IBM® Tivoli® Netcool/OMNIbus Probe for Microsoft System Center Operations Manager (SCOM) 2012
3.0

Reference Guide
July 20, 2017

SC22-5475-05

Notice

Before using this information and the product it supports, read the information in Appendix A, “Notices and Trademarks,” on page 43.

Edition notice

This edition (SC22-5475-05) applies to version 3.0 of IBM Tivoli Netcool/OMNIbus Probe for Microsoft System Center Operations Manager 2012 and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC22-5475-04.

Contents

Document control page... v

Chapter 1. Probe for Microsoft SCOM 2012... 1
Summary.. 1
Supported deployment scenarios... 3
Installing probes.. 3
Environment variables.. 4
Firewall settings... 4
Configuring Microsoft SCOM 2012.. 4
Configuring the probe... 5
Data acquisition.. 5
Rules file.. 6
Managing subscriptions..7
Inactivity.. 8
Backoff strategy..8
Command port module... 8
Peer-to-peer failover functionality.. 13
Properties and command line options.. 14
Event acknowledgement automation.. 21
Enabling automatic acknowledgement of events.. 22
Disabling automatic event acknowledgment... 23
Elements.. 24
Error messages.. 28
ProbeWatch messages... 32
Running the probe... 32
Running multiple probes...33
Additional probe tools.. 33
Desktop and Webtop tools..33
Installing the desktop and Webtop tools...36
Using the desktop and Webtop tools..37
Batch update tool...37
Troubleshooting...38
Connector registration issues... 38
Known issues.. 39
The probe cannot be run as a Windows service..39
Tally mismatch for repeat alerts.. 40
Connector name subscriptions.. 40
Known issues with automatic event acknowledgement..41
AntiXssLibrary error.. 41
East Asian locale settings.. 41
Summary field gets truncated..42
Missing assembly file message...42
Unable to install the SCOM 2007 R2 and SCOM 2012 probe on the same system......................... 42

Appendix A. Notices and Trademarks...43
Notices... 43
Trademarks.. 44

Document control page

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

The documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM® Tivoli® Knowledge Center:

http://www-01.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/common/kc_welcome-444.html?lang=en

Table 1. Document modification history

Document version	Publication date	Comments
SC22-5475-00	August 31, 2012	First IBM publication.
SC22-5475-01	March 7, 2014	"Summary" on page 1 updated. The following property was added to "Properties and command line options" on page 14 • **`AlertBatchSize`** "Enabling automatic acknowledgement of events" on page 22 added.
SC22-5475-02	July 11, 2014	Details of obsolete known issue removed.
SC22-5475-03	December 11, 2014	"Summary" on page 1 updated. Details about configuring the socket-based command port updated in "Socket-based command port" on page 8. Details about using the command port module to send SCOM commands using either HTTP or HTTPS (HTTP over SSL) added in "HTTP/HTTPS command port" on page 10. "Missing assembly file message" on page 42 added to the Known Issues section.
SC22-5475-04	December 10, 2015	The following known issue was added to the guide: "Unable to install the SCOM 2007 R2 and SCOM 2012 probe on the same system" on page 42. This guide was updated to address APAR IV76122.
SC22-5475-05	July 20, 2017	"Peer-to-peer failover functionality" on page 13 updated. This guide was updated to address APAR IV95473.

Chapter 1. Probe for Microsoft SCOM 2012

Microsoft System Center Operations Manager (SCOM) 2012 is an event and performance management tool for Windows Server System.

The Probe for Microsoft SCOM 2012 can receive and acknowledge events from, and resolve alerts in, Microsoft SCOM 2012.

The probe communicates with Microsoft SCOM 2012 using the Operations Manager Connector Framework (OMCF) API exposed by the Microsoft SCOM 2012 Software Development Kit (SDK).

This guide contains the following sections:

- "Summary" on page 1
- "Supported deployment scenarios" on page 3
- "Installing probes" on page 3
- "Firewall settings" on page 4
- "Configuring Microsoft SCOM 2012" on page 4
- "Configuring the probe" on page 5
- "Data acquisition" on page 5
- "Properties and command line options" on page 14
- "Elements" on page 24
- "Error messages" on page 28
- "ProbeWatch messages" on page 32
- "Running the probe" on page 32
- "Additional probe tools" on page 33
- "Troubleshooting" on page 38
- "Known issues" on page 39

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:

Table 2. Summary

Probe target	Microsoft SCOM 2012 Microsoft SCOM 2012 R2
Probe executable name	`nco_p_scom2012`
Probe installation package	`omnibus-`*`arch`*`-probe-nco_p_scom2012-`*`version`*
Package version	3.0
Probe supported on	For details of supported operating systems, see the following Release Notice on the IBM Software Support website: #msscomv2_summary

Table 2. Summary (continued)	
Additional probe files	`nco_p_scom2012.bat`
Properties file	`%OMNIHOME%\probes\win32\scom2012.props`
Rules file	`%OMNIHOME%\probes\win32\scom2012.rules`

Additional files	`%OMNIHOME%\probes \win32\auto_acknowledge_trigger_unix.sql`
	`%OMNIHOME%\probes \win32\auto_acknowledge_trigger_windows.sql`
	`%OMNIHOME%\probes\win32\CreateScomGroup.xml`
	`%OMNIHOME%\probes\win32\nco_p_scom2012.bat`
	`%OMNIHOME%\probes\win32\nco_p_scom2012.exe`
	`%OMNIHOME%\probes \win32\nco_p_scom2012.exe.config`
	`%OMNIHOME%\probes \win32\netcool_tivoli_socket.dll`
	`%OMNIHOME%\probes\win32\README.scom_tool`
	`%OMNIHOME%\probes\win32\scom_tool.conf`
	`%OMNIHOME%\probes\win32\scom_tool.pl`
	`%OMNIHOME%\probes\win32\scom_tool.sql`
	`%OMNIHOME%\probes\win32\ScomCSNSProbe.dll`
	`%OMNIHOME%\probes\win32\TelnetScomProbe.pl`
	`%OMNIHOME%\probes\win32\TelnetScomProbe.VBS`
	`%OMNIHOME%\probes\win32\UnixScomTools.cgi`
	`%OMNIHOME%\probes\win32\UnixScomTools.xml`
	`%OMNIHOME%\probes\win32\update_scom_unix.sql`
	`%OMNIHOME%\probes\win32\update_scom_windows.sql`
	`%OMNIHOME%\probes\win32\WindowsScomTools.cgi`
	`%OMNIHOME%\probes\win32\WindowsScomTools.sql`
	`%OMNIHOME%\probes\win32\WindowsScomTools.xml`

Requirements	The probe requires .NET Framework 3.5. If you have a later version of .Net Framework installed (for example, version 4.0), you will need to install .NET Framework 3.5 as well. Multiple .NET frameworks can exist in a single environment and will work side-by-side. The probe will still rely on .NET Framework 3.5 to run and will use the .Net Framework 4.0 library for additional support as required. For details of any additional software that this probe requires, refer to the `description.txt` file that is supplied in its download package.

Table 2. Summary (continued)

Connection method	Microsoft Operations Manager Connector Framework (OMCF) The connection is made over TCP and is secured using the Kerberos network authentication protocol.
Remote connectivity	Available
Multicultural support	Not Available
Peer-to-peer failover functionality	Available
IP environment	IPv4 and IPv6
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*.

Supported deployment scenarios

You might need to install more than one instance of the probe, depending on the Microsoft SCOM 2012 deployment scenario you are using.

The probe supports the following Microsoft SCOM 2012 deployment scenarios:

- A single server with a single management group
- Multiple servers (a cluster) with a single management group

Note : The probe does not support multiple, connected management groups. If you are using this deployment scenario, you can install a separate instance of the probe for each management group.

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*.

Environment variables

Environment variables are specific preset values that establish the working environment of the probe.

The Netcool/OMNIbus runtime environment settings required to run probes are set during the installation of Netcool/OMNIbus. This includes the environment variables `%NCHOME%`, `%OMNIHOME%`, and `%PATH%`.

For more information about setting Netcool/OMNIbus environment variables, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide* (SC14-7604).

Firewall settings

When the probe is installed on a different machine to the Microsoft SCOM 2012 Root Management Server (RMS) host, you must configure any intervening firewall to allow communication between the two hosts.

To enable communication between the probe and the RMS, configure the firewall to allow the probe to connect to TCP port 5724 on the RMS host machine.

If you configure the probe for peer-to-peer failover, you must configure the firewall to allow connections to the peer probe using the port values specified by the **PeerPort** property.

To use the probe's command line interface (CLI), you must configure the firewall to allow connections to the port specified by the **CommandPort** property. If you are using a second instance of the probe in a failover configuration, you must also configure the firewall to allow connections to the port specified by the **PeerCommandPort** property.

Configuring Microsoft SCOM 2012

The probe requires a Windows user account on the same domain as Microsoft SCOM 2012. This can be an existing user account or a new account.

Authentication

The Probe for Microsoft SCOM 2012 uses Windows user accounts to authenticate the connection.

You must use the Microsoft SCOM 2012 **Operations Console** to assign the `Operations Manager Administrator` user role to the user account that you are using to run the probe.

You can also use Microsoft Active Directory to create a group (for example, "Netcool Connectors") and user login (for example, "netcool") for use by the probe. You can then assign the `Operations Manager Administrator` user role to this group. This removes the need to specify values for the **ConnectorUser**, **ConnectorPassword**, and **ConnectorDomain** properties in the properties file.

Consult your Microsoft documentation for instructions about how to create the type of user account that you require.

Connection method

The connection between the probe and Microsoft SCOM 2012 is made over TCP and is secured using the Kerberos authentication mechanism provided by Microsoft Active Directory.

Configuring a subscription

After you have successfully installed, configured, and started the probe, use the **Product Connector Subscription** wizard in Microsoft SCOM 2012 to configure the automatic forwarding of alerts to the probe. Consult your Microsoft documentation for instructions about how to do this.

Configuring the probe

Before running the probe for the first time, you must configure the probe to work with your operating environment.

Before running the probe, you must configure the following minimum set of properties:

- **ConnectorName**

  This property specifies the unique name used to register the probe as a connector in Microsoft SCOM 2012.

- **ConnectorDomain**

  This property specifies the domain of the Windows account used for client authentication during connection to Microsoft SCOM 2012.

- **ConnectorUser**

  This property specifies the user name of the Windows account used for client authentication during connection to Microsoft SCOM 2012.

- **ConnectorPassword**

  This property specifies the password of the Windows account used for client authentication during connection to Microsoft SCOM 2012.

- **ScomHost**

  This property specifies the IP address or Fully Qualified Domain Name (FQDN) of the Root Management Server (RMS) that the probe connects to.

- **ScomSdkDir**

  This property specifies the directory where the probe can locate the following SDK .NET libraries that it requires for communication with Microsoft SCOM 2012:

  - `Microsoft.EnterpriseManagement.Core.dll`
  - `Microsoft.EnterpriseManagement.OperationsManager.dll`
  - `Microsoft.EnterpriseManagement.Runtime.dll`

  This property specifies the directory where the probe can locate the following SDK .NET libraries that it requires for communication with Microsoft SCOM 2007 R2:

  - `Microsoft.EnterpriseManagement.OperationsManager.dll`
  - `Microsoft.EnterpriseManagement.OperationsManager.Common.dll`

Data acquisition

Each probe uses a different method to acquire data. Which method the probe uses depends on the target system from which it receives data.

The probe uses the Operations Manager Connector Framework (OMCF) to create a connector on Microsoft SCOM 2012 to which it subscribes to receive alerts.

The following steps describe the data acquisition process:

1. The probe connects to Microsoft SCOM 2012 using the IP address or Fully Qualified Domain Name (FQDN) of the Root Management Server (RMS), as specified by the **ScomHost** property.
2. On connection, the probe authenticates with the Microsoft SCOM 2012 server.

For information about authentication options, see "Configuring Microsoft SCOM 2012" on page 4.

3. On successful authentication, the probe registers itself as a connector in Microsoft SCOM 2012 using the unique name specified by the **ConnectorName** property.

   The probe is now subscribed to Microsoft SCOM 2012. If the connector already exists in Microsoft SCOM 2012, the probe will reuse it. In this case, the probe will also use the existing subscription settings, unless the **CleanStart** property is set to `true`.
4. On subscription, the probe polls Microsoft SCOM 2012 for new alerts at intervals specified by the **PollInterval** property.
5. When polled, Microsoft SCOM 2012 keeps sending alerts to the probe until the probe acknowledges their receipt.
6. The probe parses the alerts and sends them as events to the ObjectServer.

   If the probe receives a resolved alert from Microsoft SCOM 2012, it sends it to the ObjectServer as a cleared event.

More details of how the probe acquires data are described in the following topics:

- "Rules file" on page 6
- "Managing subscriptions" on page 7
- "Inactivity" on page 8
- "Backoff strategy" on page 8
- "Command port module" on page 8
- "Peer-to-peer failover functionality" on page 13

Rules file

The rules file `scom2012.rules` defines how the probe processes event data to create meaningful Netcool/OMNIbus alerts.

The probe takes an event stream and parses it into elements. It then processes the event elements based on the logic in the rules file. It assigns the elements to ObjectServer fields and forwards them to the ObjectServer, where they are inserted as alerts into the `alerts.status` table.

The elements parsed by the probe are listed in "Elements" on page 24. For more information about using rules files, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide* (SC14-7608).

Severity level mappings

Netcool/OMNIbus uses different terms than Microsoft SCOM 2012 to describe the severity level of an alert. The following table shows how the probe maps Microsoft SCOM 2012 severity levels to the Netcool/OMNIbus severity levels used by the ObjectServer.

Table 3. Severity level mappings

Microsoft SCOM 2012 severity level	**Netcool/OMNIbus severity level**
Information	Clear
Warning	Warning
Error with severity Low	Minor
Error with severity Normal	Major
Error with severity High	Critical

NodeAlias event field

To enable ObjectServer automations to connect to the probe's command port, the `NodeAlias` field of each event stores the TCP address of the probe in the following format:

`host_name:command_port_number`

Where *host_name* is the name of the probe's host machine and *command_port_number* is the value of the **`CommandPort`** property.

Note : If you change the value of the **`CommandPort`** property, existing events in the ObjectServer will still contain the old port number in their `NodeAlias` fields. This might cause the failure of any automation that uses the value of the `NodeAlias` field. In this case, you must use the Microsoft SCOM 2012 **Operations Console** to manually re-send the existing events, thereby updating them in the ObjectServer with the new `NodeAlias` field value.

Managing subscriptions

You can specify how the probe manages subscriptions to Microsoft SCOM 2012 using the **`CleanUpOnShutdown`** and **`CleanStart`** properties.

Clean up on shutdown

You can use the **`CleanUpOnShutdown`** property to specify whether or not the probe de-registers its connector and unsubscribes from Microsoft SCOM 2012 at the end of a session. The probe can only clean up connectors and subscriptions that have been fully registered during a session. If, for example, the probe shuts down before the connector or subscription is fully established, the probe cannot clean up that connector or subscription on shut down.

If you specify a value of `true` for this property, the probe will remove the connector and clean up the current subscription. When the probe reconnects, it will recreate the connector and subscribe only to new alerts. If you specify a value of `false` for this property, the probe will maintain the connector and subscription between sessions. The default is `false`.

Note :

If you use the Ctrl+C command in a Windows console to shut down the probe, it can take up to one minute for the probe to perform the clean up process. During this time, the progress of the clean up is not reported in the console. It is important to allow enough time for the probe to perform the clean up before trying to shut it down a second time. Using Ctrl+C a second time, while clean up is in progress, might result in an incomplete clean up.

If you deploy two instances of the probe in a peer-to-peer failover configuration, the clean up on shut down function is only available to the master probe and not to the slave probe.

Clean start

You can use the **`CleanStart`** property to specify whether or not the probe processes alerts from a previous subscription when it starts up. If you specify a value of `true` for this property, the probe ignores any alerts created in a previous subscription. The probe will only process alerts created since the start of the current session.

If you specify a value of `false` for this property, the probe will process any alerts that were created since the previous session. The default is `false`.

Note : You can manually retrieve or remove connectors that lose their registration IDs. For more information, see "Connector registration issues" on page 38.

Inactivity

The probe has a timeout facility that enables it to disconnect from Microsoft SCOM 2012 if it does not receive an alert within a specified amount of time.

You can use the **Timeout** property to specify the period of time (in seconds) for which the probe waits for new alerts from Microsoft SCOM 2012 before shutting down. If the probe is still busy retrieving a previous alert, it will not disconnect when the timeout interval is reached.

As a guideline, you can usually specify a value for this property equal to the longest idle time before an event is raised in the actual deployment environment. In cases where there is a recurring alert at a regular interval from Microsoft SCOM 2012 (such as a heartbeat signal), you can specify a value slightly higher than the alert interval.

The default value of the **Timeout** property is `0` seconds, which instructs the probe to wait for alerts indefinitely.

Backoff strategy

If the **Retry** property is set to `true`, and the probe fails to establish a connection or loses an existing connection to Microsoft SCOM 2012, the probe reverts to a backoff strategy.

The probe's backoff strategy is to try to reestablish a connection at successive intervals of one second, two seconds, four seconds, eight seconds, and so on, up to a maximum of 4096 seconds. When the maximum interval is reached, the probe shuts down.

Note : If the probe is connecting to a Microsoft SCOM 2012 RMS cluster, set the **Retry** property to `true`. This enables the probe to tolerate any interim connection failure that might occur during the internal failover process in the target system.

Overriding the backoff strategy

You can use the **RetryMaxTime** property to specify a maximum period of time (in seconds) for which the probe will attempt to connect to Microsoft SCOM 2012 using the backoff strategy. If a successful login does not occur within this period, the probe will stop trying to connect to Microsoft SCOM 2012. The default value of this property is `9000` seconds (2.5 hours). Specify a value of `0` to make the probe retry the connection indefinitely.

If you want the probe to retry the login at a regular interval instead of at successively increasing intervals, you can use the **RetryConstantWait** property to specify the regular interval (in seconds). The default value of this property is `0`, which instructs the probe to use the standard backoff interval.

Command port module

The probe is supplied with a command port module that can be configured to support either command line interface (CLI) commands sent through a socket or SCOM commands sent using HTTP/HTTPS.

The following topics describe how to configure both types of command port and the commands that you can send with them.

Socket-based command port

By default, a socket-based command port module starts when the probe starts. For details of using an HTTP/HTTPS command port instead, see “HTTP/HTTPS command port” on page 10.

Configuring the socket-based command port

To use the socket-based command port feature, you must enable it by using the **CommandPort** property to specify a port through which commands will be sent. The default is `6970`. You can use the **CommandPortLimit** property to specify the maximum number of connections to allow at one time. The default is `10` connections.

If you are using a second instance of the probe in a failover configuration, you must use the **PeerCommandPort** property to specify a port through which commands can be sent to the slave probe.

When you want to issue CLI commands, use Telnet to connect through the specified port.

Note : Depending on the version of Windows you are using, you might have to download Telnet from http://technet.microsoft.com/ and install it manually.

CLI commands

The following table describes the CLI commands that you can use with the socket-based command port.

Table 4. CLI commands

Command	Description
acknowledge_alarm	Use this command to acknowledge one or more alarms. The command takes a list of alarm identifiers separated by semicolons as parameters in the following format: **acknowledge_alarm** *alarm_id_01*;*alarm_id_02*; ... ;*alarm_id_nn* Where *alarm_id_01*;*alarm_id_02*; ... *alarm_id_nn* represents a list or *nn* alarm IDs. The system returns the identifiers of any alarms that cannot be acknowledged. **Note :** You can use the **AckedResolutionState** property to specify a value for the `ResolutionState` field of an alarm acknowledged using the CLI. The default value is 85, which is the standard value understood by Microsoft SCOM 2012 to identify an alarm as acknowledged.
resolve_alarm	Use this command to resolve alarms in the Microsoft SCOM 2012 server. The command takes a list of alarm identifiers separated by semicolons as parameters, in the following format: **resolve_alarm** *alarm_id_01*;*alarm_id_02*; ... ;*alarm_id_nn* Where *alarm_id_01*;*alarm_id_02*; ... *alarm_id_nn* represents a list or *nn* alarm IDs. The system returns the identifiers of any alarms that cannot be resolved. **Note :** You can use the **ClearedResolutionState** property to specify a value for the `ResolutionState` field of an alarm resolved using the CLI command. The default value assigned is 255, which is the standard value understood by Microsoft SCOM 2012 to identify an alarm as resolved.
set_ticket_id	Use this command to assign a value to the ticket identifier of an alarm on the Microsoft SCOM 2012 server. The command takes the alarm identifier and the value to be assigned to the ticket identifier as parameters, in the following format: **set_ticket_id** *alarm_id ticket_id*

Table 4. CLI commands (continued)

Command	Description
set_field	Use this command to assign values to individual alarm fields on the Microsoft SCOM 2012 server. The command takes a list of alarm identifiers, field names, and field values as parameters, in the following format: **set_field** *alarm_id_01(field_name=field_value) alarm_id_02(field_name=field_value)* ... *alarm_id_nn(field_name=field_value)* Where *field_name* in each case can be one of the following fields: • `CustomField`*n* (where *n* is an integer from 1 to 10) • `Owner` • `ResolutionState` • `TicketId` The system returns the identifiers of any alarms for which field values cannot be assigned.
exit/quit	Use this command to close the connection.
help	Use this command to display online help about the CLI.
version	Use this command to display the version of the probe and the CLI.

HTTP/HTTPS command port

The probe is supplied with a command port module that supports commands sent using either HTTP or HTTPS (HTTP over SSL). To enable the HTTP/HTTPS command port module, set either or both of the following properties to `true`:

- **NHttpd.EnableHTTP**
- **NHttpd.SSLEnable**

If you set the value of both properties above to FALSE (their default values in the core library) the socket-based command port module starts when probe starts. For details of using the socket-based command port, see "Socket-based command port" on page 8.

Note : The HTTP/HTTPS command port feature relies on sending commands using the following script:

`%OMNIHOME%/bin/nco_http`

Configuring the probe to process SCOM commands sent using HTTP

To configure the probe to process SCOM commands sent using HTTP, set the following core properties in the `scom2012.props` file:

```
NHttpd.EnableHTTP     : true
NHttpd.ListeningPort  : 8080
```

Configuring the probe to process SCOM commands sent using HTTPS

To configure the probe to process SCOM commands sent using HTTPS, use the following steps:

1. Configure Netcool/OMNIbus for SSL communication between remote systems and probes. For details of how to do this, refer to the following page in the Netcool/OMNIbus Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ_8.1.0/com.ibm.netcool_OMNIbus.doc_8.1.0/omnibus/wip/probegtwy/task/omn_prb_enableremotessl.html?lang=en

2. Set the following core properties in the `scom2012.props` file as shown:

   ```
   NHttpd.SSLEnable          : true
   NHttpd.SSLListeningPort   : 7777
   NHttpd.SSLCertificate     : "CA"
   ```

   Note : **`NHttpd.SSLCertificate`** refers to the label of the SSL certificate, not to the path location of the certificate.

   The values set for **`NHttpd.SSLListeningPort`** and **`NHttpd.SSLCertificate`** can be any valid values defined by the user. The values 7777 and CA above are just examples.

3. Set the following property in the `nco_http.props` file as shown:

   ```

   NHttpd.SSLEnable: TRUE
   ```

4. Enable the ObjectServer to support SSL using the Netcool/OMNIbus Servers Editor tool. For details of how to do this, refer to the following page in the Netcool/OMNIbus Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ_8.1.0/com.ibm.netcool_OMNIbus.doc_8.1.0/omnibus/wip/install/concept/omn_con_customizingservdefns.html?lang=en

SCOM commands

The following table describes the SCOM commands that you can issue using the HTTP command port.

Table 5. SCOM commands

Command	Description
`acknowledge_alarm`	Use this command to acknowledge one or more alarms. The command takes a list of alarm identifiers as parameters, in the following format: `nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"acknowledge_alarm","params":[alarm_id_01;alarm_id_02;alarm_id_03]}' -method POST` The system returns the identifiers of any alarms that cannot be acknowledged. **Note :** You can use the **AckedResolutionState** property to specify a value for the `ResolutionState` field of an alarm acknowledged using the SCOM command. The default value is 85.

Table 5. SCOM commands (continued)

Command	Description
resolve_alarm	Use this command to resolve alarms in the Microsoft SCOM 2012 server. The command takes a list of alarm identifiers as parameters, in the following format: nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"**resolve_alarm**","params":[*alarm_id_01*;*alarm_id_02*;*alarm_id_03*]}' -method POST The system returns the identifiers of any alarms that cannot be resolved. **Note :** You can use the **ClearedResolutionState** property to specify a value for the ResolutionState field of an alarm resolved using the SCOM command. The default value is 255.
set_ticket_id	Use this command to assign a value to the ticket identifier of an alarm on the Microsoft SCOM 2012 server. The command takes the alarm identifier and the value to be assigned as parameters, in the following format: nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"**set_ticket_id**","params":[*alarm_id*;*ticket_id*]}' -method POST
set_field	Use this command to assign values to individual alarm fields on the Microsoft SCOM 2012 server. The command takes a list of alarm identifiers, field names, and field values as parameters, in the following format: nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"**set_field**", "params":[*alarm_id_01(field_name=field_value)*;*alarm_id_02(field_name=field_value)*]}' -method POST Where *field_name* in each case can be one of the following fields: • CustomField*n* (where *n* is an integer from 1 to 10) • Owner • ResolutionState • TicketId The system returns the identifiers of any alarms for which field values cannot be assigned.
help	Use this command to display online help about the command port. nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"**help**","params":[]}' -method POST
version	Use this command to display the version of the probe and the command port. nco_http -uri http://localhost:8080/probes/nco_p_scom2012 -datatype application/json -data '{"command":"**version**","params":[]}' -method POST

Note : If you are using HTTPS to send commands instead of HTTP, make the following adjustments to the commands described in the SCOM commands table:

- In each command, use `https` in the URI instead of `http`.
- For the port number, use the value that you specified for the **NHttpd.SSLListeningPort** property in the `SCOM2012.props` file.

The **acknowledge** and **resolve** commands across HTTP/HTTPS are also available using the Netcool/OMNIbus EventList GUI. For details about how to set this up, see "Desktop and Webtop tools" on page 33.

Troubleshooting Command Port modes

You can start the probe in one of the following Command Port modes:

- TELNET
- HTTP
- HTTPS

Events generated from the target SCOM system are tagged to the Command Port mode that the probe is running at the time that they are generated. This affects how they can later be acknowledged or resolved.

The @URL column of the Netcool/OMNIbus `status` table indicates the Command Port mode that each event requires when acknowledging or resolving events using the Netcool/OMNIbus EventList GUI. The **acknowledge** or **resolve** command can only be sent using a probe running in the same mode as that tagged for the event in the @URL column.

Example

If five events are generated from SCOM while the probe is running in TELNET Command Port mode, when these events are acknowledged or resolved using the Netcool/OMNIbus EventList GUI, the **acknowledge** or **resolve** command must be sent through TELNET. So at that point in time, the probe must be running the TELNET Command Port for the **acknowledge** or **resolve** command to be successful.

However, if the probe is restarted in another mode, for example HTTP, and then the **acknowledge** or **resolve** commands are sent using the EventList GUI, those commands will fail because the events have been tagged for TELNET mode (the mode in which the probe was running when they were generated).

There are two possible solutions:

- Either run the probe in the same Command Port mode that it was running when the events were generated, and then issue the **acknowledge** or **resolve** command from the Netcool/OMNIbus EventList GUI.
- Or forward the same events from SCOM to the probe again. This will update the @URL column of the Netcool/OMNIbus `status` table to the Command Port mode that the probe is currently running.

Peer-to-peer failover functionality

The probe supports failover configurations where two probes run simultaneously. One probe acts as a master probe and sends alerts to the ObjectServer. The other probe acts as a slave probe and remains on standby. If the master probe fails, the slave probe is activated.

Note : Microsoft SCOM 2012 does not allow the same event to be automatically forwarded to more than one connector at a time. This means that the peer-to-peer failover behavior of the Probe for Microsoft SCOM 2012 is slightly different from the standard peer-to-peer failover behavior of most probes.

You can use a failover configuration by starting two instances of the probe, one master and one slave. Both probes must use the same connector name, as specified by the **ConnectorName** properties in their respective properties files. Both probes must also use the same peer port, as specified by their respective **PeerPort** properties.

If you set values for the **ConnectorUser** and **ConnectorPassword** properties on the master instance of the probe, you must also set these properties to the same values on the slave instance of the probe.

Note : If you have assigned the Operations Manager Administrator user role to the probe's user group in Microsoft SCOM 2012, you do not have to specify a value for the **`ConnectorUser`** and **`ConnectorPassword`** properties.

Unlike standard peer-to-peer failover behavior for most probes, the slave instance of the probe does not connect to Microsoft SCOM 2012 at the same time as the master instance. Instead, the slave probe connects to Microsoft SCOM 2012 and starts to process events only after it detects the loss of a heartbeat signal from the master probe. When the master probe is running again, the slave probe stops polling Microsoft SCOM 2012 for new events.

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 the master probe and slave probe, except for the **`ConnectorName`** property.

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

```
ConnectorName        :    "connector_name"
Server                     :    "NCOMS"
RulesFile               :    "master_rules_file"
MessageLog              :    "master_log_file"
PeerHost                :    "slave_hostname"
PeerPort                :    5555 # [communication port between master and slave probe]
Mode                    :    "master"
```

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

```
ConnectorName        :    "connector_name"
Server                                    :    "NCOMS"
RulesFile               :    "slave_rules_file"
MessageLog              :    "slave_log_file"
PeerHost                :    "master_hostname"
PeerPort                :    5555 # [communication port between master and slave probe]
Mode                    :    "slave"
```

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 6. Properties and command line options

Property name	Command line option	Description
`AlertsBatchSize` *integer*	`-alertbatchsize` *integer*	Use this property to specify the maximum number of events the probe sends in a batch. The default is `0` (the probe does not send events in batch mode.)

Table 6. Properties and command line options (continued)

Property name	Command line option	Description
AckedResolutionState *integer*	-ackedresolutionstate *integer*	Use this property to specify the value to be set for the ResolutionState field of an alarm when you acknowledge that alarm in Microsoft SCOM 2012 using the CLI. Values between 0 and 255 (inclusive) are valid for the ResolutionState field. The default is 85. **Note :** Do not use this property unless it is required as part of a customization of Microsoft SCOM 2012.
CleanStart *string*	-cleanstart (This is equivalent to **CleanStart** with a value of true.) -nocleanstart (This is equivalent to **CleanStart** with a value of false.)	Use this property to specify whether the probe retrieves alerts created in Microsoft SCOM 2012 since the last time the probe shut down. This property takes the following values: false: The probe retrieves any alerts created since it last shut down. true: The probe ignores any alerts created since it last shut down. The default is false.
CleanUpOnShutdown *string*	-cleanuponshutdown (This is equivalent to **CleanUpOnShutdown** with a value of true.) -nocleanuponshutdown (This is equivalent to **CleanUpOnShutdown** with a value of false.)	Use this property to specify whether the probe de-registers its connector and unsubscribes from Microsoft SCOM 2012 at the end of a session. false: The probe maintains the connector and subscription between sessions. true: The probe removes the connector and cleans up the current subscription at the end of a session. The default is false. For more information about using this property, see "Managing subscriptions" on page 7.

Table 6. Properties and command line options (continued)

Property name	**Command line option**	**Description**
`ClearedResolutionState` *integer*	`-clearedresolutionstate` *integer*	Use this property to specify the value to be set for the `ResolutionState` field of an alarm when you resolve that alarm in Microsoft SCOM 2012 using the CLI. Values between 0 and 255 (inclusive) are valid for the `ResolutionState` field. The default is 255. **Note :** Do not use this property unless it is required as part of a customization of Microsoft SCOM 2012.
`CommandPort` *integer*	`-commandport` *integer*	Use this property to specify the port through which users can send commands to Microsoft SCOM 2012 using the CLI. The default is `6970`.
`CommandPortLimit` *integer*	`-commandportlimit` *integer*	Use this property to specify the maximum number of Telnet connections that can be made to the probe. The default is `10`.
`ConnectorDomain` *string*	`-connectordomain` *string*	Use this property to specify the domain of the Windows account used for client authentication during connection to Microsoft SCOM 2012. If you have assigned the `Operations Manager Administrator` user role to the probe's user group in Microsoft SCOM 2012, you do not have to specify a value for this property. The default is `""`.

Table 6. Properties and command line options (continued)

Property name	Command line option	Description
ConnectorName *string*	-connectorname *string*	Use this property to specify the unique name used to register the probe as a connector in Microsoft SCOM 2012. The connector name is case-sensitive. The default is "Netcool probe". **Note :** If you are running multiple stand-alone instances of the probe, each instance must have a unique connector name. However, if you are running two instances of the probe in a failover configuration, both instances must use the same connector name. For more information, see "Peer-to-peer failover functionality" on page 13. **Note :** Each unique **ConnectorName** must not contain duplicate settings. For more information, see "Connector name subscriptions" on page 40
ConnectorPassword *string*	-connectorpassword *string*	Use this property to specify the password of the Windows account used for client authentication during connection to Microsoft SCOM 2012. If you have assigned the Operations Manager Administrator user role to the probe's user group in Microsoft SCOM 2012, you do not have to specify a value for this property. The default is "". **Note :** Use this property in conjunction with the generic Netcool/OMNIbus **ConfigKeyFile** property. Use the Netcool/OMNIbus nco_aes_crypt utility to generate an encrypted version of the password for use in the properties file. For more information about using the nco_aes_crypt utility, see the *IBM Tivoli Netcool/OMNIbus Administration Guide* (SC14-7605).

Table 6. Properties and command line options (continued)

Property name	Command line option	Description
ConnectorUser *string*	-connectoruser *string*	Use this property to specify the user name of the Windows account used for client authentication during connection to Microsoft SCOM 2012. If you have assigned the Operations Manager Administrator user role to the probe's user group in Microsoft SCOM 2012, you do not have to specify a value for this property. The default is "".
PeerCommandPort *integer*	-peercommandport *integer*	Use this property to specify the port on the peer probe through which users can send commands to Microsoft SCOM 2012 using the CLI. You only need to specify a value for this property in cases where you have an ObjectServer automation customization that uses the event elements peerHost (the value specified by the generic Netcool/OMNIbus **PeerHost** property) and peerCommandPort (the value specified by this property). The default is 6970. For more information, see "Peer-to-peer failover functionality" on page 13.
PollInterval *integer*	-pollinterval *integer*	Use this property to specify the interval (in seconds) at which the probe polls Microsoft SCOM 2012 for new alerts. To prevent the connection timing out, use polling intervals of less than 30 minutes. You can disable this property by specifying a value of 0. This will result in continuous polling by the probe, leading to high CPU usage. The default is 10. **Note :** If you enable the **Timeout** property, you must specify a value for the **PollInterval** property that is less than the value you specify for the **Timeout** property. **Note :** Microsoft SCOM only polls for alarms at a minimal interval of 60 seconds before sending events to the probe, so the probe may experience a delay in receiving alarms.

Table 6. Properties and command line options (continued)

Property name	**Command line option**	**Description**
Retry *string*	-retry (This is equivalent to **Retry** with a value of true.) -noretry (This is equivalent to **Retry** with a value of false.)	Use this property to specify whether the probe retries the connection to Microsoft SCOM 2012 when it fails to establish a connection or loses an existing connection. This property takes the following values: false: The probe does not retry the connection. true: The probe retries the connection. The default is false. **Note :** If the probe is connecting to a Microsoft SCOM 2012 RMS cluster, set the **Retry** property to true. This enables the probe to tolerate any interim connection failure that might occur during the internal failover process in the target system. For more information about this property, see "Backoff strategy" on page 8.
RetryConstantWait *integer*	-retryconstantwait *integer*	Use this property to specify a regular interval (in seconds) for the backoff strategy enabled by the **Retry** property. The default is 0. **Note :** This property overrides the normal operation of the **Retry** property. For more information, see "Backoff strategy" on page 8.
RetryMaxTime *integer*	-retrymaxtime *integer*	Use this property to specify a maximum time period (in seconds) for which the probe will attempt to connect to Microsoft SCOM 2012 using the backoff strategy. The default is 9000 (2.5 hours). **Note :** This property overrides the normal operation of the **Retry** property. For more information, see "Backoff strategy" on page 8.

Table 6. Properties and command line options (continued)

Property name	Command line option	Description
`ScomHost` *string*	`-scomhost` *string*	Use this property to specify the IP address or FQDN of the RMS that the probe connects to. IPv6 addresses must be enclosed in square brackets. For example: `[fe80::202:b3ff:fe1e:8329]`. The default is `localhost`. **Note :** The RMS can be on a different host than the probe. If so, specify the name of the stand-alone host. If the RMS is part of a clustered RMS, specify the cluster name as the host value.
`ScomSdkDir` *string*	`-scomsdkdir` *string*	Use this property to specify the directory where the probe can locate the SDK .NET libraries it requires for communication with Microsoft SCOM 2012. If you are installing the probe on the same host machine as Microsoft SCOM 2012, it is likely that these libraries have been installed to the Global Assembly Cache (GAC) of the host machine during the Microsoft SCOM 2012 installation. In this case, the libraries are globally available on the host machine and you do not need to specify a value for the **`ScomSdkDir`** property. Consult your Microsoft documentation for information about how to verify the contents of the GAC. **Note :** If the Probe for Microsoft SCOM 2012 is not running on the same server as Microsoft SCOM 2012 you need to copy the DLL files specified in the "Configuring the probe" on page 5 section to the probe server and specify the **`ScomSdkDir`** to that location. The default is `"C:\\Program Files\\System Center 2012\\Operations Manager\\Server\\SDK Binaries"`. **Note :** Back slashes (\) are treated as escape characters. Therefore, you must use double back slashes (\\) to ensure that the directory path is read correctly.

Table 6. Properties and command line options (continued)

Property name	Command line option	Description
`Timeout` *integer*	`-timeout` *integer*	Use this property to specify the period of time (in seconds) for which the probe waits for new alerts from Microsoft SCOM 2012 before shutting down. If the probe is still busy retrieving a previous alert, it will not disconnect when the timeout interval is reached. The default is `0`. **Note :** For more information about specifying a value for this property, see "Inactivity" on page 8.

Event acknowledgement automation

The Probe for Microsoft SCOM 2012 acknowledges events that are older than 5 minutes (300 seconds), and then updates Microsoft SCOM 2012 with the new alert details.

On the Windows operating system the Probe for Microsoft SCOM 2012 uses the following scripts to enable automatic event acknowledgement. The default location for these scripts is `%OMNIHOME%\probes\win32\`

- `auto_acknowledge_trigger_windows.sql` - this script monitors the ObjectServer for events and starts an external procedure to update Microsoft SCOM 2012 with new events.
- `update_scom_windows.sql` - this script is the external procedure that updates Microsoft SCOM 2012

  .
- `TelnetScomProbe.VBS` - this script starts Telnet using the `netcool_tivoli_socket.dll` library.
- `netcool_tivoli_socket.dll` - this is a `Win32` library that provides an implementation of the Telnet client for use on Windows.

On UNIX based operating systems the Probe for Microsoft SCOM 2012 uses the following scripts to enable automatic event acknowledgement. To run on a UNIX based operating system the following scripts must be copied from the default location to `$OMNIHOME/probes/`*`arch`* where *arch* is the name of the operating system.

- `auto_acknowledge_trigger_unix.sql` - this script monitors the ObjectServer for events and starts an external procedure to update Microsoft SCOM 2012 with new events.
- `update_scom_unix.sql` - this script is the external procedure that updates Microsoft SCOM 2012

  .
- `TelnetScomProbe.pl` - this script starts Telnet using the specified socket address of the probe.

Note : To monitor events, these scripts require the OMNIbus ObjectServer to start under Process Agent control.

To use these scripts, you must install and configure them on your host system. Use the following information to do this.

- "Enabling automatic acknowledgement of events" on page 22
- "Disabling automatic event acknowledgment" on page 23

If the probe is installed on a separate machine from the SCOM server, the automated event acknowledgement feature will require the time and timezone of both the SCOM server and the probe's server to be the same for it to work properly.

Enabling automatic acknowledgement of events

To enable automatic acknowledgement of events, you must install the acknowledgement scripts, specify the hostname for the external procedures file and specify the Telnet support script for each platform.

Installing the acknowledgement scripts on windows

To install the acknowledgement scripts, use the following steps:

1. Use this command to install each script:

   ```
   %NCHOME%\bin\redist\isql.exe -U username -P password -S server_name -i
   script_name
   ```

   Where *username* is the OMNIbus user name, *password* is the OMNIbus password, *server_name* is the name of the ObjectServer, and *script_name* is the name of the script you want to install.

   Note : The `update_scom_windows.sql` script must be installed before the `auto_acknowledge_trigger_windows.sql` script.

   For example:

   - `%NCHOME%\bin\redist\isql.exe -U root -P "" -S NCOMS -i update_scom_windows.sql`
   - `%NCHOME%\bin\redist\isql.exe -U root -P "" -S NCOMS -i auto_acknowledge_trigger_windows.sql`

2. Use the following command to register the `netcool_tivoli_socket.dll` library with the operating system:

   ```
   %SYSTEMROOT%\system32\regsvr32 %OMNIHOME%\probes
   \win32\netcool_tivoli_socket.dll
   ```

3. Specify the hostname of the machine on which the external procedures file is located.

   Note : This must be the same machine on which you are running the probe.

   The script contains the following property.

   - **HOST** '*host_name*'

   Where *host_name* is the system where the file is located, and must be set to enable the script to automatically send events to Microsoft SCOM 2012. To set this property, use the following steps:

   a. Open the `update_scom_windows.sql`. The default location is `%OMNIHOME%\probes\win32\`
   b. Set the '*host_name*' variable to the system on which the file is located. For example, `HOST 'nc2000.ibm.com'`

Installing the acknowledgement scripts on UNIX

Note : As the Probe for Microsoft SCOM 2012 is not supported in a UNIX environment, you must copy the required scripts from the default installation folder to `$OMNIHOME/probes/arch` where *arch* is the name of the operating system.

To install the `update_scom_unix.sql` and `auto_acknowledge_trigger_unix.sql` scripts on a UNIX operating system use the following steps.

1. Copy these scripts to `$OMNIHOME/probes/arch`
2. Use these commands to install each scripts.

   ```
   cat $OMNIHOME/probes/arch/script_name | $OMNIHOME/bin/nco_sql -user username -
   password password -server server_name
   ```

   Where *arch* is the operating system, *username* is the OMNIbus user name, *server_name* is the name of the ObjectServer, *password* is the OMNIbus password and *script_name* is the name of the script you want to install.

Note : The `update_scom_windows.sql` script must be installed before the `auto_acknowledge_trigger_windows.sql` script.

For example:

- `cat $OMNIHOME/probes/solaris2/update_scom_unix.sql | $OMNIHOME/bin/nco_sql -user root -password "" -server NCOMS`
- `cat $OMNIHOME/probes/solaris2/auto_acknowledge_trigger_unix.sql | $OMNIHOME/bin/nco_sql -user root -password "" -server NCOMS`

3. Specify the hostname of the machine where the external procedures file is located (`update_scom.sql`)

   Note : This is usually the same machine on which you are running the probe.

4. Specify the hostname of the machine on which the external procedures file is located.

   Note : This must be the same machine on which you are running the probe.

   The script contains the following property.

   - **HOST** '*host_name*'

   where *host_name* is the system where the file is located, and must be set to enable the script to automatically send events to Microsoft SCOM 2012. To set this property use the following steps.

   a. Open the `update_scom_unix.sql` file. The file should be copied to `%OMNIHOME%\probes\arch\`
   b. Set the '*host_name*' variable to the system on which the file is located. For example, `HOST 'nc2000.ibm.com'`

Note : There are a number of common issues which can cause problems when configuring automatic event acknowledgement. For more information, see "Known issues with automatic event acknowledgement" on page 41.

Disabling automatic event acknowledgment

Once the acknowledgment scripts are installed they are enabled by default. To disable the automatic acknowledgement of events use the **Netcool/OMNIbus Administrator** to configure whether these scripts are enabled or disabled.

To open the **Netcool/OMNIbus Administrator** use the following steps:

- On UNIX enter the following command: `$OMNIHOME/bin/nco_config`
- On Windows run the **Netcool Suite** and Select **Administrator**

When the **Netcool/OMNIbus Administrator** is open navigate to the event trigger using the following steps

1. Select the Probe for Microsoft SCOM 2012 Objectserver.
2. Enter the user details.
3. Expand the **Automation** menu.
4. Select **Triggers**
5. Right click on the trigger and select **Edit Trigger**
6. In the **State** field, check or uncheck the **Enable** button.
7. Click **OK**

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.

Timestamp formats

Some elements described in this section contain timestamps that are displayed in UNIX time (also known as POSIX time). UNIX time is a standard system of time notation, defined as the number of seconds elapsed since 00:00:00 Coordinated Universal Time (UTC) on January 01, 1970.

Some elements contain timestamps that are displayed as UTC. UTC times are given in the standard format specified by RFC 3339, as follows:

```
<date-fullyear>-<date-month>-<date-mday>T<time-hour>:<time-minute>:<time-second>.<time-secfrac>Z
```

The `<time-secfrac>` portion of the timestamp is given to seven digits.

For example: the timestamp `2011-04-12T23:20:50.5200000Z` represents 20 minutes and 50.5200000 seconds after the 23rd hour of April 12th, 2011 in UTC.

Elements generated by the probe

The following table describes the elements that the probe generates. Not all the elements described are generated for each event. The elements that the probe generates depend on the event type.

Table 7. Elements

Element name	Element description
`$connectorId`	This element contains the identifier of the connector that the probe registered with Microsoft SCOM 2012.
`$connectorStatus`	This element identifies the status of an alert in relation to the connector. This element takes the following values: `NotMarkedForForwarding` - the alert is not being managed by a connector. `Pending` - the alert is waiting to be forwarded. `SuccessfullyForwarded` - the alert has been successfully forwarded.
`$context`	This element contains the context of the alarm in XML format.
`$context_`*tagName*	This element contains the content of a field in the context of the alarm, where *tagName* is the name of the field. The field can contain a property name, a parameter index (such as `Param1`), or any other field name.
`$customField`*n*	This element contains data from a user-defined field, where *n* is an integer from 1 to 10 that represents one of ten available custom fields.
`$description`	This element shows the description of the alarm.

Table 7. Elements (continued)

Element name	Element description
`$displayString`	This element contains the string to display for the alarm.
`$id`	This element identifies the unique identifier of the event.
`$isMonitorAlert`	This element contains a Boolean value that indicates whether the alert was generated by a monitor.
`$lastModified`	This element shows the time (UTC) of the most recent update to the event.
`$lastModifiedBy`	This element contains the ID of the last user to modify the event.
`$lastModifiedByNonConnector`	This element shows the time (UTC) of the most recent update to the event done through the CLI.
`$lastModifiedByNonConnectorUTC`	This element shows the time (as UNIX time) of the most recent update to the alert done through the CLI.
`$lastModifiedUTC`	This element shows the time (as UNIX time) of the most recent update to the event.
`$maintenanceModeLastModified`	This element shows the time (UTC) at which the maintenance mode was last modified.
`$maintenanceModeLastModifiedUTC`	This element shows the time (as UNIX time) when the maintenance mode was last modified.
`$managementGroupId`	This element shows the identifier of the management group.
`$managementGroupName`	This element contains the name of the management group.
`$managementPackCategoryType`	This element indicates the category type of the management pack.
`$monitoringClassId`	This element contains the identifier of the monitoring class.
`$monitoringObjectDisplayName`	This element shows the name displayed for the monitoring object.
`$monitoringObjectFullName`	This element contains the full name of the monitoring object.

Table 7. Elements (continued)

Element name	Element description
`$monitoringObjectHealthState`	This element shows the health status of the monitoring object associated with an alert. This element takes the following values: `Error` - an error condition has occurred. `Success` - the object is in the correct operational state. `Uninitialized` - the object is in an un-initialized state. `Warning` - a warning condition has occurred.
`$monitoringObjectId`	This element contains the identifier of the monitoring object.
`$monitoringObjectInMaintenanceMode`	This element contains a Boolean value that identifies whether the monitoring object is in maintenance mode.
`$monitoringObjectName`	This element shows the name of the monitoring object.
`$monitoringObjectPath`	This element contains the directory path to the monitoring object.
`$monitoringRuleId`	This element contains the identifier of the rule set for the monitoring object.
`$name`	This element shows the name of the alert.
`$netbiosComputerName`	This element contains the NetBios name of the computer that raised the alert.
`$netbiosDomainName`	This element contains the domain name of the computer that raised the alert.
`$owner`	This element shows the User ID of the owner of the event. The User ID is usually a user account.
`$paramCount`	This element shows the total number of parameters for the alert.
`$param`*n*	This element shows a parameter of the alert, where *n* is the total number of parameters minus one (`$paramCount` – 1). For example, if there is a total of three parameters for the alert, then three separate elements are created: `$param0`, `$param1`, and `$param2`.
`$principalName`	This element shows the principal name of the computer that this alert was created for.
`$priority`	This element indicates the priority of an alarm as defined by Microsoft SCOM 2012. This element takes the following values: `High`, `Low`, `Normal`.

Table 7. Elements (continued)

Element name	Element description
`$problemId`	This element contains the identifier of the problem. If the value of `$isMonitorAlert` is `true`, `$problemId` is set to the globally unique identifier (GUID) of the monitor associated with the alert.
`$repeatCount`	This element shows the number of times this alert has occurred.
`$resolutionState`	This element identifies the resolution state of the alert. This element takes values in the range `0` to `255` inclusive.
`$resolvedBy`	This element shows the name of the user account responsible for resolving the alert. It appears when the alert is resolved.
`$severity`	This element indicates the severity of the alert as defined by Microsoft SCOM 2012. This element takes the following values: `Error` - the alert occurred because of an error. `Information` - the alert contains information about the system. `MatchMonitorHealth` - the alert severity matches the health state of the monitor that is associated with the alert. `Warning` - the alert contains a warning.
`$siteName`	This element shows the name of the site where Microsoft SCOM 2012 is installed, as given in the header of the alarm buffer display.
`$stateLastModified`	This element shows the time (UTC) at which the state of the alert was last modified.
`$stateLastModifiedUTC`	This element shows the time (as UNIX time) at which the state of the alert was last modified.
`$ticketID`	This element shows the identifier of the ticket in which the alert is described.
`$timeAdded`	This element contains the time (UTC) at which the alert was added to Microsoft SCOM 2012.
`$timeAddedUTC`	This element shows the time (as UNIX time) at which the alert was added to Microsoft SCOM 2012.
`$timeRaised`	This element shows the time (UTC) at which the alert was raised.
`$timeRaisedUTC`	This element shows the time (as UNIX time) at which the alert was raised.

Table 7. Elements (continued)

Element name	Element description
`$timeResolutionStateLastModified`	This element contains the time (UTC) at which the resolution state of the alert was last modified. Changes to the `ResolutionState` of the alert will cause this element to be updated.
`$timeResolutionStateLastModifiedUTC`	This element contains the time (as UNIX time) at which the resolution state of the alert was last modified. Changes to the `ResolutionState` of the alert will cause this element to be updated.
`$timeResolved`	This element contains the time (UTC) at which the alert was resolved.
`$timeResolvedUTC`	This element contains the time (as UNIX time) at which the alert was resolved.

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 8. Error messages

Error	Description	Action
`Failed to open listening socket, could not bind to port` *`port number`* `which is already in use by another application`	The command port cannot be opened because the value specified by the **`CommandPort`** property is already in use (possibly by another instance of the probe).	Specify another port number using the **`CommandPort`** property or close the application that is using the specified port.
`Failed to send probe watch:` *exception*	The probe is unable to send ProbeWatch messages to the ObjectServer.	Check the connection between the probe and the ObjectServer.
`Failed to accept new client connection, socket error code=` *ErrorCode*	The command port is unable to accept a new incoming connection from the client because of a socket error.	Check the detailed description of the *ErrorCode* that is available from the Microsoft MSDN Library. Go to http://msdn.microsoft.com/en-us/library and search for "Windows Sockets Error Codes".

Table 8. Error messages (continued)

Error	Description	Action
`[Command Port]` *hostname* `Failed to serve client. Reason:` *reason*	The command port is unable to continue processing requests from the client running on the host *hostname*.	Check the *reason* given in the error message for more details. This problem is sometimes caused by a forced disconnection on the client side.
`[Command Port]` *hostname* `Failed to write to client:` *client*. *exception*	An I/O problem is preventing the probe from sending processed results back to the command port client.	Check the *exception* given in the error message for more details. This problem is sometimes caused by a forced disconnection on the client side.
`Command port request of` *alert field entry* `cannot be processed and is thus ignored`	The probe is unable to set the requested alert field specified by the *alert field entry* argument sent to the command port.	Ensure that the argument is specified in the following format: `alertId(field=value)` For more information, see the description of the `set_field` command in "Command port module" on page 8.
`Unable to locate SCOM SDK assembly file` *sdk file* `in the folder specified by the ScomSdkDir property`	The probe failed to load the .NET libraries located in the folder specified by the **ScomSdkDir** property.	Verify that the **ScomSdkDir** property is correctly specified. For more information, see "Properties and command line options" on page 14. If this problem persists, contact IBM Software Support.
`Failed to process alerts update. Reason:` *reason*	The command port is unable to process a client request to update specified alert fields and submit the result to Microsoft SCOM 2012.	The client commands are in the proper format but there is another error, such as a non-updatable field being specified or the connection to Microsoft SCOM 2012 is down. Check the *reason* given in the error message for more details. Check in guide on updatable fields and also *reason* for more details. For more information about updating alarm fields in Microsoft SCOM 2012 using the command port, see "Command port module" on page 8.

Table 8. Error messages (continued)

Error	**Description**	**Action**
`Probe configuration could not be loaded, probe should not reconnect.`	There is an error in the properties file that is preventing the probe configuration from being loaded.	Check the properties file for syntax errors. Refer to "Properties and command line options" on page 14 and verify that the property values are specified correctly.
`Failed to process incoming alert with id` *`alert GUID`*`, alert is discarded`	The probe failed to parse the given alert retrieved from Microsoft SCOM 2012.	Microsoft SCOM 2012 might have sent the alert in an unexpected format. Contact IBM Software Support.
`Missing assembly file` *`dll file`* `in folder` *`app dir`*`. Please make sure probe package and dependencies packages are properly installed`	A library file required by probe is missing.	This error is probably caused by a missing or manually deleted file that the probe depends on. You might need to reinstall the probe.
`Unable to get events. Reason: Failed to connect to SCOM. Further Reason: Unable to resolve configured SCOM host name of` *`hostname`*`. Even Further Reason: The requested name is valid, but no data of the requested type was found`	The host name specified by the **`ScomHost`** property cannot be resolved to an IP address.	Verify that a valid host name has been specified for the **`ScomHost`** property. Use a DNS tool to verify that the value of the **`ScomHost`** property can be resolved to an IP address. If the value of the **`ScomHost`** property is correctly specified, verify that the DNS settings on the probe's host are set correctly.

Table 8. Error messages (continued)

Error	Description	Action
`Unable to get events. Reason: Failed to connect to SCOM. Further Reason: The client has been disconnected from the server. Please call ManagementGroup.Reconnect() to reestablish the connection. Even Further Reason: The socket connection was aborted. This could be caused by an error processing your message or a receive timeout being exceeded by the remote host, or an underlying network resource issue. Local socket timeout was '00:30:00'. Next Further Reason: An existing connection was forcibly closed by the remote host`	The connection between the probe and Microsoft SCOM 2012 was forcibly closed.	If this error occurs after the probe has just started, it is probably caused by an incorrect Windows user account. Ensure that the account has been provisioned in Active Directory. If this error occurs after the probe has started and run for some time, it is caused by the timeout of an idle connection between the probe and Microsoft SCOM 2012. Specify a shorter time period for the **`PollInterval`** property.
`Unable to get events. Reason: Failed to connect to SCOM. Further Reason: The user does not have sufficient permission to perform the operation. Even Further Reason: SOAP security negotiation with` *`scom_url`* `for target` *`scom_url`* `failed. See inner exception for more details. Next Further Reason: The Security Support Provider Interface (SSPI) negotiation failed.`	The probe failed to complete client authentication with Microsoft SCOM 2012.	The value specified for the **`ConnectorPassword`** property is incorrect. It does not match the correct password for the probe's user account. If you have encrypted the password, verify that the encrypted string starts and ends with an @ symbol. Verify that the generic Netcool/OMNIbus **`ConfigKeyFile`** property is correctly specified. For more information about encrypting property values, see the *IBM Tivoli Netcool/OMNIbus Administration Guide* (SC14-7605). For more information about generic Netcool/OMNIbus properties, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide* (SC14-7608).

Table 8. Error messages (continued)

Error	**Description**	**Action**
`Unable to get events. Reason: Failed to connect to SCOM. Further Reason: The user` *`user ID`* `does not have sufficient permission to perform the operation.`	The probe successfully completed client authentication but failed it authorize with Microsoft SCOM 2012.	The user account specified by the **`ConnectorUser`** property does not have sufficient permissions for the probe to continue operation. Verify that the user account's group has been assigned the `Operations Manager Administrator` user role in the Microsoft SCOM 2012 **Operations Console**.
`Unable to get events. Reason: Timed out in receipt of any event from SCOM`	The probe timed out due to inactivity on the connection to Microsoft SCOM 2012.	Change the value of the **`Timeout`** property to ensure that events get processed. For more information about specifying a value for this property, see "Inactivity" on page 8.
`Unable to get events. Reason: Error in probe configuration. Further Reason:` *`detail`*	There is an error in the properties file. The probe cannot proceed with its operations.	Check the *detail* given in the error message and check the property listed there.
`Unable to get events. Reason:` *`detail`*	The probe has encountered an unspecified error.	Check the *detail* given in the error message for more information.

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 9. ProbeWatch messages

ProbeWatch message	**Description**	**Triggers/causes**
`Received connection from` *`IP address`*	The command port (CLI) has received a connection.	A user logged onto the command port, specified by the **`CommandPort`** property, to send a request to Microsoft SCOM 2012.

Running the probe

The probe is run from the command prompt.

Before running the probe for the first time, you must specify a minimum set of properties as described in "Configuring the probe" on page 5.

To start the probe from the command prompt, use the following command:

```
%OMNIHOME%\probes\win32\nco_p_scom2012.bat
```

Note : The probe cannot be run as a Windows service. See "The probe cannot be run as a Windows service" on page 39 for an alternative method that uses a Netcool/OMNIbus process agent (PA) to manage the probe. Process agents can be run as Windows services.

Running multiple probes

You can run multiple instances of the probe on a single host machine at the same time.

You will need to do this if you want to monitor multiple, connected Microsoft SCOM 2012 management groups.

To run multiple instances of the probe, you must specify a unique value for the **ConnectorName** property in the properties file of each instance.

Additional probe tools

Several tools are supplied with the probe that enable you to acknowledge, resolve, and modify events in Microsoft SCOM 2012.

The additional tools supplied with the probe provide alternative methods for dealing with events in Microsoft SCOM 2012. They are not required for the normal operation of the probe.

The tools are described in the following sections:

- "Desktop and Webtop tools" on page 33
- "Installing the desktop and Webtop tools" on page 36
- "Using the desktop and Webtop tools" on page 37
- "Batch update tool" on page 37

Desktop and Webtop tools

The probe is supplied with tools that enable you to acknowledge and resolve events in Microsoft SCOM 2012 from the Windows desktop and from IBM Tivoli Netcool/Webtop.

Note : The desktop and Webtop tools require Perl 5.6 (or later) with the Telnet module installed, and a local copy of the Netcool/OMNIbus SQL interactive interface tool (`nco_sql`).

The following table describes the tools files supplied for use with UNIX and Linux® operating systems:

Table 10. UNIX and Linux desktop and Webtop tools files

User interface	File name	Description and operation
Desktop	`UnixScomTools.sql`	This file creates a UI menu called **Microsoft SCOM** that contains two menu items, **Acknowledge Alert** and **Resolve Alert**. Both menu items call `TelnetScomProbe.pl` and pass it the `@AlertKey` of the event and the socket address of the probe (as specified by `@NodeAlias`) as arguments.

Table 10. UNIX and Linux desktop and Webtop tools files (continued)

User interface	File name	Description and operation
	`TelnetScomProbe.pl`	This script is called by the **Acknowledge Alert** and **Resolve Alert** menu items. It starts Telnet using the specified socket address of the probe (`@NodeAlias`). It then connects to the probe's command port and issues either an `acknowledge_alarm` or `resolve_alarm` command using the `@AlertKey`. The **Acknowledge Alert** tool sets the event status to `Acknowledged` (1) in the `alert.status` ObjectServer table. The **Resolve Alert** tool sets the event severity to `Cleared` (0) in the `alert.status` ObjectServer table.
Webtop	`CreateScomGroup.xml`	This file creates a group called `Microsoft_SCOM` and adds the user account that you specify.
	`UnixScomTools.xml`	This file registers the `UnixScomTools.cgi` script tool in Webtop. This creates a UI menu called **Microsoft SCOM** that contains two menu items, **Acknowledge Alert** and **Resolve Alert**. Both menu items call `UnixScomTools.cgi` and pass it the `@AlertKey` of the event and the socket address of the probe (as specified by `@NodeAlias`) as arguments.
	`UnixScomTools.cgi`	This script is called by the **Acknowledge Alert** and **Resolve Alert** menu items. It starts Telnet using the Perl module, connects to the probe's command port, and then issues either an `acknowledge_alarm` or `resolve_alarm` command using the `@AlertKey` value. The **Acknowledge Alert** tool sets the event status to `Acknowledged` (1) in the `alert.status` ObjectServer table. The **Resolve Alert** tool sets the event severity to `Clear` (0) in the `alert.status` ObjectServer table.

The following table describes the tools files supplied for use with Windows operating systems:

Table 11. Windows desktop and Webtop tools files

User interface	File name	Description and operation
Desktop	`WindowsScomTools.sql`	This file creates a UI menu called **Microsoft SCOM** that contains two menu items, **Acknowledge Alert** and **Resolve Alert**. Both menu items call `TelnetScomProbe.VBS` and pass it the `@AlertKey` of the event and the socket address of the probe (as specified by `@NodeAlias`) as arguments.
	`TelnetScomProbe.VBS`	This file is called by the **Acknowledge Alert** and **Resolve Alert** menu items. It starts Telnet using the `netcool_tivoli_socket.dll` library. It then connects to the probe's command port and issues either an `acknowledge_alarm` or `resolve_alarm` command using the event identifier provided. The **Acknowledge Alert** tool sets the event status to `Acknowledged` (1) in the `alert.status` ObjectServer table. The **Resolve Alert** tool sets the event severity to `Cleared` (0) in the `alert.status` ObjectServer table.
	`netcool_tivoli_socket.dll`	This is a Win32 library that provides an implementation of the Telnet client for use on Windows.
Webtop	`CreateScomGroup.xml`	This file creates a group called `Microsoft_SCOM` and adds the user account that you specify.
	`WindowsScomTools.xml`	This file registers the `WindowsScomTools.cgi` script tool in Web GUI. This creates a UI menu called **Microsoft SCOM** that contains two menu items, **Acknowledge Alert** and **Resolve Alert**.
	`WindowsScomTools.cgi`	This script is called by the **Acknowledge Alert** and **Resolve Alert** menu items. It starts Telnet using the Perl module, connects to the probe's command port, and then issues either an `acknowledge_alarm` or `resolve_alarm` command using the event identifier provided. The **Acknowledge Alert** tool sets the event status to `Acknowledged` (1) in the `alert.status` ObjectServer table. The **Resolve Alert** tool sets the event severity to `Clear` (0) in the `alert.status` ObjectServer table.

Installing the desktop and Webtop tools

The desktop and Webtop tools are installed as described in the following sections.

Note : It is not necessary to have the desktop tools installed on the same host machine as the probe. You can manually copy the tools files described in "Desktop and Webtop tools" on page 33 from the installation package to any machine running Windows, UNIX, or Linux, and install them as described below.

Installing the desktop tools on Windows operating systems

The following procedures must be performed on Windows machines that are running an Event List.

To install the desktop tools, use the following steps:

1. Extract the desktop tool files from the probe installation package to the following directory on each target machine:

   ```
   %OMNIHOME%\probes\win32
   ```

   This location is hardcoded in the `WindowsScomTools.sql` file. If you want to use a different directory, you must change the hardcoded path in `WindowsScomTools.sql`.
2. On each machine from which you want to run the tools, register the `netcool_tivoli_socket.dll` library with the operating system using the following command:

   ```
   %SYSTEMROOT%\system32\regsvr32 %OMNIHOME%\probes
   \win32\netcool_tivoli_socket.dll
   ```
3. To install the tools, run the following command on one machine:

   ```
   %NCHOME%\bin\redist\isql.exe -U username -P password -S server_name -i
   WindowsSCOMTools.sql
   ```

Installing the desktop tools on UNIX and Linux operating systems

The following procedures must be performed on UNIX or Linux machines that are running an Event List.

To install the desktop tools, use the following steps:

1. Extract the desktop tool files from the probe installation package to the following directory on each target machine:

   ```
   $OMNIHOME/probes/solaris2
   ```

   This location is hardcoded in the `UnixScomTools.sql` file. If you want to use a different directory, you must change the hardcoded path in `UnixScomTools.sql`.
2. On each machine from which you want to run the tools, edit `TelnetSCOMProbe.pl` to include the directory path to your Perl interpreter.

   The default is `/usr/local/bin/perl`.

   Note : `TelnetSCOMProbe.pl` assumes that the Firefox web browser binary is included in the PATH environment variable.
3. To install the tools, run the following command on one machine:

   ```
   cat $OMNIHOME/probes/arch/UnixScomTools.sql | $OMNIHOME/bin/nco_sql -user
   username -password password -server object_server_name
   ```

Installing the Webtop tools on Windows operating systems

To install the Webtop tools, issue the following command from the `%OMNIHOME%\probes\win32` directory:

```
"%WEBTOP_HOME%\waapi\bin\runwaapi" -file WindowsSCOMTools.xml
```

Installing the Webtop tools on UNIX and Linux operating systems

To install the Webtop tools, use the following steps:

1. Edit `UnixSCOMTools.cgi` to include the following entries:
   - The directory path to your Perl interpreter.

     The default is `/usr/local/bin/perl`.
   - The `OMNIHOME` environment variable.

     The default is `$OMNIHOME="/space/delphine/71/omnibus"`.
   - The password of the Webtop user.

     The default is `$password=""`.
2. From the `$OMNIHOME/probes/`*`arch`* directory, run the following commands in the order given:
   a. `$WEBTOP_HOME/waapi/bin/runwaapi -file`
   b. `$OMNIHOME/probes/`*`arch`*`/UnixSCOMTools.xml chmod +x`
   c. `$WEBTOP_HOME/config/cgi-bin/UnixSCOMTools.cgi`

Using the desktop and Webtop tools

The **Acknowledge Alert** and **Resolve Alert** tools issue commands using the probe's command port.

The **Acknowledge Alert** tool connects to the probe's command port and issues the `acknowledge_alarm @AlertKey` command. Running the tool has the following results:

- The value of `@AlertKey` is normally set to the value of the `$id` token of the event, which is the identifier required to acknowledge alerts in Microsoft SCOM 2012.
- The resolution state of the event in Microsoft SCOM 2012 is set to 85 (Acknowledged).
- The Netcool/OMNIbus alert is set to `Acknowledged`.

The **Resolve Alert** tool connects to the probe's command port and issues the `resolve_alarm @AlertKey` command. Running the tool has the following results:

- The value of `@AlertKey` is set to the value of the `$id` token of the event, which is the identifier required to resolve alerts in Microsoft SCOM 2012.
- The resolution state of the event in Microsoft SCOM 2012 is set to 255 (Resolved).
- The severity of the Netcool/OMNIbus alert is set to `0` (clear).

Batch update tool

The probe is supplied with a batch update tool that enables you to modify events in Microsoft SCOM 2012.

Note : The batch update tool, `scom_tool.pl`, is only supported on UNIX and Linux operating systems and requires Perl 5.6 (or later).

The batch update tool takes a list of actions found in an ObjectServer table and passes them to the probe using the probe's command port. The probe can then modify events in Microsoft SCOM 2012. The tool can be used with two probe instances running in a failover configuration.

For installation and configuration details, see the `README.scom_tool` file supplied with the probe.

The following table describes the batch update tool files supplied with the probe:

Table 12. Batch update tool files

File name	Description
`README.scom_tool`	This file contains setup and operating instructions, known issues, and requirements information about the batch update tool.

Table 12. Batch update tool files (continued)

File name	Description
`scom_tool.conf`	This is the configuration file for the batch update tool.
`scom_tool.pl`	This is the batch update Perl script.
`scom_tool.sql`	Example SQL for enabling the ObjectServer to support the batch update tool.

Troubleshooting

Various issues arise as users work with the probe. This troubleshooting information is provided to help you diagnose and resolve such issues.

This section covers the following troubleshooting topic:

- "Connector registration issues" on page 38

Connector registration issues

Connectors that have lost their registration IDs can be manually retrieved and removed.

Retrieving a connector with a lost registration ID

To retrieve a connector whose registration ID is lost, use the following steps:

1. Create a text file with the name `getConnector.txt` in the `C:\` directory.
2. Add the following commands to `getConnector.txt`:

```
use OperationsManager
Select * from Connector
```

3. Save the text file.
4. To retrieve a list of available connectors, run the following command from the command prompt:

   `C:\>Sqlcmd -i C:\getConnector.txt`

The command output will show a list of all available connectors, with their registration IDs.

Removing a connector with a lost registration ID

Note : You can also remove a connector using the Microsoft SCOM 2012 GUI. Connectors can be removed only when they are not initialized or subscribed to.

To find and remove a connector whose registration ID is lost, use the following steps:

1. Create a text file with the name`removeConnector.txt` in the `C:\` directory.
2. Add the following commands to `removeConnector.txt`:

```
use OperationsManager
execute [dbo].[p_ConnectorDelete] 'Connector_ID'
```

   where *Connector_ID* is the identifier of the connector with the lost registration ID.

   For example: `3C4111C1-E3E5-4415-B3CF-7A61056F5EF2`
3. To remove the connector, run the following command from the command prompt:

   `C:\>Sqlcmd -i C:\removeConnector.txt`

Known issues

At the time of release, several known issues were reported that you should be aware of when running the probe.

This section covers the following known issues:

- "The probe cannot be run as a Windows service" on page 39
- "Tally mismatch for repeat alerts" on page 40
- "Connector name subscriptions" on page 40
- "AntiXssLibrary error" on page 41
- "East Asian locale settings" on page 41
- "Summary field gets truncated" on page 42
- "Missing assembly file message" on page 42
- "Unable to install the SCOM 2007 R2 and SCOM 2012 probe on the same system" on page 42

The probe cannot be run as a Windows service

The probe cannot be run as a Windows service. As an alternative, you can use a Netcool/OMNIbus process agent (PA) to manage the probe. Process agents can be run as Windows services.

If you use a PA to manage the probe, you must include the current working directory (CWD) of the probe executable in the PA configuration file (`nco_pa.conf`). In this case, the startup batch file is not used to start the probe.

The default location of the PA configuration file is `%NCHOME%\omnibus\etc`.

Use the following steps to configure and run the PA:

1. Edit `nco_pa.conf` to include the following directory path entry:

   ```
   Command
   '[CWD=%OMNIHOME%\probes\win32]%OMNIHOME%\probes\win32\
   nco_p_nonnative.exe nco_p_scom2012.exe' run as 0.
   ```

   Note : The CWD entry must be contained within square brackets and must not contain any spaces.
2. If you are running the ObjectServer under PA control along with the probe, ensure that it is stopped before performing the next step.
3. To start the PA, use the following command at the command prompt on the host machine:

   `%OMNIHOME%\bin\nco_pad -name` *`process_agent`* `-authenticate WINDOWS`

   where *process_agent* is the name of the PA as defined in the Netcool/OMNIbus Server Editor file (`%NCHOME%\ini\sql.ini`).

You can monitor the status of the PA and shut it down from the command line. Using the following commands will require you to enter your Windows user account password:

- To display the service status of the PA, use the following command:

  `%OMNIHOME%\bin\nco_pa_status -server` *`process_agent`* `-user` *`username`*
- To shutdown the PA, use the following command:

  `%OMNIHOME%\bin\nco_pa_shutdown -server` *`process_agent`* `-user` *`username`*

For more information about using process agents, see the *IBM Tivoli Netcool/OMNIbus Administration Guide* (SC14-7605).

For more information about installing and configuring process agents as Windows services, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide* (SC14-7604).

Example PA configuration file

The following is an example PA configuration file that runs the ObjectServer and the probe:

```
# List of processes
#
nco_process 'MasterObjectServer'
{
Command '%OMNIHOME%\bin\nco_objserv -name NCOMS -pa NCO_PA' run as 0
Host = 'host_ip'
Managed = True
RestartMsg = '${NAME} running as ${EUID} has been restored on ${HOST}.'
AlertMsg = '${NAME} running as ${EUID} has died on ${HOST}.'
RetryCount = 0
ProcessType = PaPA_AWARE
}
nco_process 'SCOM2012Probe'
{
Command '[CWD=%OMNIHOME%\probes\win32]%OMNIHOME%\probes\
win32\nco_p_nonnative.exe nco_p_scom2012.exe' run as 0
Host = 'host_ip'
Managed = True
RestartMsg = '${NAME} running as ${EUID} has been restored on ${HOST}.'
AlertMsg = '${NAME} running as ${EUID} has died on ${HOST}.'
RetryCount = 0
ProcessType = PaPA_AWARE
}
# List of Services
#
nco_service 'Core'
{
ServiceType = Master
ServiceStart = Auto
process 'MasterObjectServer' NONE
process 'SCOM2012Probe' 'MasterObjectServer'
}
# ROUTING TABLE
#
IBM Tivoli Netcool/OMNIbus Probe for SCOM 2012
nco_routing
{
host 'host_ip' 'NCO_PA' 'user' 'password'
```

Tally mismatch for repeat alerts

The ObjectServer uses the `Tally` field to keep track of the number of times that an alert occurs. This enables it to deduplicate repeated alerts in the Netcool/OMNIbus active event list but still track the number of times a particular alert has occurred.

The probe uses the `RepeatCount` field in Microsoft SCOM 2012 as the basis for the `Tally` field. It calculates the value of the `Tally` field as `RepeatCount` + 1.

Due to a known limitation, the first time that an alert is received by the ObjectServer its `Tally` value is set to 1 regardless of the value of `RepeatCount`. This is normally not a problem. However, in cases where the probe has shut down for a period of time (or is otherwise unavailable), any new alerts that it receives on startup will be assigned a `Tally` value of 1 by the ObjectServer, regardless of the value of the `RepeatCount` field in Microsoft SCOM 2012. This results in a mismatch between the `Tally` field in the ObjectServer and the `RepeatCount` field in Microsoft SCOM 2012, but only for alerts that the probe has not previously processed.

Connector name subscriptions

With the Probe for Microsoft SCOM 2007 and Probe for Microsoft SCOM 2007 R2 it was possible to create multiple connectors with duplicate settings. This is not possible with the Probe for Microsoft SCOM 2012.

You must ensure that each unique **`ConnectorName`** does not contain duplicated subscription settings, otherwise the probe will fail to receive events.

Known issues with automatic event acknowledgement

There are a number of issues which can occur when configuring automatic event acknowledgement. To resolve some of the more common issues, use the below information.

Known issues on windows and UNIX

If events are acknowledged in OMNIbus but Microsoft SCOM 2012 is not being updated by the external procedure file, check that the following files are configured correctly.

- `NCO_PA.conf` The scripts require the ObjectServer to start under process agent control. Ensure that the PA configuration file is correctly configured.
- `NCOMS.props` Ensure the **`PA.Username`** and **`PA.Password`** properties are set to the username and password set in the `NCO_PA.conf` file to start the process agent.

Known issues on UNIX only

If events are acknowledged in OMNIbus but Microsoft SCOM 2012 is not being updated by the external procedure file, there is a possibility that the `perl` commands used in the `TelnetScomProbe.pl` script are not in the default location, `/usr/local/bin/perl`. To check this use the `which perl` command. For example;

-
  ```
  $which perl
  /usr/bin/perl
  ```

  If the command returns a different `perl` location, open the `TelnetScomProbe.pl` file and change the first line of the script to match the location of your `perl` installation. For example, `#!/usr/bin/perl`

AntiXssLibrary error

When an alert is triggered in Microsoft SCOM 2012 by shutting down a monitored windows service the probe will receive the following error:

```
Error: E-CSP-000-000: Failed to process incoming alert with id {c6095dcd-
f1d0-4cf0-bed8-95d0354e8e2d}, alert is discarded. Reason: The type initializer
for 'Microsoft.EnterpriseManagement.Mom.InternalSdkOnly.DescriptionHelper'
threw an exception. Further Reason: Could not load file or assembly
'AntiXSSLibrary' or one of its dependencies. An argument was out of its legal
range. (Exception from HRESULT: 0x80131502). Even Further Reason: Length cannot
be less than zero.
```

To configure the probe to handle this error use the following steps:

1. Copy the `AntiXssLibrary.dll` file from the Microsoft SCOM 2012 installation directory: `C:\Program Files\System Center 2012\Operations Manager\Console` to `%OMNIHOME%\probes\win32`
2. Restart the probe.

East Asian locale settings

A defect in Microsoft .NET Framework 3.5 causes problems with event processing when the probe is using East Asian character encoding and the volume of event data reaches 2GB.

If you are using East Asian locale settings (including Japanese, Chinese, and Korean), you must either enable UTF-8 mode in the probe or upgrade your .NET Framework installation on the probe's host machine to version 4.0.

For more information about multicultural support in Netcool/OMNIbus, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide*.

Using UTF-8 encoding

To enable UTF-8 mode in the probe, set the generic `-utf8enabled` command line option to TRUE.

Note : If you enable UTF-8 mode, you must ensure that the probe configuration files are UTF-8 encoded.

Summary field gets truncated

Summary field values that exceed 255 characters are truncated.

The probe uses the rules file to create the value of the ObjectServer `Summary` field by concatenating the name of the alert ($name) and its description (`$description` or `$context_EventDescription`). The `Summary` field can accept values up to 255 characters long. Concatenated values that exceed 255 characters are truncated.

Missing assembly file message

The following message may occur in the probe log when running on the Windows 2012 operating system:

```
Missing assembly file "System.IdentityModel.Selectors.dll" in folder
```

This has no functionality impact on the probe. The Probe for Microsoft SCOM 2012 requires only the following three DLLs which come with the target system:

- `Microsoft.EnterpriseManagement.Core.dll`
- `Microsoft.EnterpriseManagement.OperationsManager.dll`
- `Microsoft.EnterpriseManagement.Runtime.dll`

Unable to install the SCOM 2007 R2 and SCOM 2012 probe on the same system

Currently it is not possible to install both the Probe for Microsoft SCOM 2007 R2 and the Probe for Microsoft SCOM 2012 on the same system.

This is because both probes use the same set of SCOM tool scripting files, namely:

```
tools/README.scom_tool
tools/scom_tool.pl
tools/scom_tool.sql
tools/scom_tool.conf
tools/CreateScomGroup.xml
tools/win32/TelnetScomProbe.VBS
tools/win32/WindowsScomTools.sql
tools/win32/WindowsScomTools.xml
tools/win32/WindowsScomTools.cgi
tools/win32/auto_acknowledge_trigger_windows.sql
tools/win32/update_scom_windows.sql
tools/win32/netcool_tiv_oli_socket/netcool_tivoli_socket.dll
tools/unix/TelnetScomProbe.pl
tools/unix/UnixScomTools.sql
tools/unix/UnixScomTools.xml
tools/unix/UnixScomTools.cgi
tools/unix/auto_acknowledge_trigger_unix.sql
tools/unix/update_scom_unix.sql
```

To work around this issue, after you have installed one probe (for example, the Microsoft SCOM 2007 R2 Probe) you need to backup and remove the duplicate files listed above before installing the other probe (for example, the Microsoft SCOM 2012 Probe).

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 want 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, zSeries, 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.

SC22-5475-05