IBM® Tivoli® Netcool/OMNIbus Probe for Alcatel-Lucent 5620 SAM 3GPP v8 3.0

Reference Guide
December 2, 2011

SC14-7647-02

Note

Before using this information and the product it supports, read the information in Appendix A, “Notices and Trademarks,” on page 25.

Edition notice

This edition (SC14-7647-02) applies to version 3.0 of IBM Tivoli Netcool/OMNIbus Probe for Alcatel-Lucent 5620 SAM 3GPP v8 and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces (SC14-7647-01).

Contents

About this guide

The following sections contain important information about using this guide.

Document Control Page

Use this information to track changes between versions of this guide.

The IBM Tivoli Netcool/OMNIbus Probe for Alcatel-Lucent 5620 SAM 3GPP v8 documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM Tivoli Netcool Knowledge Center:

http://www-01.ibm.com/support/knowledgecenter/?lang=en#!/SSSHTQ/omnibus/probes/common/Probes.html

Table 1. Document modification history

Document version	Publication date	Comments
SC14-7647-00	February 25, 2011	First IBM publication.
SC14-7647-01	July 1, 2011	"Summary" on page 1 updated : • Visibroker information removed. • IBM Object Request Broker (IBM ORB) added as a requirement. "Known Issues with the Probe for Alcatel-Lucent 5620 SAM 3GPP v8 " on page 17 added.
SC14-7647-02	December 2, 2011	"Summary" on page 1 updated. "Ports required for firewalls" on page 3 added.

Conventions used in this guide

All probe guides use standard conventions for operating system-dependent environment variables and directory paths.

Operating system-dependent variables and paths

All probe guides use standard conventions for specifying environment variables and describing directory paths, depending on what operating systems the probe is supported on.

For probes supported on UNIX and Linux operating systems, probe guides use the standard UNIX conventions such as **$***variable* for environment variables and forward slashes (**/**) in directory paths. For example:

```
$OMNIHOME/probes
```

For probes supported only on Windows operating systems, probe guides use the standard Windows conventions such as **%***variable***%** for environment variables and backward slashes (****) in directory paths. For example:

```
%OMNIHOME%\probes
```

For probes supported on UNIX, Linux, and Windows operating systems, probe guides use the standard UNIX conventions for specifying environment variables and describing directory paths. When using the

Windows command line with these probes, replace the UNIX conventions used in the guide with Windows conventions. If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Note : The names of environment variables are not always the same in Windows and UNIX environments. For example, %TEMP% in Windows environments is equivalent to $TMPDIR in UNIX and Linux environments. Where such variables are described in the guide, both the UNIX and Windows conventions will be used.

Operating system-specific directory names

Where Tivoli Netcool/OMNIbus files are identified as located within an *arch* directory under NCHOME or OMNIHOME, *arch* is a variable that represents your operating system directory. For example:

$OMNIHOME/probes/*arch*

The following table lists the directory names used for each operating system.

Note : This probe may not support all of the operating systems specified in the table.

Table 2. Directory names for the arch variable

Operating system	**Directory name represented by *arch***
AIX® systems	aix5
Red Hat Linux® and SUSE systems	linux2x86
Linux for System z	linux2s390
Solaris systems	solaris2
Windows systems	win32

OMNIHOME location

Probes and older versions of Tivoli Netcool/OMNIbus use the OMNIHOME environment variable in many configuration files. Set the value of OMNIHOME as follows:

- On UNIX and Linux, set $OMNIHOME to $NCHOME/omnibus.
- On Windows, set %OMNIHOME% to %NCHOME%\omnibus.

Chapter 1. Probe for Alcatel-Lucent 5620 SAM 3GPP v8

The Alcatel-Lucent 5620 Service Aware Manager (SAM) is a network management system that is used to manage network nodes. Distributed Alcatel-Lucent 5620 SAM elements are monitored by the Alcatel-Lucent 5620 SAM database using GUI and network management components through a Java-based application.

The Probe for Alcatel-Lucent 5620 SAM 3GPP v8 collects alarms from Alcatel-Lucent 5620 SAM using a 3rd Generation Partnership Project (3GPP) Common Object Request Broker Architecture (CORBA) interface.

This guide contains the following sections:

Summary

Each probe works in a different way to acquire event data from its source, and therefore has specific features, default values, and changeable properties. Use this summary information to learn about this probe.

The following table provides a summary of the Probe for Alcatel-Lucent 5620 SAM 3GPP v8.

Table 3. Summary	
Probe target	Alcatel-Lucent 5620 Service Aware Manager (SAM) version R8.0
Probe executable name	`nco_p_alcatel_5620_sam_3gpp_v8`
Package version	3.0
Probe supported on	For details of supported operating systems, see the following Release Notice on the IBM® Software Support Website: https://www-304.ibm.com/support/docview.wss?uid=swg21462452
Properties file	`$OMNIHOME/probes/`*`arch`*`/alcatel_5620_sam_3gpp_v8.props`
Rules file	`$OMNIHOME/probes/`*`arch`*`/alcatel_5620_sam_3gpp_v8.rules`

Table 3. Summary (continued)	
Requirements	A currently supported version of IBM Tivoli Netcool/OMNIbus. IBM Object Request Broker (IBM ORB, supplied with Netcool/OMNIbus) `probe-sdk-java-2` `probe-nonnative-base-13` (or later) probe-command-port-3 (or later) **Note :** If you are running Netcool/OMINbus 7.3.0, these dependencies are supplied with the probe. Otherwise, you must download these dependencies separately.
Connection method	CORBA
Multicultural support	Not Available
Peer-to-peer failover functionality	Available
IP environment	IPv4 and IPv6 **Note :** The probe is supported on IPv6 when running on IBM Tivoli Netcool/OMNIbus V7.3.0, 7.3.1 and 7.4.0 on all UNIX and Linux operating systems.
Federal Information Processing Standards (FIPS)	IBM Tivoli Netcool/OMNIbus uses the FIPS 140-2 approved cryptographic provider: IBM Crypto for C (ICC) certificate 384 for cryptography. This certificate is listed on the NIST website at http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2004.htm. For details about configuring Netcool/OMNIbus for FIPS 140-2 mode, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide*.

Installing probes

All probes are installed in a similar way. The process involves downloading the appropriate installation package for your operating system, installing the appropriate files for the version of Netcool/OMNIbus that you are running, and configuring the probe to suit your environment.

The installation process consists of the following steps:

1. Downloading the installation package for the probe from the Passport Advantage Online website.

   Each probe has a single installation package for each operating system supported. For details about how to locate and download the installation package for your operating system, visit the following page on the IBM Tivoli Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/probes/all_probes/wip/reference/install_download_intro.html
2. Installing the probe using the installation package.

   The installation package contains the appropriate files for all supported versions of Netcool/OMNIbus. For details about how to install the probe to run with your version of Netcool/OMNIbus, visit the following page on the IBM Tivoli Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/probes/all_probes/wip/reference/install_install_intro.html
3. Configuring the probe.

This guide contains details of the essential configuration required to run this probe. It combines topics that are common to all probes and topics that are peculiar to this probe. For details about additional configuration that is common to all probes, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Firewall considerations

When using CORBA probes in conjunction with a firewall, the firewall must be configured so that the probe can connect to the target system.

Most CORBA probes can act as both a server (listening for connections from the target system) and a client (connecting to the port on the target system to which the system writes events). If you are using the probe in conjunction with a firewall, you must add the appropriate firewall rules to enable this dual behavior.

There are three possible firewall protection scenarios, for which you must determine port numbers before adding firewall rules:

1. If the host on which the probe is running is behind a firewall, you must determine what remote host and port number the probe will connect to.
2. If the host on which the target system is running is behind a firewall, you must determine the incoming port on which the probe will listen and to which the target system will connect.
3. If each host is secured with its own firewall, you must determine the following four ports:
   a. The outgoing port (or port range) for the probe.
   b. The hostname and port of the target system.
   c. The outgoing port on which the target system sends events if the probe is running as a client.
   d. The incoming port on which the probe listens for incoming events.

Note : Most, but not all, CORBA probes listen on the port specified by the **ORBLocalPort** property. The default value for this property is 0, which means that an available port is selected at random. If the probe is behind a firewall, the value of the **ORBLocalPort** property must be specified as a fixed port number.

CORBA probes that use EventManager or NotificationManager objects may use different hosts and ports from those that use NamingService and EntryPoint objects. If the probe is configured to get object references from a NamingService or EntryPoint object, you must obtain the host and port information from the system administrator of the target system. When you have this information, you can add the appropriate firewall rules.

Ports required for firewalls

You need to consider all ports described in this topic. Those that you need to use must be enabled on the firewall.

The following table describes the ports that must be opened on the 5620 SAM server to allow application components to communicate effectively when using firewalls.

Table 4. Ports that require enabling on the firewall

Default port	Type	Description
161	UDP	Port used by the SAM server to send SNMP messages to managed routers.
162	UDP	Port used by the SAM server to receive SNMP traps.
1094, 1095, 4448, 4449	TCP	Ports used by commands on the SAM auxiliary workstation.
1096, 1097, 4446, 4447	TCP	Ports used to register for JMS notifications.

Table 4. Ports that require enabling on the firewall (continued)		
Default port	**Type**	**Description**
1098	TCP	Port used to enable communication between SAM server and SAM GUI clients.
1099	TCP	Port used to enable SAM GUI clients to initialize with the SAM server.
1100	TCP	Port used to enable SAM GUI clients to initialize with the SAM server when there are redundant servers.
4444, 4450	TCP	Ports used by SAM GUI clients to send and received information to and from the SAM server.
8080	TCP	Port used to provide an HTTP interface for SAM-O clients to access the SAM server.
8085	TCP	Port used by the SAM client to verify the existence of the server

Data acquisition

The Probe for Alcatel-Lucent 5620 SAM 3GPP v8 gathers events from Alcatel-Lucent 5620 Service Aware Manager (SAM) using a Common Object Request Broker Architecture (CORBA) interface. CORBA is an Object Management Group specification that provides a standard interface definition between objects in a distributed environment; that is, it allows applications to communicate with one another regardless of where they are located or who has designed them.

Data acquisition is described in the following topics:

CORBA interface

The probe connects to the CORBA interface using an Interoperable Object Reference (IOR) file.

Connecting to the EPIRP object using an IOR file

The probe uses the IOR file of the Entry Point IRP (EPIRP) object to access the alarms and notifications generated by the Alcatel-Lucent 5620 SAM server.

The probe uses the IOR file of the Entry Point IRP object specified by the **EPIRPIorFile** property to get a reference on the EPIRP object. From this object, and using the values set in the **AlarmIrpName** and **NotificationIrpName** properties, the probe retreives the AlarmIRP and NotificationIRP objects. The AlarmIRP object provides details of all existing alarms for resynchronization; the NotificationIRP object provides details of all new alarms as they are created.

Using FTP to access the EPIRP object IOR file

You can access the Entry Point IRP (EPIRP) object IOR file using FTP.

Use the **EPIRPFtpCommand** property to specify the FTP command that the probe should use. Enter the FTP command in the following format:

`ftp://`*`user`*`:<password>@`*`host`*`/`*`path`*

Where:

`user` is a user name with FTP access to the host.

`host` is the machine on which the Alcatel-Lucent 5620 SAM server is running.

`path` is the path to the EPIRP object IOR file.

Note : If you want to specify the password in encrypted format, keep <password> in the value set for the **EPIRPFtpCommand** property and specify the password (encrypted using the nco_aes_crypt utility) using the **FtpPassword** property. If you do not want to specify the password in encrypted format, replace <password> with the password. For details about the nco_aes_crypt utility, see the IBM Tivoli Netcool/OMNIbus Administration Guide (SC23-6371).

Resynchronizing with the AlarmIRP object

When the probe starts, it requests a list of all active alarms on the AlarmIRP object, then sends a resynchronization request. You can also send a resynchronization request when the probe is running by using the command line interface.

To specify which base object the probe resynchronizes with, use the **ResyncBaseObjectFilter** property.

If you want the probe to receive all alarms on the network, leave this property blank.

If you want the probe to receive only those alarms on a given subnetwork, set this property to `SubNetwork=`*`X`*, where *X* identifies the subnetwork.

If you want the probe to receive only those alarms generated by a specific element on a given subnetwork, set this property to `SubNetwork=`*`X`*`,ManagedElement=`*`Y`* where *X* identifies the subnetwork and *Y* identifies the managed element on that subnetwork.

To reduce the number of events that the probe receives based on their probable cause, perceived severity, alarm type, notification type, or managed object instance, use the **ResyncFilter** property. For details, see "Filtering alarms" on page 6.

Subscribing to the NotificationIRP object

After resynchronizing with the AlarmIRP object on startup to receive all existing events, the probe subscribes to the NotificationIRP object to receive new objects as they are created.

To specify which categories of events the probe subscribes to, use the **SubscribeNotifCategories** properties. You can specify one or more of the following categories:

- 32.111-3 V8.0: The probe receives alarms.
- 32.663 V8.1: The probe receives kernel events.
- 32.353 V8.0: The probe receives communications surveillance events.

Note : If you want to specify more than one category, separate each category with a semi-colon.

To reduce the number of events that the probe receives based on their probable cause, perceived severity, alarm type, notification type, or managed object instance, use the **SubscribeFilter** property. For details, see "Filtering alarms" on page 6.

Whenever a new event is ready for forwarding, the NotificationIRP object notifies the probe by calling the reference specified by the **ManagerIdentifier** property.

Once subscribed, the probe performs a getSubscriptionStatus operation with the frequency specified by the **TimeTick** property. If the probe fails to perform this operation within this interval, the NotificationIRP object removes the subscription.

Unsubscribing from the NotificationIRP object

When the probe subscribes to the NotificationIRP object, the Alcatel-Lucent 5620 SAM server returns an ID for the subscription. When the probe shuts down, it automatically unsubscribes from the NotificationIRP object.

If the probe quits unexpectedly, it may not unsubscribe properly. In this case, you can manually unsubscribe by using the command line interface; for details, see "Sending commands to the CORBA interface using a command line interface" on page 8.

Filtering alarms

Filters can be used to reduce the number of alarms received by the probe both during resynchronization and when the probe receives new events as they are generated.

The following table displays the attributes that you can use as a filter. For each attribute, the table shows the string to use in the filter property that corresponds to the attribute, which properties support the attribute, and the values to which you can set the attribute.

Table 5. Filters that can be used to reduce the alarms received by the probe

Attribute name	String to specify in the filter property that corresponds to the attribute	Filter property in which attribute supported	Supported values and their descriptions
ackState	$.filterable_data(n)	**ResyncFilter**	1 - Acknowledged 2 - Unacknowledged
perceivedSeverity	$.filterable_data(h)	**ResyncFilter** and **SubscribeFilter**	1 - Indeterminate 2 - Critical 3 - Major 4 - Minor 5 - Warning 6 - Cleared
probableCause	$.filterable_data(g)	**ResyncFilter** and **SubscribeFilter**	A valid value for this attribute is an integer that corresponds to its probable cause. For a full list of the probable causes that the probe supports, refer to the rules file.

Table 5. Filters that can be used to reduce the alarms received by the probe (continued)

Attribute name	String to specify in the filter property that corresponds to the attribute	Filter property in which attribute supported	Supported values and their descriptions
alarmType	$.header.fixed_head.event_name	**ResyncFilter** and **SubscribeFilter**	x1 - Communications alarms x2 - Processing errors alarms x3 - Environmental alarms x4 - Quality of service alarms x5 - Equipment alarms x6 - Integrity violations x7 - Operational violations x8 - Physical violations x9 - Security service or mechanism violations x10 - Time domain violations
Notification_type	$.header.fixed_head.type_name	**SubscribeFilter**	x1 - New alarms x2 - Changed alarms x3 - Alarms with acknowledged state changed x5 - Cleared alarms
Object Instance	$.filterable_data(e)	**SubscribeFilter**	A valid value for this attribute is a string representing the name of the object.

Specifying a filter

You can specify a filter using the **ResyncFilter** and **SubscribeFilter** properties. Enter the filter in the following format:

attribute_name==value

You can specify that the probe uses more than one filtering criterion by separating each by AND or OR. For example, if you want the probe to receive all critical alarms that are unacknowledged, set the **ResyncFilter** property to the following value:

```
$.filterable_data(h)==2 AND $.filterable_data(n)==2
```

Sending commands to the CORBA interface using a command line interface

The Probe for Alcatel-Lucent 5620 SAM 3GPP v8 is supplied with a command line interface (CLI). This interface allows you to perform commands using the probe (for example, to acknowledge alarms or to request a resynchronization).

To use the CLI, you must use the **CommandPort** property in the properties file to specify a port through which commands will be sent. When you want to execute commands, Telnet to this port.

Important : If a firewall separates the probe and the Alcatel-Lucent 5620 SAM device, keep the following ports open: TCP 21115, TCP 21126, and TCP 21111.

The following table describes the commands that you can use with the CLI.

Table 6. CLI commands

Command	Description
exit/quit	Use this command to close the Telnet connection to the probe.
acknowledgeAlarms *alarm_id_1 alarm_id_2 ... alarm_id_n*	Use this command to acknowledge one or more alarms in the CORBA interface by specifying the ID of the alarms being acknowledged. **Note :** You must specify a user ID using the **UserId** property and the system ID using the **SystemId** property before you can perform this command using the CLI.
cleanupSubscription *subscription_id*	Use this command to detach the specified subscription from the SAM server. **Note :** You only need to use this command if the probe has shut down unexpectedly. During normal usage the probe unsubscribes as it shuts down.
clearAlarms *alarm_id_1 alarm_id_2 ... alarm_id_n*	Use this command to clear one or more alarms in the CORBA interface by specifying the ID of the alarms being cleared. **Note :** You must specify a user ID using the **UserId** property and the system ID using the **SystemId** property before you can perform this command using the CLI.
commentAlarms *comment_text alarm_id_1 alarm_id_2 ... alarm_id_n*	Use this command to add comments to one or more alarms in the CORBA interface by specifying the text to be added, followed by the ID of the alarms being commented. If the comment that you want to add to the alarm contains whitespace, you must put the comment portion of the command inside double quotation marks. **Note :** You must specify a user ID using the **UserId** property and the system ID using the **SystemId** property before you can perform this command using the CLI.
getEventQueueSize	Use this command to display the current size of the internal event queue.

Table 6. CLI commands (continued)	
Command	**Description**
getSubscriptionStatus *subscription_id*	Use this command to display the status of the specified subscription.
help	Use this command to display online help about the CLI.
resync	Use this command to perform a resynchronization with the CORBA interface. This command reuses the filter specified by the **ResyncFilter** property.
shutdown	Use this command to shut down the probe.

Note : As the CLI is based on Telnet connections, you can connect to the probe from anywhere by creating desktop tools to open a Telnet connection, send a command, and then to close the connection. This means that simple scripts can be set up to allow users to acknowledge selected events from the IBM Tivoli Netcool/OMNIbus event list.

Peer-to-peer failover functionality

The probe supports failover configurations where two probes run simultaneously. One probe acts as the `master` probe, sending events to the ObjectServer; the other acts as the `slave` probe on standby. If the master probe fails, the slave probe activates.

While the slave probe receives heartbeats from the master probe, it does not forward events to the ObjectServer. If the master probe shuts down, the slave probe stops receiving heartbeats from the master and any events it receives thereafter are forwarded to the ObjectServer on behalf of the master probe. When the master probe is running again, the slave probe continues to receive events, but no longer sends them to the ObjectServer.

Example property file settings for peer-to-peer failover

You set the peer-to-peer failover mode in the properties files of the master and slave probes. The settings differ for a master probe and slave probe.

Note : In the examples, make sure to use the full path for the property value. In other words replace `$OMNIHOME` with the full path. For example: `/opt/IBM/tivoli/netcool`.

The following example shows the peer-to-peer settings from the properties file of a master probe:

```
Server      :    "NCOMS"
RulesFile   :    "master_rules_file"
MessageLog  :    "master_log_file"
PeerHost    :    "slave_hostname"
PeerPort    :    6789 # [communication port between master and slave probe]
Mode        :    "master"
PidFile     : "master_pid_file"
```

The following example shows the peer-to-peer settings from the properties file of the corresponding slave probe:

```
Server      :    "NCOMS"
RulesFile   :    "slave_rules_file"
MessageLog  :    "slave_log_file"
PeerHost    :    "master_hostname"
PeerPort    :    6789 # [communication port between master and slave probe]
Mode        :    "slave"
PidFile     : "slave_pid_file"
```

Properties and command line options

You use properties to specify how the probe interacts with the device. You can override the default values by using the properties file or the command line options.

The following table describes the properties and command line options specific to this probe. For more information about generic Netcool/OMNIbus properties and command line options, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 7. Properties and command line options

Property name	Command line option	Description
Agentheartbeat *integer*	`-agentheartbeat` *integer*	Use this property to specify the frequency (in seconds) with which the probe checks the status of the SAM server. The default is `60`.
AlarmIrpName *string*	`-alarmirpname` *string*	Use this property to specify the name of the Alarm IRP object to which the probe sends a resynchronization request. The default is `AlarmIRP=1`.
CommandPort *integer*	`-commandport` *integer*	Use this property to specify the port to which users can use Telnet to communicate with the CORBA interface using the Command Line Interface (CLI) supplied with the probe. For details about the CLI, see "Sending commands to the CORBA interface using a command line interface" on page 8. The default is `6970`.
CommandPortLimit *integer*	`-commandportlimit` *integer*	Use this property to specify the maximum number of Telnet connections that the probe can make. The default is `10`.

Table 7. Properties and command line options (continued)		
Property name	**Command line option**	**Description**
EPIRPFtpCommand *string*	`-epirpftpcommand` *string*	Use this property to specify the complete FTP command that the probe uses to access the Interoperable Object Reference (IOR) file of the EPIRP object. The default is `""`. The syntax of the FTP command is `ftp://`*`user`*`:<password>@`*`host/path`*. Enter a valid username, host, and path. If you want to specify the password in encrypted format, leave `<password>` as it is. The probe retrieves the encrypted password from the **FtpPassword** property and places it in the command. If you do not want to specify the password in encrypted format, replace <password> with the password.
EPIRPIorFile *string*	`-epirpiorfile` *string*	Use this property to specify the location of the IOR file of the EPIRP object. The default is `/opt/5620sam/server/nms/cnbi/home/ior/EPIRP.ior`.
FlushBufferInterval *integer*	`-flushbufferinterval` *integer*	Use this property to specify how often (in seconds) the probe flushes all alerts in the buffer to the ObjectServer. The default is `0` (which instructs the probe to never flush the alerts to the ObjectServer).
FtpPassword *string*	`-ftppassword` *string*	Use this property to specify the password that the probe uses with the FTP command submitted by the CLI. The default is `""`. **Note :** Enter the password encrypted using nco_aes_crypt.
ManagerIdentifier *string*	`-manageridentifier` *string*	Use this property to specify the name of the manager that the NotificationIRP object references when events are available for sending. The default is `Netcool`.

Table 7. Properties and command line options (continued)

Property name	Command line option	Description
NotificationIrpName *string*	`-notificationirpname` *string*	Use this property to specify the name of the NotificationIRP object to which the the probe subscribes to receive new alarms as they are created. The default is `NotificationIRP=1`.
ORBDebug *string*	`-orbdebug` *string*	Use this property to specify whether or not ORB messages are written to a debug file. This property takes the following values: `false`: ORB messages are not written to a debug file. `true`: ORB messages are written in a debug file. The default is `false`.
ORBDebugFile *string*	`-orbdebugfile` *string*	Use this property to specify the file in which the ORB will write log messages when the **ORBDebug** property is set to `true`. The default is `$OMNIHOME/log/orb.debug`.
ORBLocalPort *integer*	`-orblocalport` *integer*	Use this property to specify the local port to which the Object Request Broker (ORB) listens. The default is `0` (ORB selects an available port at random).

Table 7. Properties and command line options (continued)		
Property name	**Command line option**	**Description**
ResyncBaseObjectFilter *string*	`-resyncbaseobjectfilter` *string*	Use this property to specify the base object from which the probe retrieves objects. If you want the probe to receive all alarms on the network, leave this property blank. If you want the probe to receive only those alarms on a given subnetwork, set this property to `SubNetwork=`*X*, where *X* identifies the subnetwork. If you want the probe to receive only those alarms generated by a specific element on a given subnetwork, set this property to `SubNetwork=`*X*`,ManagedElement=`*Y* where *X* identifies the subnetwork and *Y* identifies the managed element on that subnetwork. The default is `""`.
ResyncFilter *string*	`-resyncfilter` *string*	Use this property to specify the filter that the alarm IRP uses to reduce the number of the alarms sent to the probe during resynchronization. The default is `""`. **Note :** For details about specifying filtering criteria, see “Filtering alarms” on page 6.
SubscribeFilter *string*	`-subscribefilter` *string*	Use this property to specify the filter that the notification IRP uses to reduce the number of the new alarms sent to the probe. The default is `""`. **Note :** For details about specifying filtering criteria, see “Filtering alarms” on page 6.

Table 7. Properties and command line options (continued)

Property name	Command line option	Description

SubscribeNotifCategories *string*	`-subscribenotifcategories` *string*	Use this property to specify the notification categories to which the probe subscribes. You can specify one or more of the following categories:
		• 32.111-3 V8.0 - Alarms
		• 32.663 V8.1 - Kernel events
		• 32.353 V8.0 - Communications surveillance events
		Separate the notification categories with semicolons:
		`category1;category2; category3`
		For example, to subscribe to alarms and kernel events, enter the following categories:
		`32.111-3 V8.0;32.663 V8.1`
		The default is `""` (which subscribes the probe to all notification categories).

SystemDN *string*	`-systemdn` *string*	Use this property to specify the distinguished name of the system on which the SAM server is running as defined in the following configuration file: `$CNBI_HOME/config/ cnbi.properties` The default is `sam`.
SystemId *string*	`-systemid` *string*	Use this property to specify the system ID to use when acknowledging, clearing, or commenting on alarms within the SAM system. The default is `""`.
UserId *string*	`-userid` *string*	Use this property to specify the user ID used to connect to SAM when acknowledging, clearing, or commenting on alarms The default is `Netcool`.

Elements

The probe breaks event data down into tokens and parses them into elements. Elements are used to assign values to ObjectServer fields; the field values contain the event details in a form that the ObjectServer understands.

The following table describes the elements that the Probe for Alcatel-Lucent 5620 SAM 3GPP v8 generates. Not all the elements described are generated for each event; the elements that the probe generates depends upon the event type.

Table 8. Elements

Element name	Element description
$AckState	This element indicates whether the alarm has been acknowledged. It can take the following values: 1 - Acknowledged 2 - Unacknowledged
$AckTime	This element displays the time when the alarm was acknowledged.
$AckSytemId	This element indicates the ID of the system from which the alarm was acknowledged.
$AckUserId	This element contains the ID of the user who acknowledged the alarm.
$AdditionalText	This element contains additional information about the alarm.
$AlarmClearedTime	This element displays the time at which the alarm was cleared.
$AlarmId	This element contains the identifier of the alarm.
$AlarmRaisedTime	This element contains the time at which the alarm was raised.
$ChannelID	This element contains the channel through which the alarm was raised.
$ClearUserId	This element displays the ID of the user who cleared the alarm.
$HeartbeatPeriod	This element displays the length of the heartbeat period.

Table 8. Elements (continued)

Element name	Element description
$Event_name	This element indicates the type of the event. It can take the following values: x1 - Communications x2 - Processing error x3 - Environmental x4 - Quality of service x5 - Equipment x6 - Integrity violation x7 - Operational violation x8 - Physical violation x9 - Security service or mechanism violation x10 - Time domain violation
$EventTime	This element contains the time at which the event occurred.
$ManagedObjectInstance	This element identifies the object on which the alarm occurred.
$ManagerIdenifier	This element identifies the manager that subscribed to receive this notification.
$NotificationId	This element contains the notification ID of the event.
$PerceivedSeverity	This element indicates the perceived severity of the event. It can take the following values: 1 - Indeterminate 2 - Critical 3 - Major 4 - Minor 5 - Warning 6 - Cleared
$ProbableCause	This element indicates the probable cause of the event. For a full list of the probable causes that the probe supports, refer to its rules file.
$SpecificProblem	This element contains additional information about the problem being reported.
$Status	This element indicates the status of the alarm.

Table 8. Elements (continued)	
Element name	**Element description**
$SystemDN	This element contains the distinguished name of the system on which SAM is running.
$TriggerFlag	This element indicates what triggered the alarm.
$Domain_name	This element identifies the category of the event. It can take the following values: 32.111-3 V8.0 - Alarm 32.663 V8.1 - Kernel event 32.353 V8.0 - Communications surveillance event
$Type_name	This element indicates the type of the event being reported. It can take the following values: x1 - New alarm created x2 - Changed alarm x3 - Alarm with acknowledged state changed x4 - Comment added to an alarm x5 - Alarm cleared x6 - Alarm list rebuilt x7 - Potentially faulty alarm list built

Known Issues with the Probe for Alcatel-Lucent 5620 SAM 3GPP v8

This section contains the known issues that may occur while using the Probe for Alcatel-Lucent 5620 SAM 3GPP v8:

If the Netcool system clock and the SAM system clock are not synchronized, some events may be dropped

If the clock on the Netcool system and the clock on the SAM system are not synchronized, when a clear event does not hold the timestamp of when the alarm was cleared, and the alarm is sent to the ObjectServer with a timestamp of "now" (referring to the current time on the Netcool system) there is the possibility that this timestamp will be older than the timestamp of the original alarm (on the SAM system). If the time stamp is older it will be dropped by the deduplication trigger.

In this instance, both clocks should be synchronized, or the deduplication trigger should be disabled.

Note : Disabling the deduplication trigger may cause other performance issues.

Error messages

Error messages provide information about problems that occur while running the probe. You can use the information that they contain to resolve such problems.

The following table describes the error messages specific to this probe. For information about generic Netcool/OMNIbus error messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 9. Error messages

Error	Description	Action
`The acknowledge_alarms failed for the following alarms : alarm_id: alarm_id, reason: reason, failureCategory: failure_category`	The probe could not acknowledge the alarm for the reason stated in the message.	Address the exception reported, and resubmit the acknowledgeAlarm command.
`The acknowledge_alarms failed with the following exception : exception_description` `The alarm id information received is not valid`	The probe could not acknowledge the alarm because the ID that you specified is not valid.	Check the alarm ID, and resubmit the acknowledgeAlarm command.
`The acknowledge_alarms failed with the following exception : exception_description` `This indicates a generic software error.`	An error occurred while the probe was attempting to acknowledge an alarm.	Check the ID of the alarm that you want to acknowledge, and make sure you have specified valid values in the **`SystemID`** and **`UserID`** properties.
`There are several SubscriptionIds still active, which are results of previous subscribe operations performed by this IRPManager.` `SubscriptionId : subscription_id`	There is more than one subscription currently active.	Use the cleanupSubscription command to remove the superfluous subscriptions.
`Exception raised while calling 'get_subscription_status' :` `This indicates that the specified SubscriptionId is invalid.`	The probe could not display the details of the subscription because the subscription ID specified is not valid.	Check the subscription ID, and resubmit the getSubscriptionStatus command.

Table 9. Error messages (continued)

Error	Description	Action
`Exception raised while calling 'get_subscription_status' :` `This indicates a generic software error.`	An error occurred while the probe was attempting to display the status of a subscription.	Check the ID of the subscription whose status details you want to display, and make sure you have specified the correct value in the **ManagerIdentifier** property.
`The command received via the CLI is trying to detach the probe's current subscription.` `Aborting the operation otherwise the probe won't receive any events anymore.`	The subscription ID specified by the cleanupSubscription command is that of the current subscription.	Check the subscription ID, and resubmit the cleanupSubscription command to detach a subscription that is not currently being used by the probe.
`The probe doesn't know the 'managerReference' value for this subscriptionId. It is therefore unable to detach the subscription.`	The subscription specified by the cleanupSubscription command does exist on the manager specified by the **ManagerIdentifier** property.	Check the value specified by the **ManagerIdentifier** property, and resubmit the cleanupSubscription command.
`The detach failed with the following exception :` *`exception_description`* `This indicates that one of the parameters was invalid.`	The probe could not detach the subscription specified by the command line interface.	Check the value specified by the **ManagerIdentifier** property and the ID of the subscription that you are detaching, and resubmit the cleanupSubscription command.
`The detach failed with the following exception :` *`exception_description`* `This indicates a generic software error.`	An error occurred while the probe was attempting to unsubscribe from an object.	Check the ID of the subscription from which you wanted to detach, and make sure you have specified the correct value in the **ManagerIdentifier** property.
`The clear_alarms failed for the following alarms : alarm_id:` *`alarm_id`*`, reason:` *`reason`*	The probe could not clear the alarm for the reason stated in the message.	Address the exception reported, and resubmit the clearAlarm command.

Table 9. Error messages (continued)

Error	Description	Action
`The clear_alarms failed with the following exception : exception_description` `The alarm id information received is not valid.`	The probe could not clear the alarm because the ID that you specified is not valid.	Check the alarm ID, and resubmit the clearAlarm command.
`The clear_alarms failed with the following exception : exception_description` `This indicates a generic software error.`	An error occurred while the probe was attempting to clear an alarm.	Check the ID of the alarm that you want to clear, and make sure you have specified valid values in the **SystemID** and **UserID** properties.
`We never found the end of the comment. Syntax of the command is incorrect`	The comment portion of the commentAlarms command contains whitespace, which prevents the probe from being to determine the where the comment ends and the first alarm ID starts.	Resubmit the commentAlarms command, but put the comment portion of the command inside double quotation marks.
`The comment_alarms failed for the following alarms : => alarm_id: alarm_id, reason: reason)`	The probe could not add the comment to the the alarm for the reason stated in the message.	Address the exception reported, and resubmit the commentAlarms command.
`The comment_alarms failed with the following exception : exception_description` `The alarm id information received is not valid.`	The probe could not add the comment to the alarm because the ID that you specified is not valid.	Check the alarm ID, and resubmit the commentAlarms command.
`The comment_alarms failed with the following exception : exception_description` `This indicates a generic software error.`	An error occurred while the probe was attempting to add a comment an alarm.	Check the ID of the alarm to which you wanted to add an alarm, and make sure you have specified valid values in the **SystemID** and **UserID** properties.

Table 9. Error messages (continued)

Error	Description	Action
exception_description exception raised while calling 'get_alarm_list' Invalid filter or base object, calling 'get_alarm_list' with empty parameters	The probe could not use the filter that you specified with the **ResyncFilter** property or the **ResyncBaseObjectFilter** property.	Check the values that you specified in the filtering properties. For details about the values that you can specify, see "Filtering alarms" on page 6.
exception_description exception raised while calling 'get_alarm_list' Chances are you interrupted a previous resynchronization request, and the previous one is not finished yet Carrying on for now.	You requested a resynchronization operation while the probe was already performing one.	Wait until the current resynchronization operation has completed before submitting a request for the probe to perform another.
'ResyncBaseObjectFilter ' property value is invalid *property_value* Overwritting it to \"\" (empty string)	The probe could not use the filter that you specified with the **ResyncBaseObjectFilter** property. The property will instead return the alarms generated by all objects.	Check the value that you specified in the**ResyncBaseObjectFilter** property. For details about the values that you can specify, see "Filtering alarms" on page 6.
No encrypted 'FTPPassword' specified, the command will probably not work	You did not specify a password for the FTP command.	Specify a password, encrypted by nco_g_crypt, using the **FtpPassword** property. The probe inserts this password into the FTP command specified by the **EPIRPFtpCommand** property.
Unable to FTP the IOR, please check if the command is valid	The probe could not perform the FTP command.	Check the value that you specified using the **EPIRPFtpCommand** property. For details about the format of the FTP command, see "Using FTP to access the EPIRP object IOR file" on page 5.
'AgentHeartbeat' value does not belong to the accepted range (15mn-1440mn) Overwriting its value to 15mn	The value specified for the **AgentHeartbeat** property is not valid.	Change the value specified for the **AgentHeartbeat** property. Valid values are between 15 and 1440 inclusive.

Table 9. Error messages (continued)

Error	Description	Action
`Exception raised while calling 'attach_push': exception_description` `This indicates the manager_reference parameter is empty or invalid.`	An error occurred while the probe was attempting to subscribe to an object because the value set for the **ManagerReference** property is not valid or no value has been set for the property.	Check the value set for the **ManagerReference** property, and restart the probe.
`Exception raised while calling 'attach_push': exception_description` `This indicates the manager_reference is already subscribed to the notification category.`	The probe attempted to subscribe to a notification category to which it is already subscribed.	Use the cleanupSubscription command to remove any superfluous notification.
`Exception raised while calling 'attach_push': exception_description` `This indicates one or more of the notification categories is already supported.`	The probe attempted to subscribe to a notification category to which it is already subscribed.	Use the cleanupSubscription command to remove any superfluous notification.
`Exception raised while calling 'attach_push': exception_description` `This indicates a generic software error.`	An error occurred while the probe was attempting to subscribe to a notification category.	Use the cleanupSubscription command to remove any superfluous notification.
`createTopicConnection with details ( user: xxxxx, passw: xxxxx) failed : org.jboss.jms.exception . MessagingNetwork FailureException`	The probe cannot connect to the JMS.	Check that the user name and password are set correctly. If they are correct, check whether or not the SAM server has suspended the user name and password - the server will suspend a user name if it has been inactive for a set amount of time.

Table 9. Error messages (continued)		
Error	**Description**	**Action**
`Sleeping until receiving the SAM server's data`	When the probe starts, it connects to the SAM server using an HTTP connection on port 8080 and receives the resynchronization alarms. After resynchronization, the probe tries to connect to the JMS using port 1099. If the probe does not receive the `SystemInfoEvent` from the Alcatel SAM server through the JMS connection, the probe writes this sleeping message to the log file.	When the probe has written this message, it waits to receive the `SystemInfoEvent` for the time specified by the **`AliveTimeout`** property. When this time has elapsed, if the probe has still not received the `SystemInfoEvent`, it shuts down. To instruct the probe to wait longer to receive the `SystemInfoEvent` before shutting down, increase the value set for the **`AliveTimeout`** property.

ProbeWatch messages

During normal operations, the probe generates ProbeWatch messages and sends them to the ObjectServer. These messages tell the ObjectServer how the probe is running.

The following table describes the ProbeWatch messages that the probe generates. For information about generic Netcool/OMNIbus ProbeWatch messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 10. ProbeWatch messages		
ProbeWatch message	**Description**	**Triggers or causes**
`Shutting down.`	The probe is shutting down.	The probe is shutting down after performing the shutdown routine.

Appendix A. Notices and Trademarks

This appendix contains the following sections:

- Notices
- Trademarks

Notices

This information was developed for products and services offered in the U.S.A.

IBM® may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA

3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, ibm.com®, AIX, Tivoli®, and Netcool® are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Adobe, Acrobat, Portable Document Format (PDF), PostScript, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

SC14-7647-02