of the evaluation agent LTUs that are in effect on the node. While setting this variable, you must specify three comma separated integer values in the descending order. **For example:** `ovconfchg -ns eaagt -set ALERT_LTU_EXPIRY_DAYS` *DAY1,DAY2,DAY3* The alert messages will arrive in the OM console on DAY1, DAY2, and DAY3 days before the evaluation LTU expires. You can also configure the Operations Agent to stop sending the alert messages by setting the comma separated integer value to `-1`. **For example:** `ovconfchg -ns eaagt -set ALERT_LTU_EXPIRY_DAYS "-1,-1,-1"`			
MAX_NBR_WBEM_PARALLEL_THREADS	This variable is in effect only on Windows nodes. The maximum number of threads that can run parallel to process the objects retrieved by **opcwbemi.** This variable can be used if **opcwbemi** is processing huge number of events. Maximum value is 10.	YES	1	Integer
OPC_LE_SKIP_NO_OF_INVALID_EVENTS	An event provider represents an application. Each registered provider informs the system about the set of events, metadata related to these events, and descriptions of the events that the provider can publish. When the providers are not registered correctly, it will not be able to publish its events, or the event description. These events are called Invalid Events. Set the number of invalid events to be	YES	0	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
	skipped by Operations Agent. When the number of invalid events exceeds the specified value, a warning message is sent to the OM message browser.			
OPC_MON_SAVE_STATE_CACHE_THRESHOLD	Specify the threshold value after which an `opcmona` state change is written to the Save State file. Specify a value of 1 if you want to write every opcmona state change to the Save State file.	YES	10	Integer
CODA_CONNECTION_TIMEOUT	During **ovc** restart, **opcmona** connection to **oacore** may fail as **oacore** is in initialization state. Specify the timeout value in minutes after which the `opcmona` can retry connecting to **oacore**.	YES	1	Integer
OPC_LE_READ_MULTILINE_FOR_UNREGSOURCES	Set this variable to TRUE if **opcle** is reading an eventlog on Windows 2008 server and the event has multiline description for an unregistered source event.	YES	FALSE	Boolean
OPC_FORMAT_OCTET_STR	If this variable is set to TRUE, varbind data is attempted to represent as string. If fails, then the data is represented in hex-string (octets from SNMP). Data is validated with respect to OPC_TRAP_CHARSET and OPC_NODE_CHARSET. If this variable is set to FALSE, data is printed without any charset conversion. But, if the data has non-printable characters, varbind data is represented in hex-string.	YES	TRUE	Boolean
OPC_SET_DATASOURCENAME_SI	This variable is used by the monitor agent to set the summarization interval while querying data for the data sources like CODA, SCOPE, VISPI, etc. For example, OPC_SET_VISPI_SI=15m. Possible values are: "1s", "5s", "15s", "30s", "1m", "5m", "15m", "30m", "1h", "3h", "6h", "12h", "1d", "1w", "1M", "1Q".	YES	5 minutes	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPCGENI_DISABLE_LOCAL_STORE	Set this variable to TRUE to disable data logging in the Operations Agent's local datastore. Namespace - `eaagt` **Note:** If OPCGENI_DISABLE_LOCAL_STORE is set to *true* and *no* data forwarding policy is enabled then a *warning* message is logged into the log file (**System.txt**) as well as the message is sent to the management server as an internal message indicating that data is not getting forwarded or logged locally and this may result in loss of metrics processed by **opcgeni**.	NO	FALSE	Boolean
OPC_LOG_NON_RESOLVABLE_HOST	If OPC_LOG_NON_RESOLVABLE_HOST is set to TRUE, then **opcmsga** writes all unresolvable nodes to **System.txt**. Namespace - `eaagt`	NO	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued

Variable	Description	Restart Required	Default Value	Type
OPC_INSTALLED_VERSION	Defines the version string of the agent (**opcinfo**) and the server (**opcsvinfo**). Namespace: `eaagt`	YES		String
OPC_PRIMARY_MGR	Defines primary manager for messages. It corresponds to `mgrconf` policy. Namespace: `eaagt`	YES		String
OPC_TRAP_CHARSET	Specifies the character set of the incoming SNMP traps. If the variable is not set, then the default character set (utf8) is considered as the incoming traps character set and is converted to the value set in OPC_NODE_CHARSET. Namespace: `eaagt`	YES		String
SNMP_TRAP_PORT	Specifies the port number to which **opctrapi** connects when not using NNM pmd. Namespace: `eaagt`	YES	162	Integer, values: > 0
SNMP_STREAM_NAME	Specifies the event streams that are forwarded from NNM pmd to **opctrapi.** See NNM Manuals. Namespace: `eaagt`	YES		String
SNMP_EVENT_LIST	Specifies a filter that is assigned to NNM pmd. The filter defines the events that have to be forwarded to **opctrapi**. Namespace: `eaagt`	YES		String
SNMP_EVENT_FLOW	Determines the events that have to be forwarded from NNM pmd to **opctrapi**. Namespace: `eaagt`	YES	CORR	String
SNMP_REMOTE_PMD	Denotes a host where the SNMP trap interceptor tries to connect a postmaster process (PMD).	YES	"" (local host)	String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Namespace: `eaagt`			
OPC_ INTEGER_ MSG_ SEVERITY	Use the variable to set the message severity of the system error. Severity of the error message can be set to **minor, major, warning** or **critical**. The **opcle** message will be sent to the Operations Manager (OM) with the set severity. **For example:** To set the message severity to minor: `ovconfchg -ns eaagt -set OPC_ Integer_MSG_SEVERITY minor` Namespace: `eaagt`	YES	Warning	String
CSM_CFG_ MON_SNMP_ SESSION_ MODE *Not Supported with Operations Agent version 12.00 onwards*	If this variable is set to PRE-NETSNMP, the monitoring agent uses NNM APIs to perform SNMP GET request and if this variable is set to NETSNMP, the monitoring agent will use NETSNMP APIs to perform SNMP GET request. SNMP GET requests over IPv6 are supported only in NETSNMP mode. Namespace: `eaagt`	YES	NETSNMP	String
OPC_ BUFLIMIT_ SEVERITY	Defines the minimum severity for the message when buffer-file on agent is full but messages should be received. Set this in Advanced Modify Node screen. **Note:** Do not change this setting on the node using: `ovconfchg -ns eaagt -set <variable name> <value>` Namespace: `eaagt`	YES	Major	String
OPC_AGTMSI_ ENABLE_NODE	Allows all messages to be either forwarded or copied to Message	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Stream Interface (MSI). The variable should be set on an agent node. Precedence is given to the MSI settings in the policy. The following options are available: • **MSI_DIVERT**: Forwards messages only to MSI if MSI is open. • **MSI_COPY** : Forwards messages to the browser and to MSI. • **MSI_DIVERT_ACTION**: Forwards messages to MSI and executes action if defined. Namespace: `eaagt`			
OPC_EVENT_CONVERT_SID_TO_NAME	On Windows 2008 machine, set the variable to TRUE, if the conversion of Security Identifier (SID) to principal name is needed in the event text in a Windows EventLog. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_HBP_INTERVAL_ON_AGENT	Set the variable in OM Admin Interface to modify the time interval for *I_am_alive* packages that are sent from agent to primary manager. **Note:** Do not change this setting on the node using `ovconfchg -ns eaagt -set <variable name> <value>` command. Namespace: `eaagt`	YES	-1	Integer
OPC_BUFLIMIT_SIZE	Use the variable to set the buffer-file limitation on agent. The set value defines the limit for `msgagtdf` file in kilobytes. Set this value in Advanced Modify Node screen.	YES	10000	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Note: Do not change this setting on the node using `ovconfchg -ns eaagt -set <variable name> <value>` command. Namespace: `eaagt`			
OPC_MAX_ QUEUE_ DUMP_LEN	Limits the number of bytes that are not used. **Note:** To use this feature, switch on `OPC_DUMP_QUEUE_CONTENTS`. Namespace: `eaagt`	YES	36	Integer
OPC_USE_ LOWERCASE	Converts all node names to lower case in the name service cache. This avoids problems with the name server returning the same name with differing case. When switching to OPC_USE_ LOWERCASE, you should also run **opcdbidx -lower**, to convert all node names of IP nodes in the database to lower case. Namespace: `opc` **Note:** This configuration variable can be used only on the Operations Agent running on the management server.	YES	FALSE	Boolean
OPC_SOCK_ BUF_SIZE	Specifies the number of bytes that can be transferred over a particular socket using either **opctss** or **opcdista** process. Namespace: `eaagt, opc`	YES	4096	Integer
OPC_COMM_ PORT_RANGE	Provides the Distributed Computing Environment (DCE) port range specification to be used on managed nodes or management server. Namespace: `eaagt, opc`	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
OPC_TRACE	Enables agent or server trace. Namespace: `eaagt, opc`	YES	FALSE	Boolean
OPC_TRACE_ AREA	Specifies agent or server trace areas. Namespace: `eaagt, opc`	YES	MSG, ACTN	String
OPC_TRC_ PROCS	Lists all process names that have to be traced. The process names should be separated by commas. Namespace: `eaagt, opc`	YES		String

OPC_DEBUG_ AREA	Specifies agent or server trace areas for special debug-output (separated by comma with no whitespace). If OPC_TRACE_AREA DEBUG is set and OPC_DBG_AREA is not set then area ALL is used.
	For example:
	`OPC_TRACE TRUE`
	`OPC_TRACE_AREA DEBUG`
	`OPC_DEBUG_AREA RQS,NAMESRV`
	Namespace: `eaagt, opc`
		YES		String

OPC_DBG_ PROCS	Lists the names of all traced processes that you should debug. The process names should be separated by commas. **For example:** `OPC_DBG_PROCS ovoareqsdr,opcmsgrd` Namespace: `eaagt, opc`	YES		String
OPC_DBG_ EXCLUDE_ AREA	Gives a comma-separated list of debug-areas which were ignored. For Example: `OPC_DBG_EXCLUDE_ AREA MUX, CONF, NLS` Namespace: `eaagt, opc`	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
OPC_EXEC_ PROCS	Gives a comma-separated list of programs that should be considered when running special tracing tasks. Namespace: `eaagt, opc`	YES		String
OPC_ INSTALLED_ VERSION	Specifies the version string of the agent (**opcinfo**) or server (**opcsvinfo**). Namespace: `eaagt, opc`	YES		String
OPC_TRACE_ UNKN_AREAS	Set the variable to TRUE, to find statements from areas that are not in the OPC_TRACE_AREA list. Namespace: `eaagt, opc`	YES	FALSE	Boolean
OPC_TRACE_ TRUNC	Specifies whether the agent or server trace line length is limited. Namespace: `eaagt, opc`	YES	TRUE	Boolean
SNMP_MASK_ COMMUNITY *Introduced with the Operations Agent 11.00.*	Set the variable to TRUE, to conceal the community value in the message sent to the server by the SNMP trap interceptor. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_ RESOLVE_ NAME_IP *Introduced with the Operations Agent 11.00.*	Set the variable to TRUE, to resolve IP to hostname and the reverse. Namespace: `eaagt`	YES	TRUE	Boolean
OPC_USE_ PROTECTTOOLS *Introduced with the Operations Agent 11.00.*	Set the variable to TRUE, to initialize the COM library on the current thread and to identify the concurrency model as single-thread apartment Namespace: `eaagt`.	YES	FALSE	Boolean
EC_SHOW_ TIMESTAMP	Set the variable to TRUE, to enable the timestamp writing for `ecenglg`	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
Introduced with the Operations Agent 11.00.	and `ecengtr` files. Namespace: `eaagt`			
OPC_HA_ TRAPI *Introduced with the Operations Agent 11.00.*	In high-availability (HA) environment, set the variable to TRUE, to ignore connection errors made on passive node. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_ENABLE_ MDM *Introduced with the Operations Agent 11.00.*	Set the variable to TRUE, to run an embedded VBScript with Process Debug Manager (PDM). The PDM is a component of Visual Studio that manages programs and processes, making them available to the session debug manager and the debug engines. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_ ENCRYPT_ ACTREQ_ FIELDS *Introduced with the Operations Agent 11.00.*	Set the variable to TRUE, to encrypt the password sent in the action request. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_CODA_ COLLECTION_ RETRY_SLEEP *Introduced with the Operations Agent 11.00.*	Specifies the inactive period between retries. It works in combination with OPC_CODA_COLLECTION_TRY which specifies the number of retries. Namespace: `eaagt` **Note:** OPCMONA retries to connect to **oacore** if **oacore** is in initializing state.	YES	5	Integer
OPC_MON_ PERFLIB_ REINIT_TIME	Specifies the time interval (in milliseconds) to reinitialize the Perfmon database connection from **opcmona,** when a data source fails to	YES	21600	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
Introduced with the Operations Agent 11.00.	collect data. Namespace: `eaagt`			
OPC_MON_ INTERVALS_ WAIT_FOR_ DATA *Introduced with the Operations Agent 11.00.*	Specifies the time duration, the program would wait for data from the monitor program, before it is stopped or a new collection is started. Namespace: `eaagt`	YES	2	Integer
OPC_MON_ WAIT_FOR_ PROCTABLE *Introduced with the Operations Agent 11.00.*	Specifies the number of program retries that must be performed to start the monitor program if a **proc table full** error occurs. Namespace: `eaagt`	YES	60	Integer
FAILED_INIT_ RETRIES *Introduced with the Operations Agent 11.00.*	Specifies the number of retries that must be performed to reinitialize a failed datasource. Namespace: `eaagt`	YES	3	Integer
OPC_KILL_LE_ CMD_TIMEOUT	During **opcle** processing, if embedded commands stop responding, **opcle** fails to process remaining policies. Set this variable to enable **opcle** to stop embedded commands, if they are running longer than the specified timeout. Namespace: `eaagt`	YES	0	Integer
OPC_MON_ DISABLE_ PROG_VARS	If set to TRUE, the agent does not resolve the program variables set in measurement threshold policies. Namespace: `eaagt`	YES	TRUE	Boolean
OPC_ NODENAME	This variable is assigned the value of local host name. It is set by **opcmsga** (and also reset by **opcmsga**)	YES		String

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	whenever IP address of an agent node is changed. Namespace: `eaagt`			
OPC_COND_ EVT_LOG_ SRC_PAT	Set the variable to TRUE, to enable pattern matching in the source field of Windows EventLog policy. Namespace: `eaagt`	YES	FALSE	Boolean
OPC_LE_ REPLACE_ DSNAMES	If set to TRUE, DSNAMES in Windows Events will be expanded during event processing. Namespace: `eaagt`	YES	TRUE	Boolean
OPC_ CONFIGFILE_ TIMEOUT	Specifies the timeout for the command that is executed during the processing of **ConfigFile** policy type. Namespace: `eaagt`	YES	30 Seconds	Integer
OPC_THREAD_ STACK_SIZE	Set this variable to increase the application thread stack size when default application thread stack size on AIX/HPUX/SOL/LINUX is not sufficient. Namespace: `eaagt`	YES	Linux: 524288, AIX: 262144, Solaris and HPUX: 5242880	Integer
MAX_ NUMBER_ PROCESSED_ TRAPS	If the number of processed traps exceeds the set value or the default value, an error message indicates that the processing of traps is in process by providing the maximum time taken for formatting traps. Namespace: `eaagt`	YES	100	Integer
OPC_MSI_ GROUP	Enumerates the members of the OS group who are authorized to access the MSI agent. Namespace: `eaagt`	YES	OvMsiGroup	String
SNMP_RECV_ BUF	Specifies a method to configure the NETSNMP read buffer value.	YES	65536 (bytes)	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Increment SNMP_RECV_BUF cautiously in multiples of 64 (64KB,128KB, 192KB, 256KB and so on) and then choose a value that suits better. **NOTE:** Do not change **KERNEL UDP** buffer values along with this configuration variable. Namespace: `eaagt`			
ECM_PERLFILE	Specifies the name of perl script, which should be loaded into ECS engine for a managed server and should reside in `MGMT_SV_EC_TEMPL_DIR` directory (`$OV_VAR_DIR/conf/OpC/mgmt_sv/`). Namespace: `opc`	YES		String
EC_SHOW_TIMESTAMP	Shows the timestamp in `ecenglg` and `ecengtr` files. Namespace: `opc`	YES	FALSE	Boolean
NUM_SRV_THREADS	Specifies the number of threads created by a DCE RPC (Distributed Computing Environment Remote Procedure Calls) server. Namespace: `opc`	YES	40	Integer
OPC_Q_MAX_RETRY_TIME	Specifies for how long queue elements can be appended when the disk is full or if the maximum allowed queue size is exceeded. If the threshold is passed, the associated processes terminate with an error. Namespace: `opc`	YES	MAX_Integer (quasi infinite)	Integer
OPC_RPC_ONLY	Use this variable on Operations Agent to prevent them from sending ping packages if the management server is not reachable.	YES	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Namespace: `eaagt`			
OPC_AGENT_ LOG_SIZE	Specifies the maximum size of a log file on an agent (in 1/10 KB) or server (in KB). On a managed node, a new log file is created when the current log file size is 1/10 KB * 10000 = 1 MB. On a server, a new log file is created when the current log file size is about 10 MB. Log files are named `opcerror1`, `opcerror2`, and so on and are set in OM Admin interface. Namespace: `eaagt`	YES	10000	Integer
OPC_AGENT_ LOG_DIR	Specifies the log directory on agent and is set in OM Admin interface. Namespace: `eaagt`	YES	platform dependent directory under CSM_ OVBIN _ DIR()	String
OPC_ENABLE_ OID_TO_TEXT_ CONVERSION	If this variable is set to TRUE, **opctrapi** will convert OID to its textual representation. This variable is applicable for the interception of all versions of traps SNMPv1, SNMPv2c and SNMPv3. **Note:** When OPC_ENABLE_ OID_TO_TEXT_CONVERSION is enabled: 1. `$E` will resolve to converted text (human readable format) for enterprise id. 2. `$e` will resolve to converted text (human readable format) for enterprise id.	NO	FALSE	Boolean

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	3. `$O` will remain as it is in numeric form (ex:.1.4.3.2.2) 4. `$o` will remain as it is in numeric form (ex:.1.4.3.2.2) Namespace: `eaagt`			
OPC_MIB_DIR	Specifies the path of MIB directory which contains the MIB definition files. Conversion of OID to textual representation happens from the MIB definition files provided in the directory. This variable is applicable for interception of all versions of traps SNMPv1, SNMPv2c and SNMPv3. Namespace: `eaagt`	NO	`<OvDataDir>/mibs`	String
OPCMONA_ CACHE_AND_ STORE_DATA_ INTERVAL	Specifies the maximum time Interval (in seconds) for **opcmona** to submit the metrics from cache to **oacore**. The minimum value you can set for this variable is 15. Namespace: `eaagt` **Note:** On HP-UX Precision Architecture (PA) and Linux PowerPC (PPC) architecture, it is recommended to set this variable to the default value of 15 seconds.	YES	15	Integer
OPCMONA_ CACHE_AND_ STORE_BULK_ INSTANCE_ COUNT	Specifies the maximum number of instances of a class for which **opcmona** submits the metrics in bulk from cache to **oacore**. The minimum value you can set for this variable is 1000. Namespace: `eaagt`	YES	1000	Integer

Table 4a List of Configuration Variables for the Operations Monitoring Component, continued , continued

Variable	Description	Restart Required	Default Value	Type
	Note: On HP-UX Precision Architecture (PA) and Linux PowerPC (PPC) architecture, it is recommended to set this variable to the default value of 1000.			
OPC_MSGTXT_USE_OUTGOING	If set to TRUE, the OPC_MSGTXT_USE_OUTGOING variable resolves <$MSG_TEXT> in the auto action field. Namespace: `eaagt`	NO	FALSE	Boolean
OPCMONA_POL_RUN_WITH_SHELL	This variable provides backward compatibility with respect to opcmona command execution. If this variable is set to TRUE, the command execution is with shell and if set to FALSE, the command execution is without shell. Namespace: `eaagt`	NO	TRUE	Boolean
SNMP_OVERRIDE_DISABLE_DNS *Introduced with the Operations Agent 12.00*	If set to TRUE, **opctrapi** will ignore the DISABLE_DNS configuration variable settings and will resolve $A with the appropriate hostname of the node. Namespace: `eaagt` **Note:** This configuration variable is only applicable for resolving $A value.	NO	FALSE	Boolean

Configuration Variables for Message Storm

You can use the variables listed in the table, to configure Operations Agent to detect and suppress the message storm on a managed node:

For more information, see *Operations Agent User Guide.*

Table 4b List of Configuration Variables for Message Storm

Variable	Namespace	Description	Restart Required	Default Value	Type
OPC_MSG_ STORM_ DETECTION	`eaagt.msgstorm`	Set the variable to TRUE, to enable the message storm detection. This variable is mandatory.	YES	FALSE	Boolean
OPC_MSG_ STORM_ DETECTION_ CATEGORY	`eaagt.msgstorm`	OPC_MSG_ STORM_ DETECTION_ CATEGORY is enabled only if the OPC_MSG_ STORM_ DETECTION variable is set to TRUE. This variable is mandatory.	YES		String
OPC_MSG_ STORM_RATE	`eaagt.msgstorm`	Defines the following parameters: Threshold, Time, and Reset. This variable is mandatory.	YES		Integer
OPC_SEND_ INTERNAL_MSG_ ON_MSGSTORM	`eaagt.msgstorm`	Decides whether to send or stop the internal messages.This variable is optional.	YES	TRUE	Boolean
OPC_SUPPRESS_ MSG_ON_MSG_ STORM	`eaagt.msgstorm`	This variable decides whether to send or suppress the messages. By default if the threshold condition is met and the message storm state is detected, all the	YES	TRUE	Boolean

Table 4b List of Configuration Variables for Message Storm, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		messages beyond the threshold value will be suppressed. This variable is optional.			
OPC_MSG_ STORM_TRACE_ SUPPRESSED_ MSGS	`eaagt.msgstorm`	This variable defines whether to log the messages to the log file only when the OPC_ SUPPRESS_ MSG_ON_MSG_ STORM is set to TRUE. This variable is optional.	YES	FALSE	Boolean

Configuration Variables for SNMP Trap Interceptor (NNMi Integration)

To set the message severity based on severity level available in SNMP trap, you can configure SNMP trap interceptor (opctrapi). Use the variables listed in the table to modify the default behavior of opctrapi:

For more information, see *Operations Agent User Guide.*

Table 4c List of Configuration Variables for SNMP Trap Interceptor (NNMi Integration):

Variable	Namespace	Description	Restart Required	Default Value	Type
OPC_ SNMP_ SET_ SEVERITY	`eaagt.integration.nnm`	Set the variable to TRUE, to enable the SNMP trap interceptor to read SNMP traps with the specific varbind OID (.1.3.6.1.4.1.11.2.17.19.	NO	FALSE	Boolean

Table 4c List of Configuration Variables for SNMP Trap Interceptor (NNMi Integration):, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		2.2.12) and set the severity of the message.			
OPC_SNMP_OVERRIDE_SEVERITY_OID	`eaagt.integration.nnm`	Use the variable to set a new Object Identifier (OID).	NO	.1.3.6.1.4.1.11.2.17.19.2.2.12	String
OPC_SPLIT_NNM_CUSTOM_ATTR	`eaagt.integration.nnm`	If the variable is set to TRUE, values of all NNMi CIAs present in the varbind (.1.3.6.1.4.1.11.2.17.19.2.2.20) will appear as individual Custom Message Attribute (CMA).	NO	FALSE	Boolean
OPC_SPLIT_NNM_CUSTOM_ATTR_MAX	`eaagt.integration.nnm`	Defines the number of NNMi custom attributes that can be read and interpreted by OM message. OPC_SPLIT_NNM_CUSTOM_ATTR_MAX is enabled only if the OPC_SPLIT_NNM_CUSTOM_ATTR is set to TRUE.	NO	20	Integer
OPC_NODENAME_FROM_NNM_FRWD_TRAP	`eaagt.integration.nnm`	If the variable is set to TRUE, all the traps are searched for the varbinds - .1.3.6.1.4.1.11.2.17.2.19.1.1.2.0 and .1.3.6.1.4.1.11.2.17.2.19.1.1.3.0.	NO	FALSE	Boolean

Configuration Variables for Health View

Use the variables listed in the table to modify the default behavior of Operations Agent Health View:

Table 4d List of Configuration Variables for Health View:

Variable	Namespace	Description	Restart Required	Default Value	Type
OPC_ SELFMON_ ENABLE	`agent.health`	Set this variable to TRUE to enable Health Monitoring on the node.	NO	FALSE	Boolean
OPC_ SELFMON_ SERVER	`agent.health`	IP address or hostname of the system where Operations Agent Health View dashboard is configured. By default, the management server is configured. Use this variable to modify the default Health View server.	NO	<Management Server IP address or hostname>	String
OPC_ SELFMON_ INTERVAL	`agent.health`	Defines the frequency at which the system health information is collected and exposed to the Health View Server.	NO	300	Integer
OPC_ SELFMON_ HBP	`agent.health`	Set this variable to FALSE to disable default health monitoring HBP configuration.	NO	TRUE	Boolean
OPC_HB_ MSG_ INTERVAL	`agent.health`	The variable specifies the time interval (in seconds) at which Operations Agent sends heartbeat messages to the management server. Only if this variable is enabled, the Operations Agent 12.xx can send heartbeat messages to the management server. The variable is disabled by default.	NO	-	Integer

Table 4d List of Configuration Variables for Health View:, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		Note: Operations Agent sends heartbeat messages to the management server only if there is no regular message sent within the heartbeat interval. Normal heartbeat messages are sent as log-only messages. If you stop Operations Agent, then a critical heartbeat message is sent.			

Configuration Variables for the Communication Component

The communication component of the Operations Agent enables you to establish communication with the agent node in a highly secure environment. You can use a group of configuration variable to modify the default behavior of the communication component.

Table 4e Configuration Variables for the Communication Components

Variable	Namespace	Description	Restart Required	Default Value	Type
CHROOT_PATH	`bbc.cb`	This variable is in effect only on UNIX/Linux nodes. The chroot path for the **ovbbccb** process. This variable provides a way to protect the Communication Broker port.	YES	`/var/opt/OV`	String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		If this parameter is set, the **ovbbccb** process performs the chroot operation to this path. As a result, the file system above the `<OvDataDir>` directory becomes invisible to the **ovbbccb** process. Therefore, all files in the `/etc` directory become inaccessible. For example: `/etc/hosts`, `/etc/resolv.conf`, and `/etc/nsswitch.conf`. Communication Broker parameters such as SERVER_BIND_ADDR must therefore use IP addresses instead of hostnames when the CHROOT_PATH parameter is active. **Note:** If you change the value of this variable, **ovbbccb** is restarted automatically.			
SSL_ REQUIRED	`bbc.cb`	If this parameter is set to true, the communication broker component will require SSL authentication for all administration connections to the communication broker on other machines. Otherwise, non-SSL administration connections will be allowed to the communication broker itself. See also the	YES	TRUE	Boolean

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		ENFORCE_CLIENT_PROTOCOL and ENFORCE_SERVER_SSL parameters in the `[bbc.http]` namespace.			
SERVER_PORT	`bbc.cb`	Use this variable to use a non-default port for the communication broker. By default, the communication broker uses the port 383. If the port 383 is not free on the agent node, set this variable to a port that is available for use. **Note:** If you change the value of this variable, **ovbbccb** is restarted automatically.	YES	383	Integer
LOCAL_CONTROL_ONLY	`bbc.cb`	If this parameter is set to true, the communication broker will only allow local connections to run administrative commands such as start, stop, kill, or reinit.	YES	TRUE	Boolean
LOCAL_INFO_ONLY	`bbc.cb`	If this parameter is TRUE, the communication broker will only allow local connections to retrieve details like status information, registered services, or started resource groups.	YES	FALSE	Boolean
RESTRICT_REG	`bbc.cb`	This variable helps you restrict programs from registering with the	YES	FALSE	Boolean

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type

		communication broker.
		If the variable is set to true, only the programs with the write access to the following folder can register with the communication broker:
		On Windows:
		`%ovdatadir%temp\bbc`
		On UNIX/Linux:
		`/var/opt/OV/tmp/bbc`			

REQUEST_ TIMEOUT	`bbc.cb`	Specifies the number of seconds the **ovbbccb** server (system where the communication broker is running) waits for data on an incoming request. If the data is not received within the specified number of seconds, the request is requeued.	YES	1	Integer
LOCAL_ CONTROL_ ONLY	`bbc.cb`	If this parameter is TRUE, the Communication Broker will only allow local connections to execute administrative commands such as start and stop, reinit or kill.	YES	true	Boolean
ENABLE_ REVERSE_ ADMIN_ CHANNELS	`bbc.cb`	This parameter enables/disables RAC on the Communication Broker server. To enable RAC, set the variable to TRUE and to disable RAC, set the variable to FALSE.	YES		Boolean
RC_	`bbc.cb`	This is alternative to	NO	NULL	String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
CHANNELS_ CFG_FILES		configuration variable RC_CHANNELS where details like Reverse Channel Proxy Name and port are stored in File instead of XPL config settings.			
RC_MAX_ WORKER_ THREADS	`bbc.cb`	The maximum number of threads that can be used by the Communication Broker component while establishing a Reverse Admin Channel.	NO	1	Integer
RC_MIN_ WORKER_ THREADS	`bbc.cb`	The minimum number of threads that will always remain active on the node while the Communication Broker component establishes a Reverse Admin Channel.	NO	0	Integer
RETRY_RC_ FAILED_ CONNECTION	`bbc.cb`	Use this option to enable the Communication Broker component to retry a failed attempt to connect to a reverse channel proxy (RCP).	NO	FALSE	Boolean
GENERATE_ OVEVENT_ FOR_FAILED_ RC_NODES	`bbc.cb`	This option enables you to send messages to the OM message browser when the status of an RCP node is FAILED.	NO	FALSE	Boolean
CB_PORTS_ CFG_FILE	`bbc.cb .ports`	This is an alternative to the variable CB_PORTS where the entries information is stored in File instead of configuration settings.	NO	NULL	String
AUTO_ CONNECTION_ CLOSE_	`bbc. http, bbc.http`	This defines the interval after which an application	NO	-1(inactive)	Integer

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
INTERVAL	`.ext.*.`	with inactive connections from connection pool will be closed. This setting will affect applications using the HTTPS communication APIs.			
LOCAL_INFO_ONLY	`bbc.cb`, `bbc.http`	Boolean parameter that specifies whether the Communication Broker may answer requests for information from remote hosts. If this parameter is true, no information will be sent (only a ping gets answered). Affected are requests for the status of the server, registered services, running resource groups.	YES	False	Boolean
PORTS	`bbc.cb .ports`	This variable defines the list of ports for all communication brokers on the network that may be contacted by applications on this host. This configuration parameter must be the same on all managed nodes and the associated management server. If multiple systems in the OM-managed environment use non-default `bbc.cb` ports, you can set this variable to the list of ports, separated by commas, in the following manner: *<system1>:<port1>, <system2>:<port 2>*,...	YES		String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		For example, if the nodes `system1.domain.com` and `system2.domain.com` use the ports 400 and 401, respectively, for the communication broker, set the `PORTS` variable on all agent nodes and the management server to `system1.domain.com:400, system2.domain.com:401`. You can use IP addresses instead of the fully-qualified domain names. You can also use the wildcard character (*) when you want to specify a group of systems. For example, `*.domain.com:400` indicates all the systems with the domain " `domain.com` " use 400 as the `bbc.cb` port.			
SERVER_ PORT	`bbc.http`	By default, this port is set to 0. If set to 0, the operating system will assign the first available port number. This is the port where the application will wait for requests. **Note:** It is recommended to explicitly set this parameter in the `bbc.http.ext.<app Name>` namespace.	YES	0	Integer

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
SERVER_BIND_ADDR	`bbc.cb, bbc.http, bbc.rcp, bbc.http.ext.*`	Specifies the bind address of a http server. **Note:** If you change the value of this variable, **ovbbccb** is restarted automatically.	YES	localhost	String
MAX_CONNECTIONS	`bbc.http`	Specifies the maximum number of connections a node can accept. The default value on UNIX is the maximum number of file descriptors per process minus 30%. The default on Windows is 2000. If it is set to 0, the default values are used.	YES	0	Integer
CLIENT_PORT	`bbc.http`	The bind port for the client requests. This may also be a range of ports, for example, 10000-10020. The bind port belongs to the node from where the request originates. Default is the port '0'. The operating system will assign the first available port. **Note:** The Windows systems do not immediately release ports for reuse. This parameter should be set to a large range on Windows systems.	YES	0	String
CLIENT_BIND_ADDR	`bbc.http`	Bind address for the client requests.	YES	INADDR_ANY	String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
LOG_SERVER_ACCESS	`bbc.http`	If set to true, the Operations Agent logs every access to the server providing information about sender's IP, requested HTTP address, requested HTTP method, and the response status.	YES	FALSE	Boolean
ENFORCE_CLIENT_PROTOCOL	`bbc.http`	This parameter enables you to set the communication protocols for client requests. This parameter may be set to one of the following values: HTTP: All client requests will use the HTTP protocol. HTTPS: All client requests will use the HTTPS protocol. This parameter is ignored if set to any other value. The HTTP client will then use the protocol specified by the application when creating HTTP requests. The parameter is case insensitive. **Note:** Use caution when setting this parameter as it will disable security features if set to 'HTTP'.	YES	HTTPS	Boolean
ENFORCE_SERVER_SSL	`bbc.http`	This parameter controls the connections allowed at the HTTP server.	YES	ALL	String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		This parameter may be set to one of the following values: NONE: Both SSL and non-SSL connections will be accepted by the HTTP server. REMOTE: All remote connections to the HTTP server must use SSL. Remote connections that do not use SSL will be automatically rejected. Local connections may use SSL or non-SSL. ALL: All connections to the HTTP server must use SSL. Connections that do not use SSL will be rejected automatically. This parameter is ignored if set to any other value. The HTTP server will then use the authentication specified by the application that created the HTTP server. This parameter is not case sensitive. **Note:** Use caution when setting this parameter as it will disable security features if set to 'NONE' or 'REMOTE'.			
LOCAL_INFO_ONLY	`bbc.cb`, `bbc.http`	Boolean parameter that specifies whether the CB may answer requests for	YES	TRUE	Boolean

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		information from remote hosts. If this parameter is TRUE then no information is sent (only a ping gets answered). Affected are requests for the status of the server, registered services, and running resource groups.			
PROXY	`bbc.http`	Defines which proxy and port to use for a specified hostname. Format: proxy:port +(a)-(b);proxy2:port2+(a)-(b); ... a: Comma- or semicolon-divided list of hostnames for which this proxy can be used. b: Comma- or semicolon-divided list of hostnames for which the proxy cannot be used. The Operations Agent chooses the first matching proxy. Example: PROXY=web-proxy:8088-(*.hp.com)+(*.domain.hp.com;*) The proxy 'web-proxy' will be used with the port 8088 for every server (*) except hosts that match *.hp.com (for example, www.hp.com). If the hostname matches *.domain.hp.com (for example, machine1.domain.hp.com), the proxy server will	YES		String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		be used. It is also possible to use IP addresses instead of hostnames. Therefore, 15.*.*.* or 15:*:*:*:*:*:*:* will be valid.			
PROXY_CFG_FILE	`bbc.http`	Instead of specifying large expressions for the PROXY variable, you can use an external configuration file to specify all the proxy server-related details and set the PROXY_CFG_FILE variable to the location of the configuration file. You must create the external configuration file in the XML format. For more information on the structure of this XML file, see the *PROXY_CFG_FILE Parameter Syntax* section in the *Operations Agent (11.02) Deployment Guide*. This variable takes precedence over the PROXY variable.	YES		String
DOMAIN	`bbc.http`	Default DNS domain to use if no domain is specified for a target host. This domain name will be appended to hostnames without a DNS domain name if a match for the hostname alone cannot be found. This can be done for PROXY lookups	YES		String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		and lookups in the [cb.ports] table. For example, if the hostname machine is specified and if the DOMAIN is `domain.hp.com`, the [cb.ports] entries will first be searched for the match of machine. If there is no match found for the hostname machine, a search will be made for `machine.domain.hp.com` , `*.domain.hp.com` , `*.hp.com` , `*.com`, and `*` in the given order.			
FX_MAX_ RETRIES	`bbc.fx`	The maximum number of retries to be attempted for the successful transfer of the object.	YES	3	Integer
FX_BASE_ DIRECTORY	`bbc.fx`	The base directory for which files may be uploaded or downloaded.	YES	<OvDataDir>	String
FX_TEMP_ DIRECTORY	`bbc.fx`	The temporary directory where uploaded files will be placed while the upload process is in progress. When the upload process is complete, the file will be moved to the FX_ UPLOAD_DIRECTORY directory.	YES	<OvDataDir> /tmp/bbc/fx	String
FX_UPLOAD_ DIRECTORY	`bbc.fx`	The target directory for uploaded files. The upload target directory may be overridden with this configuration parameter.	YES	FX_BASE_ DIRECTORY	String
BUFFER_PATH	`bbc.snf`	Specifies the SNF path where the buffered	YES	<OvDataDir>	String

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		requests are stored.		/datafiles/ bbc/snf/ <appName>	
CLEANUP_ INTERVAL	`bbc.snf`	This variable specifies the interval (in seconds) at which the Operations Agent cleans up the unnecessary buffer files on the system.	YES	100	Integer
MAX_FILE_ BUFFER_SIZE	`bbc.snf`	Specifies the maximum amount of disk space available for the buffer to consume on the hard disk. By default, this parameter is set to 0, which means no disk space limit is set to the buffer.	YES	0	Integer
DELIVERY_ INTERVAL	`bbc.snf`	Defines the interval in which the component tries to deliver the requests that are stored in the buffer Default: 1000 milliseconds	YES	1000	Integer
MAX_ DELIVERY_ THREADS	`bbc.snf`	Defines the maximum number of threads that are started simultaneously to deliver the messages.	YES	5	Integer
KEEP_ CONNECTIONS_OPEN	`bbc.snf`	If set to "true," the Snf Client does not close the connections after a queue has been processed. The connections will stay open until SnfClient object is alive. **Note:** To increase the message throughput on	YES	false	boolean

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		management servers, it is recommended to set the variable to TRUE.			
MAX_INPUT_BUFFER_SIZE	`bbc.snf`	Defines the size of the internal message buffer of a SnfOutputRequest object. Up to this size, the component tries to deliver a message directly. Default: 100 KB	YES	100	Integer
DELIVERY_QUEUE_METHOD	`bbc.snf`	Defines the order in which the requests are delivered; two possibilities are offered: FIFO: Messages are delivered in the temporal order. PRIORITY: Messages with the high priority are sent first.	YES	FIFO	String
SNF_CONTENT_TYPE	`bbc.snf`	Use this parameter to specify a content type other than application/octetstream for ALL the requests sent by the Snf Client.	YES		Integer
RC_HEALTH_CHECK_INTERVAL	`bbc.rcp`	After a Reverse Administration Channel (RAC) is established, the Operations Agent continuously performs checks to monitor the health of the RAC. This variable defines the interval (in seconds) at which the agent performs this check. If you set this variable to -	YES	60	Integer

Table 4e Configuration Variables for the Communication Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		1, no health check is performed.			
MAX_ SOCKETSET_ THREADS	`bbc.http`	Specifies the maximum number of threads serving a pool of sockets. **Note:** To increase the message throughput on management servers, it is recommended to set the MAX_ SOCKETSET_ THREADS variable to 3 and SOCKETS_ PER_SOCKETSET variable to 10.	YES	6	Integer
RC_IDLE_ TIMEOUT	`bbc.cb`	This parameter remains inactive for a specified time, then checks if the connection is IDLE, if yes, it closes the Reverse Channel (RC) connection.	YES	0	Integer
USE_PROXY_ CACHE	`bbc.http`, `bbc.http.ext.*`	If this parameter is set to TRUE, the proxy resolution is cached to receive a faster response for subsequent requests.	YES	FALSE	Boolean
DISABLE_REL_ PATH	`bbc.fx`	Use this variable to prevent the applications, which consumes **ovbbc** and **ovdeploy** APIs, not to use relative paths while using the file transfer mechanism.	YES	FALSE	Boolean

Configuration Variables for the Security Component

The Operations Agent includes a certificate client and keystore to enable secure communication between the OM management server and nodes. You can use a group of configuration variable to modify the default behavior of the security components.

Table 4f Configuration Variables for the Security Components

Variable	Namespace	Description	Restart Required	Default Value	Type
CERTIFICATE_SERVER	`sec.cm.client`	The certificate server configured for the node. A certificate server is the system that receives certificate requests from the node and issues certificates to nodes. It could be the management server that manages the node, or a dedicated server that can grant certificates to the node to facilitate secure communication between the node and the management server.	YES	When you remotely install the agent on the node from the OM console, the variable is automatically set to the FQDN of the management server.	String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
				When you install the agent manually on a node and do not configure a management server with it, the variable is not set to any value.	
MANAGER	`sec.core.auth`	Name (FQDN) of the management server configured for the node.	YES		String
CERTIFICATE_DEPLOYMENT_TYPE	`sec.cm.client`	The type of certificate deployment to the node. Possible values are: **Automatic:** When set to `automatic` , the node automatically requests the certificate server to issue a new certificate when required. **Manual:** When set to `manual` , the certificate must be installed on the node manually.	YES	If you install the agent in an OM-managed	String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
				environment, the variable is set to `automatic` or `manual` based on the settings on the OM management server that was configured with the node. If you do not configure the agent with an OM	

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
				management server, the variable is not set to any value.	
ENCRYPTION_LEVEL	`sec.core.ssl`	Data encryption level for the data exchanged between the node and the management server. Possible values are: • Full • Best • Export • None	YES	Full	String
CLIENT_VERIFICATION_MODE	`sec.core.ssl`	The SSL client verification mode. Possible values are: • Anonymous • RequireCertificate If set to `Anonymous`, the node will receive messages that are not encrypted from different sources.	YES	RequireCertificate	String
SESSION_CACHING	`sec.core.ssl`	When a node starts communicating with a management server or another node in the secure mode, a session is created. This variable helps you retain a session for some time. Possible values are:	YES	Enabled	String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		- Enabled - Disabled Setting the variable to `Enabled` helps you retain a session for some time. Setting the variable to `Disabled` will cause the session to break after one cycle of data communication.			
SESSION_TIME_OUT	`sec.core.ssl`	Valid only if you set `SESSION_CACHING` to enabled. This variable helps you set the duration (in seconds) for which a session cache can survive.	YES	7200	Integer
SESSION_CACHE_SIZE	`sec.core.ssl`	Valid only if you set `SESSION_CACHING` to enabled. When you enable `SESSION_CACHING` , multiple sessions are stored in the cache until they exceed the `SESSION_TIME_OUT` value. This variable helps you set the upper limit of the number of sessions to be stored in the cache.	YES		Integer
RANDOM_FILENAME	`sec.core.ssl`	Setting to configure a file that will be used to feed the random number generator (absolute path name).	YES		String
RANDOM_FILE_BYTES_TO_READ	`sec.core.ssl`	Setting to configure how many bytes are read from the file specified with `RANDOM_FILENAME`.	YES	1024	Integer
COMM_PROTOCOL	`sec.core.ssl`	Use this variable to set the protocol to be used for communication. The values supported for COMM_PROTOCOL are TLSv1.2, TLSv1.1 and	YES		String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		TLSv1 If you set TLSv1.2, then only TLSv1.2 is used for secure communication. If you set TLSv1.1, then TLSv1.2 and TLSv1.1 are used for secure communication. If you set TLSv1, then TLSv1.2, TLSv1.1 and TLSv1 can be used for secure communication. **Note:** If any value other than the supported values are set, then communication over all protocols are allowed.			
ASYMMETRIC_KEY_LENGTH	`sec.cm`	Use this variable to specify the asymmetric key length for creating an RSA key pair. A popular algorithm for public-key encryption is the RSA. Using an RSA key length of 2048 increases the strength of the encryption. Asymmetric key encryption is much slower than symmetric key encryption and hence is used only for digital signatures and key exchanges. Possible RSA values are: • 2048 • 3072	YES	2048	Integer
DEF_SYM_KEY_ALGO	`sec.core`	Use this variable to set the default symmetric key algorithm for encryption. The supported algorithm values are eBlowfish, eDES, eDES3, eAES128, eAES192, eAES256, and eDefault (uses AES128 as the default algorithm). The MigrateSymKey tool is used to migrate the contents of an existing keystore, based on the algorithm set for the DEF_SYM_KEY_ALGO variable.	NO	AES128	String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
ENABLE_DEF_SYM_KEY_ALGO	`sec.core`	Use this variable to enable the use of the default symmetric key algorithm set to DEF_SYM_KEY_ALGO. The supported value is TRUE and for any other value it is considered as not set.	NO	TRUE	Boolean
HASH_ALGO	`sec.core`	Use this variable to set the hash algorithm.	NO	eSHA256	String
CORE_ID	`sec.core`	Specifies the unique identifier (OvCoreId) of a managed node.	YES		String
MANAGER_ID	`sec.core.auth`	Specifies the OvCoreId of a remote system that is allowed to contact the local node.	YES		String
CERT_INSTALLED	`sec.cm.certificates`	Specifies a Boolean value that is set to TRUE, when a valid node certificate is installed. The value is set to FALSE when no valid node certificate is installed.	YES		Boolean
LAST_CERT_UPDATE	`sec.core.auth`	Defines a value that is set to the actual date and time, whenever the node certificate gets updated (installed, reinstalled, or removed).	YES		String
NOT_BEFORE_OFFSET	`sec.cm.server`	Use the variable to set the validity of the certificate, when a new certificate is generated. **For example:** If the certificate is generated at 15:00 and the value of the configuration variable is set to 3600 seconds: NOT_BEFORE_OFFSET = 3600, the certificate will be valid from 14:00. The value of this variable is set to synchronize the node with other time zones across	YES		Integer

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		geographies.			
SEED_FILE	`sec.core`	Specifies a file that is used to provide additional seed data for encryption. After setting this value, you may get decryption errors, if data was encrypted without using the additional seed file. If the variable is not set, the default location: `<OvDataDir>/conf/sec/seed.dat` will be checked.	YES		String
eaagt.actr	`sec.core.auth.mapping.actionallow`	Enables the backup manager to execute an action on a node.	YES	1	Integer
ENABLE_DEF_SYM_KEY_ALGO	`sec.core`	Enables the use of the default Symmetric Key Algorithm set for the configuration variable: DEF_SYM_KEY_ALGO, for data encryption. If ENABLE_DEF_SYM_KEY_ALGO variable is not set by default, use Blowfish as the default Symmetric Key Algorithm.	YES	FALSE	Boolean
DEF_SYM_KEY_ALGO	`sec.core`	Use the variable to choose the default Symmetric Key Algorithm used for data encryption. If DEF_SYM_KEY_ALGO variable is not set by default, use AES-128 bit algorithm as the default Symmetric Key Algorithm.	YES		String
POLICYSIGNER_ID	`sec.core.auth.custom`	Use the variable to upload policies other than the primary manager. If POLICYSIGNER_ID is set, then the policies signed by the IDs mentioned along with the variable are also uploaded. This can be used for custom SPI and factory signed policies. Multiple signers must be separated by whitespaces. **For example:** `POLICYSIGNER_ID=ID1 ID2` **Note:** By default, the ID in the **Issuer CN**	YES		String

Table 4f Configuration Variables for the Security Components, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		(of the certificate) is used to upload policies.			
POLICY_AUTH_RETRY_COUNT	`sec.core.auth`	Defines the number of attempts required (with one second sleep between each attempt) to check whether `mgrconf` policy(`OVO` authorization) is deployed or not. In some cases, it is noticed that due to network delay, the `mgrconf` policy is not always the first policy to be installed on the node. In such cases, you can set this variable to wait for few additional seconds before reading `mgrconf` policy.	NO	0	Integer

SSL_CIPHER_LIST	`sec.core.ssl`	Use this variable to make the communication ciphers used by the **ovbbccb** and the **http** client provided by the Operations Agent configurable.
		For example:
		The following configuration is made to configure the communication ciphers:
		`[sec.core.ssl]`
		`SSL_CIPHER_LIST= HIGH:MEDIUM:!DH:!ADH:!SSLv2:+SHA:!MD5:+RSA:+DSS:!RC4:+IDEA:!3DES:!RC2:+DES`
			YES		String

Configuration Variables for the oacore Process

You can use the variables listed in the table to configure the default behavior of the **oacore** process provided by the Performance Collection Component:

Table 4g Variables for the oacore Process

Variable	Namespace	Description	Restart Required	Default Value	Type
READ_ONLY_ MODE	`oacore.dml`	If this variable is set to TRUE, data logging to the Metrics Datastoreis completely disabled. You can only perform read operations from the datastore.	YES	FALSE	Boolean
UPDATED_ MODEL_ AVAILABLE	`oacore`	If this variable is set to TRUE, it instructs **oacore** to re-read and re-create Model from XML files present under `<DataDir>/conf/oa/Model`.	YES	FALSE	Boolean
ENABLE_ BASELINE	`oacore`	Set this variable to TRUE to enable baselining.	YES	FALSE	Boolean
DATAMATRIX_ VERSION	`coda`	You can set this variable to 1 or 0. When the variable is set to 1, the agent breaks the data into chunks of five records, and then sends the data to the client program (such as Performance Manager). Breaking the data into chunks enhances the performance of data transfer process.	YES	1	Integer
DATAMATRIX_ ROWCOUNT	`coda`	If the DATAMATRIX_ VERSION variable is set to 1, use this variable to control the number of records that the agent can transfer to the client with every chunk.	YES	5	Integer
SSL_ SECURITY	`coda`	This parameter controls the connections at Coda-HTTP server for remote and local nodes. This parameter may be set to one of the following values: REMOTE: All remote connections to the Coda-HTTP server must use SSL. Coda-HTTP server	YES		String

Table 4g Variables for the oacore Process, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		automatically rejects the remote connections that do not use SSL. Local connections may use SSL or non-SSL. ALL: All connections to the Coda-HTTP server must use SSL. Coda-HTTP server automatically rejects the connections that do not use SSL. NONE: Coda-HTTP server can accept both the SSL and non-SSL connections. This parameter is ignored if set to any other value. The HTTP server will then use the authentication specified by the application that created the HTTP server. This parameter is not case sensitive. **Note:** Use caution when setting this parameter as it will disable security features if set to 'NONE' or 'REMOTE'.			
SERVER_ PORT	`coda.comm`	Specifies the HTTP server port for CODA module.	YES	0	Integer
RELEASE_ MEMORY	`oacore.sqlite`	This variable helps in releasing **oacore** memory unused by SQLite per connection. If set to TRUE, **oacore** periodically releases unused memory held by SQLite.	YES	FALSE	Boolean

Configuration Variables for the hpsensor Process

You can use the variables listed in the table to configure the default behavior of the **hpsensor** process provided by the Performance Collection Component:

Table 4h Variables for the hpsensor Process

Variable	Namespace	Description	Restart Required	Default Value	Type
ENFORCE_SERVER_SSL	`hpsensor`	This parameter controls the connections allowed at the HTTP server. This parameter may be set to one of the following values: NONE: Both SSL and non-SSL connections will be accepted by the HTTP server. REMOTE: All remote connections to the HTTP server must use SSL. Remote connections that do not use SSL will be automatically rejected. Local connections may use SSL or non-SSL. ALL: All connections to the HTTP server must use SSL. Connections that do not use SSL will be rejected automatically. This parameter is ignored if set to any other value. The HTTP server will	YES	ALL	String

Table 4h Variables for the hpsensor Process, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		then use the authentication specified by the application that created the HTTP server. This parameter is not case sensitive. - Use caution when setting this parameter as it will disable security features if set to 'NONE' or 'REMOTE'. - This variable is also available under `hpcsrvd` namespace for the **HPCS Server** process.			
PUBLISH_ INTERVAL	`hpsensor`	This parameter sets the interval at which custom and system performance metrics data is published to the target subscriber.	NO	10	Integer

Configuration Variables for the Cross-Platform Component

You can use the variables listed in the table to configure the default behavior of the cross-platform component:

Table 4i Variables for the Cross-Platform Component

Variable	Namespace	Description	Restart Required	Default Value	Type
filecount	`xpl.log.OvLogFileHandler`	The maximum number of log files to be created for any given logging entity.	YES	10	filecount
filesize	`xpl.log.OvLogFileHandler`	The file size limit for any log file in millions of bytes. Whenever a log entry is written causing the log file to exceed this value, a new log file is created	YES	1	filesize
IsBindAny	`xpl.trc.server`	This attribute represents the trace server bind address (INADDR_ANY or localhost). If the value is NO, the bind address is localhost. If the value is YES, the bind address is INADDR_ANY	YES	Y	IsBindAny
server	`xpl.dir.shares`	Defines base directories for resource groups	YES		server

Table 4i Variables for the Cross-Platform Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
SocketPoll	`xpl.net`	Flag indicating whether to use poll() or select() system call. Set this variable to `TRUE` on the managed node and the management server if you are monitoring a large environment (that is, an environment with more than 1024 nodes). **Tip:** To configure this variable on all managed nodes at the time of agent installation, use profile file-based installation. Add the following statement in the profile file before installation: `set xpl.net:SocketPoll=true`	YES	False	SocketPoll
IPCACHE_DISABLED	`xpl.net`	`getaddrinfo()` is a system call used to resolve	YES	False	Boolean

Table 4i Variables for the Cross-Platform Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		hostname to IP address. XPL caches the name resolution done by this system call. To disable it set IPCACHE_DISABLED to TRUE and then restart the agent.			
HOSTNAME_CACHE_TIMEOUT	`xpl.net`	This configures the timeout for the cache in seconds. By default it is 30 Minutes (1800 seconds). If the value is negative then the timeout is considered as infinite.	YES	1800 seconds	Integer
SOCKETS_PER_SOCKETSET	`xpl.net`	Specifies the number of sockets created in a socket pool. New socket pools are created for new incoming connections as and when required. **Note:** To increase the message throughput on management servers, it is recommended to set MAX_	YES	60	Integer

Table 4i Variables for the Cross-Platform Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		SOCKETSET_ THREADS variable to 3 and SOCKETS_ PER_ SOCKETSET variable to 10.			
addlocales	`xpl.log`	Use the variable to add any number of locales separate by whitespaces.	YES		String
DISABLE_ DNS	`xpl.net, xpl.net.ext.*`	If this variable is set to TRUE, reverse address lookup is not performed. The conversion from address to name can take longer time in some environments. **Note:** Use caution when you set this variable as it can cause problems if proxy server requires a reverse lookup.	NO	FALSE	Boolean
LOCAL_ NODE_	`xpl.net`	The value set to the variable is	YES		String

Table 4i Variables for the Cross-Platform Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
NAME		used as the local hostname of the system.			

Configuration Variables for the Configuration Component

You can use the variables listed in the table to configure the default behavior of the configuration component:

Table 4j Variables for the Configuration Component

Variable	Namespace	Description	Restart Required	Default Value	Type
CLUSTER_TYPE	`conf.cluster`	The runtime cluster solution determined during the **ovconfd** startup is set with this variable.	YES	Depending on the type of cluster solution installed, one of the following strings will be set: VERITAS Cluster Server (VCS) Sun Cluster (SC) MC/Service Guard (MC/SG) AIX Cluster (HACMP) Red Hat	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
				Advanced Server (RHAS) Microsoft Cluster Server (MSCS)	
MONITOR_MODE	`conf.cluster`	Cluster monitoring will be enabled based on the value set with this configuration variable.	YES	TRUE	Boolean
POLLING_ INTERVAL	`conf.cluster`	The status of the cluster will be checked for every polling interval specified with this variable.	NO	10000	Integer
CLUSTER_LOCAL_ NODENAME	`conf.cluster`	For an agent node that belongs to a high availability cluster, the node hostname in the agent configuration can be different from the node hostname in the cluster configuration. In that case, set this variable to the hostname of	YES	NULL	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		the node that is used in the cluster configuration. If the node belongs to an HA cluster, it is necessary that you set the `MAX_RETRIES_FOR_CLUSTERUP` variable on the node to an integer value. The profile file-based installation ensures that the variable is set to an appropriate value on every node at the time of installation.			
MAX_RETRIES_FOR_CLUSTERUP	`conf.cluster`	The value set in this variable is the number of times the cluster is being checked for availability when **ovconfd** is being started either during boot-time or regular agent start with a 35 seconds gap between each try.	YES	1	Integer
MERGED_POLICY_LIST_FILENAME	`conf.core`	The file name used for writing	YES	ov_policies.txt	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		a list of all installed policies (ovpolicy -dump)			
FORMAT_POLICY_LIST	`conf.core`	Format the policy list (ovpolicy -list) if the policy name is not displayed correctly (for example, the name is too long).	YES	FALSE	Boolean
ASYNC_CONTROL_NOTIFY	`conf.core`	Defines if the notification trigger from the control service should be executed asynchronously. If set to TRUE, no status or error messages will be reported to config.	YES	FALSE	Boolean
CACHE_CONFIGSETTINGS_POLICIES	`conf.core`	Specifies if the policies for 'configsettings' policy type are cached in memory.	YES	TRUE	Boolean
AUDIT_LOGGING	`conf.server`	Toggles security audit logging for policy and configuration settings.	YES	FALSE	Boolean

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
AUDIT_LOG_MODE	`conf.server`	Toggles the audit log level ; FAILURE = log security failures only, ALL = log all (this is the default and fallback value)	YES	ALL	String
LOCATE_SERVER	`conf.server`	Number of retries for the startup check	YES	5	Integer
PING_SERVER	`conf.server`	The **ovconfd** process pings itself at the time of startup. This variable sets the number of pings performed by **ovconfd** for the startup check.	YES	15	Integer
WAIT_TIME	`conf.server`	Time (in seconds) between retries for the startup check by **ovconfd**.	YES	3	Integer
NOMULTIPLEPOLICIES	`conf.server`	The policy types mentioned in `NOMULTIPLEPOLICIES` will be allowed to have only one policy (of that type) to be installed on the node. The value of `NOMULTIPLEPOLICIES` can be a comma-	YES	NULL	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		separated list of policy types.			
ONLINE	`conf.cluster.RGState.VCS`	Resource group `Online` state setting for Veritas Cluster	YES	online	String
OFFLINE	`conf.cluster.RGState.VCS`	Resource group `offline` state setting for Veritas Cluster	YES	offline	String
PARTIAL	`conf.cluster.RGState.VCS`	Resource group `Partial` state setting for Veritas Cluster	YES	unknown	String
UNKNOWN	`conf.cluster.RGState.VCS`	Resource group `Unknown` state setting for Veritas Cluster	YES	unknown	String
up	`conf.cluster.RGState.MCSG`	Resource group `Up` state setting for MCSG Cluster	YES	online	String
down	`conf.cluster.RGState.MCSG`	Resource group `Down` state setting for MCSG Cluster	YES	offline	String
starting	`conf.cluster.RGState.MCSG`	Resource group `Starting` state setting for MCSG Cluster	NO	unknown	String
halting	`conf.cluster.RGState.MCSG`	Resource group `Halting` state setting for MCSG Cluster	YES	unknown	String
ClusterGroupStateUnknown	`conf.cluster.RGState.MSCS`	Resource group `Unknown` state	YES	unknown	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		setting for Microsoft cluster			
ClusterGroupOnline	`conf.cluster.RGState.MSCS`	Resource group `Online` state setting for Microsoft cluster	YES	online	String
ClusterGroupOffline	`conf.cluster.RGState.MSCS`	Resource group `Offline` state setting for Microsoft cluster	YES	offline	String
ClusterGroupFailed	`conf.cluster.RGState.MSCS`	Resource group `Failed` state setting for Microsoft cluster	YES	offline	String
ClusterGroupPartialOnline	`conf.cluster.RGState.MSCS`	Resource group `Partial Online` state setting for Microsoft cluster	YES	offline	String
UNMANAGED	`conf.cluster.RGState.SC`	Resource group `Unmanaged` state setting for Sun Cluster	YES	unknown	String
ONLINE	`conf.cluster.RGState.SC`	Resource group `Online`state setting for Sun Cluster	YES	online	String
OFFLINE	`conf.cluster.RGState.SC`	Resource group `Offline` state setting for Sun Cluster	YES	offline	String
PENDING_ONLINE	`conf.cluster.`	Resource group	YES	unknown	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
	RGState.SC	Pending Online state setting for Sun Cluster			
PENDING_ OFFLINE	conf.cluster. RGState.SC	Resource group Pending Offline state setting for Sun Cluster	YES	unknown	String
ERROR_STOP_ FAILED	conf.cluster. RGState.SC	Resource group error stop failed state setting for Sun Cluster	YES	unknown	String
started	conf.cluster. RGState.RHAS	Resource group error stop failed state setting For Red Hat (Linux)	YES	online	String
ONLINE	conf.cluster.RGState.VCS	Defines the resource group online state setting for Veritas Cluster.	YES	Online	String
OFFLINE	conf.cluster.RGState.VCS	Defines the resource group Offline state setting for Veritas Cluster.	YES	Offline	String
ACQUIRING	conf.cluster.RGState.HACMP	Defines the resource group offline state setting for High Availability Cluster Multi-Processing (HACMP) Cluster.	YES	Offline	String

Table 4j Variables for the Configuration Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
ERROR_ SECONDARY	`conf.cluster.RGState.HACMP`	Defines the resource group error of secondary state setting for HACMP Cluster.	YES	Unknown	String
OFFLINE_ SECONDARY	`conf.cluster.RGState.HACMP`	Defines the resource group offline secondary state setting for HACMP Cluster.	YES	Offline	String
ONLINE_ SECONDARY	`conf.cluster.RGState.HACMP`	Defines the resource group online secondary state setting for HACMP Cluster.	YES	Online	String
RELEASING	`conf.cluster.RGState.HACMP`	Defines the resource group releasing state setting for HACMP Cluster.	YES	Offline	String
UNMANAGED_ SECONDARY	`conf.cluster.RGState.HACMP`	Defines the resource group unmanaged secondary state setting for HACMP Cluster.	YES	Unknown	String

Configuration Variables for the Control Component

You can use the table to change the variable settings of the control component:

Table 4l Variables for the Control Component

Variable	Namespace	Description	Restart Required	Default Value	Type
RUN_PROFILE	`ctrl`	If set to true, the profile of the user, which runs `Ctrl`, is also executed.	YES	false	Boolean
START_ON_BOOT	`ctrl`	If the value is true, the `Ctrl` service gets started at reboot.	YES	false	Boolean
ACTION_ TIMEOUT	`ctrl.ovcd`	This is the time-out period (in seconds) for an action that is initiated by the **ovcd** process.	YES	60	Integer
PROCESS_ TIMEOUT	`ctrl.ovcd`	The time period (in seconds) within which every process must establish a particular state. During state transitions, if a process does not reach a state within this time period, **ovcd** reports the process as an aborted or abnormal process.	YES	120	Integer
KILL_TIMEOUT	`ctrl.ovcd`	The time-out period in seconds before killing a monitored process forcefully in case the process does not exit normally.	YES	15	Integer
MONITOR_ CHECK_ INTERVAL	`ctrl.ovcd`	The interval in milliseconds in which `Ctrl` will scan the OS to monitor an already monitored process.	YES	2000	Integer
MONITOR_ TIMEOUT	`ctrl.ovcd`	The interval in milliseconds within which `Ctrl` scans the OS whether any	YES	30000	Integer

Table 4I Variables for the Control Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		registered process got newly started.			
BBC_INIT_ CHECK_RETRY	`ctrl.ovcd`	The number of attempts that **ovc** makes to start **ovcd**.	YES	3	Integer
WIN_COMPAT_ VARS	`ctrl.ovcd`	Applicable only on Windows. For actions executed by `Ctrl`, slashes within environment variables like `%OvInstallDir%` get reversed (‘\’ gets reversed to ‘/’).	YES	false	Boolean
OV_SUDO	`ctrl.sudo`	The OV_SUDO variable can be specified to define a sudo-like program to run `Ctrl` Service in the non-root environment.	YES		String
OV_SUDO_ GROUP	`ctrl.sudo`	The preferred sudo group can be specified using the variable OV_SUDO_ GROUP <sudo group>	YES		String
OV_SUDO_USER	`ctrl.sudo`	The preferred sudo user can be specified using the variable OV_SUDO_USER <sudo user>	YES		String
PROCESS_ TIMEOUT	`ctrl.ovcd`	Specifies the timeout, in seconds, within which `Ctrl` expects a particular state to be established, during the state transition. If the particular state is not established within the timeout then it is reported as an	YES	120	Integer

Table 4l Variables for the Control Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		abnormal behavior.			
MAX_ PROCESSES_ LIMIT	`ctrl.ovcd`	This variable specifies a maximum limit on the number of processes running on a Windows system that control daemon can monitor. You can set the variable to a value greater than or equal to the number of processes running on the system.	NO	2048	Integer
LOG_PROCESS_ DISABLE	`ctrl.ovcd`	Whenever a process is started or stopped, the control daemon logs the process details such as process name and pid in the `System.txt` file. If LOG_PROCESS_ DISABLE variable is set to 1, the Control daemon does not log process details in the `system.txt` file when each process is started or stopped. If LOG_PROCESS_ DISABLE variable is set to 0 or not set, the logging of process details is enabled.	YES	0	Integer
OVCD_KILL_ TIMEOUT	`ctrl.ovcd`	Defines the maximum time period in seconds before forcefully aborting **ovc** process in case the process does not exit normally.	YES	40	Integer

Configuration Variables for the Deploy Component

You can use the table to change the variable settings of the deploy component:

Table 4l Variables for the Deploy Component

Variable	Namespace	Description	Restart Required	Default Value	Type
CMD_TIMEOUT	`depl`	Deploy uses CMD_TIMEOUT seconds to wait for the completion of deploy commands.	NO	60000 milliseconds	Integer
INSTALLATION_TIME	`depl`	Deploy uses INSTALLATION_TIME seconds to install bundles and packages.	YES	60000 seconds	Integer
MAX_BLOCK_SIZE	`depl`	File Transfer Block Size for transferring files.	YES	4096	long
DEPLOY_MECHANISMS	`depl`	Deploy uses the specified mechanism (like ssh) to deploy packages on remote nodes.	YES		String
COPY	`[depl].mechanism.***`	Deploy uses the specified mechanism specified under DEPLOY_MECHANISMS to deploy packages on remote nodes.	YES		String

Table 4l Variables for the Deploy Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		For a particular deploy mechanism, the **copy** command to be used should be specified here.			
EXEC	`[depl].mechanism.***`	Deploy uses the mechanism specified under DEPLOY_MECHANISMS to deploy packages on remote nodes. For a particular deploy mechanism, the **execute** command to be used should be specified here.	YES		String
BUNDLE_DIR	`depl.bootstrap`	Deploy uses this as the source directory for bundles for bootstrap.	YES		String
BUNDLE_NAME	`depl.bootstrap`	Deploy uses this as the bundle name for bootstrap.	YES		String
BUNDLE_VERSION	`depl.bootstrap`	Deploy uses this as the bundle version for bootstrap.	YES		String
INCLUDEUPDATES	`depl`	Use this variable for remote deployment of base, patch, and	YES	True	Boolean

Table 4l Variables for the Deploy Component, continued

Variable	Namespace	Description	Restart Required	Default Value	Type
		hotfix installation. For more information, see the *Operations Agent and Smart Plug-ins for Infrastructure Installation Guide*. .			
DISABLE_REL_PATH	`depl`	Use this variable to prevent the applications, which consumes ovbbc and ovdeploy APIs, not to use relative paths while using the file transfer mechanism.	YES	False	Boolean

Chapter 5: Agent Application Programming Interface

Your use of the *Agent Application Programming Interface* section is subject to the license terms for the Operations Agent software.

The agent Application Programming Interface (API) enables you to integrate your own applications and programs with OM. Operations Agent includes the following APIs:

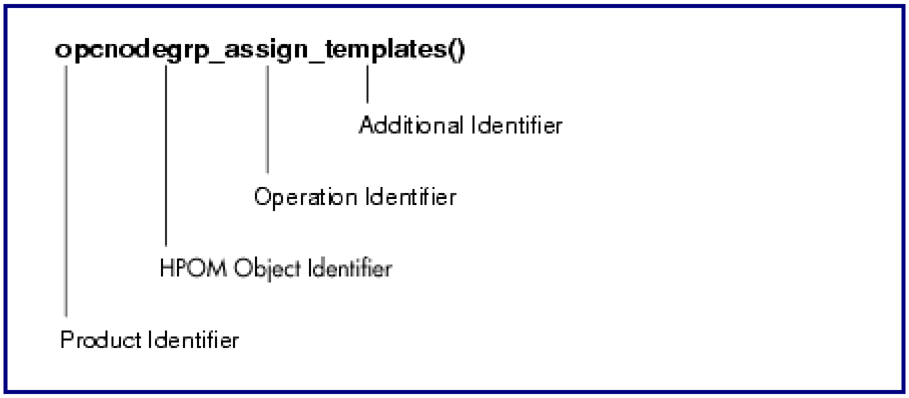
- Agent message API
- Agent monitor API
- Java API

> **Note:** The Operations Agent API includes support for C/C++ and Java, as well as for every language that supports DCOM automation (for example, VB, VBScript, JScript, and so on). However, the agent message stream interface supports C APIs only. All of the APIs are built using Microsoft Visual Studio 2005.

Function-Naming Conventions

The functions of the OM APIs have consistent names which reflect the operation they perform and the OM object on which they perform it. See the figure below for an example of how the OM API functions are named.

Figure 1: Naming the OM API Functions



The function names consist of the following parts:

- product identifier: Identifies the product, in OM this is always opc.
- OM object identifier: Identifies the OM object on which the function performs the operation.
- operation identifier: Identifies the operation which the function performs.
- additional identifier: Additional description to identify what the function does or expects.

The figure above gives an overview of all available identifiers.

> **Note:** Not all operations are available on all OM objects and not every addition is available for each operation.

Table 13 Overview of Function-name Combinations

Product Identifier	OM Object Identifier	Operation Identifier	Additional Identifier	Additional Identifier
opc	appl	_add	_all	_layoutgrps
	applgrp	_modify	_list	_nodes
	data	_delete	_node	
	if	_get	_nodes	
	msg	_assign	_nodegrps	
	msggrp	_deassign	_templates	
	msgregrp	_move	_templgrps	
	node		_nodehier	
	nodegrp		_layoutgrp	
	nodehier		_layoutgrps	
	profile		_appls	
	reg		_applgrps	
	sync		_parentusers	
	templ		_profiles	
	templfile		_resps	
	templgrp		_defaults	
	transact			
	user			

Figure 1 gives an overview of all available OM objects which can be manipulated with the APIs. The opcdata type must be used to describe the objects when using the APIs.

OM Objects

OM Object	Description	opcdata Type
Action Request	Action request to start an action on a managed node. Used by the Legacy Link Interface.	`OPCDTYPE_ACTION_REQUEST`
Action Response	Action response from a previously started action on a managed node. Used by the Legacy Link Interface.	`OPCDTYPE_ACTION_RESPONSE`

Annotation	Message annotation.	`OPCDTYPE_ANNOTATION`
Application	Application used in OM	`OPCDTYPE_APPLIC`
Application Configuration	Configuration of an OM application. This object type is used to configure OM applications.	`OPCDTYPE_APPL_CONFIG`
Application Group	Application group; an application group is a container of applications and other application groups.	`OPCDTYPE_APPL_GROUP`
Application Response	An application response is the response of a previously started OM application. Application responses can be received using the Application Response Interface.	`OPCDTYPE_APPLIC_RESPONSE`
Container	A container contains a list of objects of one type.	`OPCDTYPE_CONTAINER`
Layout Group	A layout group contains a list of layout elements in a node hierarchy.	`OPCDTYPE_LAYOUT_GROUP`
Message	A message is the central management information element of the managed nodes.	`OPCDTYPE_MESSAGE`
Message Event	A message event is sent when a message was changed.	`OPCDTYPE_MESSAGE_EVENT`
Message Group	A message group is a grouping criteria of incoming messages.	`OPCDTYPE_MESSAGE_GROUP`
Message ID	A message ID contains the unique identifier of a message.	`OPCDTYPE_MESSAGE_ID`
Monitor Message	A monitor message is a monitor value which can be sent using the Agent Monitor API.	`OPCDTYPE_MONITOR_MESSAGE`
Node	A node is an OM managed node.	`OPCDTYPE_NODE`
Node Configuration	A node configuration is the configuration of an OM managed node. It contains all necessary parameters to specify a node with all its characteristics.	`OPCDTYPE_NODE_CONFIG`
Node Group	A node group collects several nodes.	`OPCDTYPE_NODE_GROUP`
Node Hierarchy	A node hierarchy is a tree structure containing node layout elements and nodes as its leaves.	`OPCDTYPE_NODEHIER`
Regroup Condition	A regroup condition regroups messages matching the specified condition.	`OPCDTYPE_REGROUP_COND`
Template	A template is used to configure message conditions on managed nodes.	`OPCDTYPE_TEMPLATE_INFO`

Template Group	A template group collects several templates and other template groups. Template groups are handled like templates.	`OPCDTYPE_TEMPLATE_INFO`
Template File	A template file contains the complete configuration of a template including its conditions. Template files are only used by the template file API.	`[char *]`
Template Info	A template info object contains the name, description, and type of a template. It can be used to get a list of all available templates instead of the complete template configuration.	`OPCDTYPE_TEMPLATE_INFO`
User Configuration	A user configuration contains the properties of an OM user.	`OPCDTYPE_USER_CONFIG`
User Profile	A user profile contains the properties of users and is assigned to users so that the user takes over the properties defined in the profile.	`OPCDTYPE_USER_PROFILE`

Libraries on the Managed Nodes

Instrumentation programs which use the Operations Agent APIs must be developed on a system with an Operations Agent installed, so that the OM shared libraries and `opcapi.h` header files are both available.

Platforms that support multi-threaded environments must also supply reentrant system calls that work in this environment. Some platforms only supply reentrant libraries which also work for single-threaded applications. Some have separate libraries-a standard library and a reentrant library; for example, `libc` and `libc_r`, or `libsocket` and `libsocket_r`.

On platforms with two sets of libraries, it is important to link the application using the standard library to the `crt0` object file, and the reentrant library using the `crt0_r` object file. `crt0` and `crt0_r` contain code that is executed before `main()` and is responsible for setting up or initializing the environment before calling any of the library APIs. Mixing reentrant and non-reentrant `crt0` and libraries is not allowed.

Lightweight Libraries for the Operations Agent

HTTPS agent version 8.53 or higher provide lightweight libraries, which use less memory and provide better performance than previous libraries. Link the lightweight libraries if you develop new applications that use Operations Agent APIs.

The lightweight libraries provide the same interfaces as the previous libraries. Therefore, you can recompile existing applications to link the lightweight libraries.

Examples of how to use the lightweight libraries are available in the following folder on nodes that have the agent version 8.53 or higher:

`<OvInstallDir>/examples/copcagtapi`

Table 14 Lightweight Libraries

Operating system	Libraries	
Windows [a]	32 bit	`%OvInstallDir%\bin\libopcagtapi.dll`
	64 bit	`%OvInstallDir%\bin\win64\libopcagtapi.dll`
HP-UX PA-RISC[b]	`/opt/OV/lib/libopcagtapi.sl`	
HP-UX Itanium	`/opt/OV/lib/hpux32/libopcagtapi.so`	
Linux[a.b.]	32 bit	`/opt/OV/lib/libopcagtapi.so`
	64 bit[c]	`/opt/OV/lib64/libopcagtapi.so`
Solaris[b]	32 bit	`/opt/OV/lib/libopcagtapi.so`
	64 bit [d]	`/opt/OV/lib64/libopcagtapi.so`
AIX[b]	32 bit	`/usr/lpp/OV/lib/libopcagtapi.a`
	64 bit [d]	`/usr/lpp/OV/lib64/libopcagtapi.a`

[a] On operating systems for which the agent provides both 64 bit and 32 bit lightweight libraries, link the appropriate library for your program (for example, link the 32 bit libraries to a 32 bit program, even if the program runs on a 64 bit operating system).

[b] To use the lightweight library on UNIX and Linux operating systems, you must also link the following HPE BTO Software shared library:

[c] The 64 bit libraries that are included with 32 bit Linux agents do not support message stream interface functions. To compile a 64 bit application that uses message stream interface functions, link the 64 bit libraries from a 64 bit Linux agent.

[d] Available in HTTPS agent version 8.60 or higher.

Compiler Versions and Options for Agent APIs

To use the Operations Agent APIs, you must use the correct compiler version and options. The following tables list the compiler versions and options for each platform.

Microsoft Windows Server 2003 on x86 (32 bit)

Compiler	Microsoft Visual Studio 2005 Team Edition plus VS2005 Service Pack 1
Required compiler options	• **/GR** enable RTTI • **/MD** Mutithreaded DLL (Use for release version) • **/MDd** Debug Multithreaded DLL (use for debug versions) • **/EHa** enable C++ exception handling • **/W3** Warning level 3 • **/Wp64** Detect 64-bit portability issues • **/GF** Enable string pooling • **/J** Default unsigned char • **/Zc:wchar_t** wchar_t is a native type • **/Gd** Use __cdecl calling convention. • **/analyze** Enterprise code analysis
Other requirements	Embed the manifest for all DLLs, loadable modules, and executables into the binary using mt.exe.

Microsoft Windows Server 2003 on x64 (64 bit)

Compiler	Microsoft Visual Studio 2005 Team Edition plus VS2005 Service Pack 1
Required compiler options	• **/GR** enable RTTI • **/MD** Mutithreaded DLL (Use for release version) • **/MDd** Debug Multithreaded DLL (use for debug versions) • **/EHa** enable C++ exception handling • **/W3** Warning level 3 • **/Wp64** Detect 64-bit portability issues • **/GF** Enable string pooling • **/J** Default unsigned char • **/Zc:wchar_t** wchar_t is a native type • **/Gd** Use __cdecl calling convention.

	- **/analyze** Enterprise code analysis
Other requirements	Embed the manifest for all DLLs, loadable modules, and executables into the binary using mt.exe.

Microsoft Windows Itanium (64 bit)

Compiler	Microsoft Visual Studio 2005 Team Edition. Itanium cross-compiler with VS2005 SP 1
Required compiler options	• **/GR** enable RTTI • **/MD** Mutithreaded DLL (Use for release version) • **/MDd** Debug Multithreaded DLL (use for debug versions) • **/EHa** enable C++ exception handling • **/W3** Warning level 3 • **/Wp64** Detect 64-bit portability issues • **/GF** Enable string pooling • **/J** Default unsigned char • **/Zc:wchar_t** wchar_t is a native type • **/Gd** Use __cdecl calling convention. • **/analyze** Enterprise code analysis
Other requirements	• Build Windows Server 2003 Itanium binaries on an x86 system using a cross compiler. • Embed the manifest for all DLLs, loadable modules, and executables into the binary using mt.exe.

HP-UX 11.11, 11.23 PA (32 bit API)

Compiler	aCC A.03.80
Required compiler options	• **-AP** use older C++ runtime libraries **Note:** This is the default • **-mt** for thread-safe code
Recommended options	• **-Aa** enables newly supported ANSI C++ standard features • **-D__HPACC_STRICTER_ANSI__** enables additional ANSI compliance of STL • **+hpxstd98** enables new, standards compliant compilation mode
Other requirements	Runtime patch PHSS_33945

HP-UX 11.23 IA64 (in native IPF mode) (32 Bit API)

Compiler	HP aC++ Compiler (Version: A.06.05)
Required compiler options	• **-AA** use ANSI-standard STL and IOStreams (this is the default) • **-mt** for thread-safe code • **+DD64** create 64 bit mode binaries (only for HPUX11.23_IPF64 executables)
Recommended options	• **-Aa** enable newly supported ANSI C++ standard features (this is only necessary if –AA is not used explicitly) • **+DSitanium2** optimize code for Itanium 2 CPU (also runs on Itanium 1)
Other requirements	• Build patch PHSS_33350 11.23 aC++ Runtime (IA: A.06.05) • Build patch PHSS_33352 11.23 Integrity Unwind Library

SuSE Linux ES 9, SuSE 9.1, 9.2, 9.3, RedHat Enterprise Linux 4.0 (32 or 64 bit API)

Compiler	gcc version 3.3.3-43 (Standard compiler of SuSE Linux ES 9)
Required compiler options	**-lpthread** Executables must be linked with the pthread library if any directly or indirectly used shared library is dependent on the pthread library, even if the executable itself is a single-threaded application.
Other requirements	To compile 32 bit binaries on a 64 bit system, use the **–m32** compiler switch.

SuSE Linux ES 10, RedHat Enterprise Linux 5.0 (64 bit CPU required) (64 Bit API)

Compiler	gcc version 4.1.0 (Standard compiler of SuSE Linux ES 10)
Required compiler options	**-lpthread** Executables must be linked with the pthread library if any directly or indirectly used shared library is dependent on the pthread library, even if the executable itself is a single-threaded application.
Other requirements	To compile 32 bit binaries on a 64 bit system, use the **–m32** compiler switch.

SuSE Linux ES 10, RedHat Enterprise Linux 5.0 (Itanium) (64 bit API)

Compiler	gcc version 4.1.0 (Standard compiler of SuSE Linux ES 10)
Required compiler options	**-lpthread** Executables must be linked with the pthread library if any directly or indirectly used shared library is dependent on the pthread library, even if the executable itself is a single-threaded application.

SuSE Linux ES 10, RedHat Enterprise Linux 5.0 (x64 or Itanium) (32 Bit API)

Compiler	gcc version 3.3.3-43 (Standard compiler of SuSE Linux ES 9)
Required compiler options	**-lpthread** Executables must be linked with the pthread library if any directly or indirectly used shared library is dependent on the pthread library, even if the executable itself is a single-threaded application.
Other requirements	To compile 32 bit binaries on a 64 bit system, use the –m32 compiler switch.

Solaris 10 (SPARC) (32 and 64 bit API)

Compiler	Sun Studio 11
Required compiler options	**-mt** for thread-safe code
Other requirements	Build patches: • 122149 Update checking binary • 124862 Debuginfo handling • 120760 Compilers Back-End • 121017 C++ • 121019 Fortran 95 • 121021 Fortran 95 Libraries • 121015 C 5.8 compiler • 121023 dbx • 120761 Performance Analyzer • 122135 Sun Performance Library • 122142 Sun Studio IDE Runtime patches: • 117557 OpenMP support libmtsk • 108434 32-bit shared library patch for C++ • 108435 64-bit shared library patch for C++ • 111721 SunOS 5.8 Math Library libm patch • 109147 Linker patch • 111697 SCCS and make • 114802 Assembler • 108652 X11 Xsun For Solaris 8: • 108434-08 SunOS 5.8: 32-Bit Shared library patch for C++ • 108993-25 LDAP2 client, libc, libthread, libnsl libraries patch

	• 109147-15 SunOS 5.8: Linker catch

Solaris 10 (x86/x64 – 32 bit)

Compiler	Sun Workshop Compiler 11
Required compiler options	• **-mt** for thread-safe code • **-fast -xtarget=pentium**
	(option sequence is important) Compiler patches: • 122148 Update checking binary • 124859 Debug info handling • 120759 Compilers Back-End • 121018 C++ • 121020 Fortran 95 • 121022 Fortran Libraries • 121016 C 5.8 compiler • 121616 dbx • 120762 Performance Analyzer • 122136 Sun Performance Library • 122143 Sun Studio IDE Operating system patches: • 118677 SunOS 5.10_x86: SCCS and make utilities • 118345 SunOS 5.10_x86: ld & libc.so.1 • 119961 SunOS 5.10_x86: Assembler • 119964 SunOS 5.10_x86 Shared library patch for C++_x86 • 120754 SunOS 5.10_x86 libmtsk • 121621 MediaLib

AIX 6.1 (64 bit)

Compiler	IBM Visual Age C++ Professional / C for AIX Compiler, Version 9.0.
Required compiler options	• **xlC_r** compile thread-safe code • **-qrtti=all** enable RTTI
Other requirements	• Runtime : AIX 6.1 TL2 • xlC.aix61.rte 10.1.0.2 C F XL C/C++ Runtime for AIX 6.1

	• xlC.rte 10.1.0.2 C F XL C/C++ Runtime

AIX 5.3 (32 bit)

Compiler	IBM Visual Age C++ Professional / C for AIX Compiler, Version 5.0.
Required compiler options	• **xlC_r** compile thread-safe code • **-qrtti=all** enable RTTI

AIX 5.3 (64 bit API)

Compiler	IBM Visual Age C++ Professional / C for AIX Compiler, Version 9.0.
Required compiler options	• **xlC_r** compile thread-safe code • **-qrtti=all** enable RTTI

Java

Compiler	Sun JDK 1.5_14
Other requirements	Required Java runtime for AIX 6.1 is JRE 1.6 minimum

Using APIs in Internationalized Environments

All OM API functions are internationalized. This means that they will initialize the language setting, check the codeset for compatibility, and convert codesets if necessary, provided your API programs support Native Language Support (NLS) environments.

When writing API programs for internationalized environments, you must ensure that your programs do select the appropriate locale. In C programs, you do this by calling the function setlocale() at the beginning of your program.

It is recommended to use `setlocale(LC_ALL,"")`. The category `LC_ALL` names the program's entire locale. **""** adopts the setting of the current shell.

Agent Message API

OM provides a set of APIs to handle messages on managed nodes. These functions enable you, for example, to send messages and acknowledge them at a later time. See Agent Monitor API for functions to send monitor values.

Data Structures

- OPCDTYPE_MESSAGE_ID
- OPCDTYPE_MESSAGE

Usage

The managed node processes must be running. To use the functions, include the header file `opcapi.h` in your application.

Prerequisites

Each opdata structure must be allocated using opcdata_create() before it can be used in any of these functions. After the execution of your program, each opcdata structure must be freed using opcdata_free().

Multithread Usage

All function of the Agent Message API are safe to be called by multithreaded applications, and are thread-safe for both POSIX Thread and DCE User Threads. They are neither async-cancel, async-signal, nor fork-safe, and cannot be safely called in kernel threads.

Agent Configuration

Operations on messages out of managed nodes require to send these message operations to the manager. Unfortunately it is not possible to deliver the responsible manager of a message from the message ID. Additionally, the configuration could be changed since the message was sent so that it is necessary to send the message operation to all managers. This can produce a lot of network load.

To prevent this, the message agent holds information about the manager to which the messages were sent. After a defined time, the information is deleted to save memory, disk space, and processing time. This time is configurable with a nodeinfo policy using the parameter OPC_STORE_TIME_FOR_MGR_INFO. The specified value is the time in hours, with a default setting of one hour if this parameter is not changed.

The storage of the manager information must be enabled for each message to be sent by setting the message parameter OPCDATA_DATA_INFO to OPC_REMARK_FOR_ACK.

opcdata_set_long(message, OPCDATA_DATA_INFO, OPC_REMARK_FOR_ACK);

opcmsg()

opcagtmsg_send()

opcagtmsg_ack()

opcagtmsg_ack()

```
#include opcapi.h

int opcagtmsg_ack (

opcdata     message_id     /* in */

);
```

Parameters

```
message_id

        Message ID of type OPCDTYPE_MESSAGE_ID.
```

Description

Use the function `opcagtmsg_ack()` to acknowledge a message out from a managed node. A message operation will be sent to the message agent.

If the message attribute `OPCDATA_DATA_INFO` of a previously sent message was set to `OPC_REMARK_FOR_ACK`, the message agent holds the information about the responsible manager in its memory. If this attribute was not set, the message operation will be sent to all managers.

Return Values

```
OPC_ERR_OK:

        OK

OPC_ERR_INVALID_INPARAM:

        message_id is NULL

OPC_ERR_INVALID_OPCDATA_TYPE:

        message_id is not of type OPCDTYPE_MESSAGE_ID

OPC_ERR_INCOMPLETE_PARAM:

        message ID is not set

OPC_ERR_NO_MEMORY:

        memory allocation failed
```

opcagtmsg_send()

```
#include opcapi.h
```

```
int opcagtmsg_send (
opcdata     message     /* in/out */
);
```

Parameters

`message`

`        Message of type OPCDTYPE_MESSAGE.`

Description

Use the function `opcagtmsg_send()` to send a message, created on the managed node, to its responsible manager. The message must be of type `OPCDTYPE_MESSAGE`. The message ID can be retrieved from the message object using `opcdata_get_str()` immediately after the send call was executed.

Only the message attributes Severity, Application, Message Group, Object, Message Text, Option Strings and Node are used in `opcagtmsg_send()`.

If you want to save the information about the responsible manager, remark the message to be acknowledged later. To do this, set `OPCDATA_DATA_INFO` to `OPC_REMARK_FOR_ACK`.

After `opcagtmsg_send()` was called with `OPC_REMARK_FOR_ACK` it is possible to get the ID of the sent message using:

opcdata_get_str() (`message, OPCDATA_MSGID`)

Return Values

```
OPC_ERR_OK:
        OK
OPC_ERR_APPL_REQUIRED:
        attribute OPCDATA_APPLICATION not set
OPC_ERR_OBJ_REQUIRED:
        attribute OPCDATA_OBJECT not set
OPC_ERR_TEXT_REQUIRED:
        attribute OPCDATA_MSGTEXT not set
OPC_ERR_INVAL_SEVERITY:
        set severity invalid
OPC_ERR_MISC_NOT_ALLOWED:
        message group 'misc' not allowed
```

```
OPC_ERR_INVALID_INPARAM:

        message is NULL

        message is not of type OPCDTYPE_MESSAGE

OPC_ERR_WRONG_OPTION_VARS:

        The field OPCDATA_OPTION_VAR of the message has an incorrect format. It can only
contain assignments separated by spaces.

OPC_ERR_NO_MEMORY:

        memory allocation failed
```

opcmsg()

```
#include opcapi.h

int opcmsg (

const int      severity,      /* in */

const char *   application,   /* in */

const char *   object,        /* in */

const char *   msg_text,      /* in */

const char *   msg_group,     /* in */

const char *   nodename,      /* in */

);
```

Parameters

`severity`

Severity level of the new message.

The following severities are supported:

OPC_SEV_NORMAL

OPC_SEV_WARNING

OPC_SEV_MINOR

OPC_SEV_MAJOR

OPC_SEV_CRITICAL.

`application`

Application of the message source.

`object`

Object of the message source.

`msg_text`

Message text

`msg_group`

Message group

`nodename`

Name of the node originating the message.

Description

Use the function `opcmsg()` to send a message, created on the managed node, to the management server. This function does not return the message ID so that it is not possible to acknowledge the message later, on the managed node.

Return Values

`OPC_ERR_OK:`

OK

`OPC_ERR_APPL_REQUIRED`:

The application parameter is not set.

`OPC_ERR_OBJ_REQUIRED:`

The object parameter is not set.

`OPC_ERR_TEXT_REQUIRED:`

The msg_text parameter is not set.

`OPC_ERR_INVAL_SEVERITY:`

The severity parameter value is invalid

`OPC_ERR_MISC_NOT_ALLOWED:`

message group 'misc' is not allowed

`OPC_ERR_NO_MEMORY:`

out of memory

Agent Monitor API

OM provides a set of functions to send monitor values to the monitor agent.

Data Structures

OPCDTYPE_MONITOR_MESSAGE

Usage

To use these functions, the managed node processes must be running. To use the functions, include the header file opcapi.h in your application.

Prerequisites

Each opdata structure must be allocated using opcdata_create() before it can be used in any of these functions.

Multithread Usage

All functions of the Agent Monitor API are safe to be called by multithreaded applications, and are thread-safe for both POSIX Threads and DCE User Threads. They are neither async-cancel, async-signal, nor fork-safe, and cannot be safely called in kernel threads.

opcmon()

opcagtmon_send()

opcagtmon_send()

```
#include opcapi.h

int opcagtmon_send (

opcdata     mon_msg     /* in */

);
```

Parameters

`mon_msg`

Monitor message/value of type: OPCDTYPE_MONITOR_MESSAGE.

Description

Use the function `opcagtmon_send()` to send a monitor value, created on the managed node, to the monitor agent. The mon_msg must be of type OPCDTYPE_MONITOR_MESSAGE.

Only the message attributes Monitor Name, Monitor Value, Object and Option String are used in `opcagtmon_send()`.

Return Values

`OPC_ERR_OK:`

OK

`OPC_ERR_INVALID_INPARAM:`

`mon_msg` is NULL

`mon_msg` is not of type OPCDTYPE_MONITOR_MESSAGE

`OPC_ERR_OBJNAME_REQUIRED:`

attribute `OPCDATA_MON_VAR` not set

`OPC_ERR_NO_AGENT:`

agent is not running

`OPC_ERR_NO_MEMORY:`

out of memory

`OPC_ERR_WRONG_OPTION_VARS:`

attribute `OPCDATA_OPTION_VAR` not set correctly

opcmon()

```
#include opcapi.h

int opcmon (

const char     *objname,     /* in */

const double   monval        /* in */

);
```

Parameters

`objname`

Name of the monitored object.

`monval`

Actual value of the monitored object.

Description

Use the function `opcmon()`to send a monitor value, created on the managed node, to its responsible management server.

Return Values

`OPC_ERR_OK`:

OK

`OPC_ERR_OBJNAME_REQUIRED`:

`objname` is NULL

`OPC_ERR_NO_AGENT:`

agent is not running

`OPC_ERR_NO_MEMORY:`

out of memory

Agent Message Stream Interface (MSI)

The Agent Message Stream Interface allows you to tap the message flow of an OM managed node to enable additional message processing by external applications before a message is sent to the management server. This can help to reduce the amount of network traffic considerably. A typical external application might be an event correlation engine, for example ECS.

> **Note:** The Operations Agent API includes support for C/C++ and Java, as well as for every language that supports DCOM automation (for example, VB, VBScript, JScript, and so on). However, the agent message stream interface supports C APIs only. All of the APIs are built using Microsoft Visual Studio 2005.

Enable the Agent Message Stream Interface

The Agent Message Stream interface is disabled per default on the managed nodes. To allow external programs to use the MSI on the agent, you must first enable it. To enable it, create a nodeinfo policy containing `OPC_AGTMSI_ENABLE TRUE` on the management server, and then deploy it to the managed nodes on which the MSI should be enabled.

Per default, it is also not allowed to write messages containing automatic or operator initiated command to the MSI. The message agent discards the actions in the messages.

To allow the definition of automatic actions add the following to the nodeinfo policy:

```
OPC_AGTMSI_ALLOW_AA TRUE
```

To allow the definition of operator initiated actions add the following to the nodeinfo policy:

```
OPC_AGTMSI_ALLOW_OA TRUE
```

Configure messages to be sent to the Agent Message Stream Interface

Even if the Agent MSI is enabled and an application is registered for messages, you need to specify that a message should be sent to the agent MSI. You can do so in the policy editors on the Message stream interface and external services tab of the window Outgoing Message.

To define that a message should be sent to the agent MSI, select Agent Message Stream Interface and choose whether messages are copied or diverted.

msiconf()

Name

`msiconf` is the configuration file for the OM for Windows message manager

Synopsis

Server MSI

```
<SERVER_CONFIG_DIR>/msiconf
```

For example:

```
/etc/opt/OV/share/conf/OpC/mgmt_sv/msiconf
```

Agent MSI

```
<AGENT_CONFIG_DIR>/msiconf
```

For example:

On HP-UX:

```
/var/opt/OV/conf/OpC/msiconf
```

Description

The file `msiconf` is an ASCII file containing a list of entries consisting of an OM Message Stream Interface (MSI) instance name followed by an order number. Each field is separated by a space, several spaces, or a tab. Each entry is separated from the next by a new line.

The MSI instance name may be a string up to 13 alphanumeric characters. The order number may be an integer value between -127 and 127. Lines or portions of lines beginning with # are assumed to be comments and are ignored. Blank lines are also ignored.

The MSI instance name corresponds to the name of a server MSI application that registers with the OM message manager. The order number specifies the order in which the registered MSI application will receive a message from the message manager (lowest to highest). Registered MSI applications that are not listed in the `msiconf` file are given an order number of 0.

The `msiconf` file is read by the message manager or message agent whenever an MSI instance opens or closes a connection to the MSI.

Example

counter -10

opcecm 0

proca 10

proca 10

enhtt 20

It is possible for a registered MSI instance to alter or completely suppress a message before writing back to the message stream. The proca and procb entries in the above example demonstrate a parallel MSI configuration, where one message entering the message stream may result in two messages exiting the message stream.

Java API

OM provides a set of Java classes on the Operations Agent to:

- create and send a message to the OM management server
- acknowledge a previously sent message
- send a monitor value to the OM monitor agent

> **Note:** The Operations Agent API includes support for C/C++ and Java, as well as for every language that supports DCOM automation (for example, VB, VBScript, JScript, and so on). However, the agent message stream interface supports C APIs only. All of the APIs are built using Microsoft Visual Studio 2005.

JAR files

The JAR files `jopcagtbase.jar` and `jopcagtmsg.jar` that are necessary to use the APIs are installed together with the agent on the managed node.

On Windows:

To use the Java OM classes:

- the `-classpath` parameter used for the `javac` and `java` commands must include the `jopcagtbase.jar` and `jopcagtmsg.jar` files
- the `PATH` system variable must include the directory where the shared library files reside. The agent installation does this automatically.

See `%OvInstallDir%www/htdocs/jdoc_agent/index.html` for a javadoc style class documentation.

To compile and run the example code:

1. Go to the `%OvInstallDir%\examples\OVOW\DevelopmentKit\Agent\Java` directory
2. Compile the example code with `javac -classpath "%OvInstallDir%/java/jopcagtbase.jar:%OvInstallDir%/java/jopcagtmsg.jar"` <*java source code file*>
3. Run the example code with `java -classpath ".:%OvAgentDir%/java/jopcagtbase.jar:%OvAgentDir%/java/jopcagtmsg.jar"` <*java class*>

where <*java source code file*> could be `JOpcAgtMsgTest.java` or `JOpcMonValueTest.java`; <*java class*> would then be `JOpcAgtMsgTest` or `JOpcMonValueTest`

On UNIX:

To build the managed node sample program you have to copy the source files to the managed node. The Operations Agent software must be installed on the managed node; otherwise, the OM JAR files will not be present. Copy the sample programs to any location (for example, /tmp).

To use the Java OM API wrapper classes:

- the `-classpath` parameter used for the `javac` and `java` commands must include the `jopcagtbase.jar` and `jopcagtmsg.jar` files
- the `PATH` system variable must include the directory where the shared library files reside. The agent installation does this automatically.

See `/opt/OV/www/htdocs/jdoc_agent/index.html` for a javadoc style class documentation.

To compile and run the example code:

1. Copy the source code to the managed node into a temporary directory and cd to the directory.
2. Compile the example code with `javac -classpath "/opt/jar/jopcagtbase.jar:/opt/jar/jopcagtmsg.jar"`<*java source code file*> .
3. Run the example code with `java -classpath ".:/opt/jar/jopcagtbase.jar:/opt/jar/jopcagtmsg.jar"` <*java class*>
   where <*java source code file*> could be `JOpcAgtMsgTest.java` or `JOpcMonValueTest.java`; <*java class*> would then be `JOpcAgtMsgTest` or `JOpcMonValueTest`

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Reference Guide (Operations Agent 12.05)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to docfeedback@hpe.com.

We appreciate your feedback!