IBM® Tivoli® Netcool/OMNIbus Probe for Microsoft Windows Event Log
4.0

Reference Guide
June 28, 2019

SC14-7648-06

Notice

Before using this information and the product it supports, read the information in Appendix A, “Notices and Trademarks,” on page 21.

Edition notice

This edition (SC14-7648-06) applies to version 4.0 of IBM Tivoli Netcool/OMNIbus Probe for Microsoft Windows Event Log and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC14-7648-05.

© Copyright International Business Machines Corporation 2006, 2019.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this guide

The following sections contain important information about using this guide.

Document control page

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

The IBM Tivoli Netcool/OMNIbus Probe for Microsoft Windows Event Log documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM® Tivoli® Information Center:

http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.tivoli.namomnibus.doc/welcome_ptsm.htm

Table 1. Document modification history

Document version	Publication date	Comments
SC14-7648-00	February 25, 2011	First IBM publication.
SC14-7648-01	April 6, 2012	"Summary" on page 1 updated. "East Asian locale settings" on page 3 added. "Running the probe" on page 3 updated. "Known issues" on page 17 added.
SC14-7648-02	August 3, 2012	"Summary" on page 1 updated.
SC14-7648-03	August 31, 2012	"Summary" on page 1 updated. New known issue added to "Known issues" on page 17 .
SC14-7648-04	March 14, 2013	"Forwarded event log" on page 10 updated. "Properties and command line options" on page 11 updated with the **SecondaryLocale** and **SourceType** properties. "Known issues" on page 17 updated.
SC14-7648-05	March 7, 2014	"Summary" on page 1 updated. "Known issues" on page 17 updated.
SC14-7648-06	June 28, 2019	"Summary" on page 1 updated for version 4 of the probe. Support for Windows Server 2019 Standard Edition Event Log added.

© Copyright IBM Corp. 2006, 2019

Conventions used in this guide

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

Operating system-dependent variables and paths

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

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

`$OMNIHOME/probes`

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

`%OMNIHOME%\probes`

For probes supported on UNIX, Linux, and Windows operating systems, probe guides use the standard UNIX conventions for specifying environment variables and describing directory paths. When using the Windows command line with these probes, replace the UNIX conventions used in the guide with Windows conventions. If you are using the bash shell on a Windows system, you can use the UNIX conventions.

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

Operating system-specific directory names

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

`$OMNIHOME/probes/`*`arch`*

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

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

Table 2. Directory names for the arch variable

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

OMNIHOME location

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

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

Chapter 1. Probe for Microsoft Windows Event Log

The Probe for Microsoft Windows Event Log monitors any event log files on Windows Server.

This guide contains the following sections:

Summary

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

The following table provides a summary of the Probe for Microsoft Windows Event Log.

Table 3. Summary

Probe target	Windows Server 2008 (32 bit or 64 bit) Event Log Windows Server 2012 Standard and Datacenter Editions Event Log Windows Server 2016 Standard and Datacenter Editions Event Log Windows Server 2019 Standard and Datacenter Editions Event Log
Probe executable name	`nco_p_wineventlog.bat`
Probe supported on	For details of supported operating systems, see the following Release Notice on the IBM Software Support Website: https://www-304.ibm.com/support/docview.wss?uid=swg21625800
Package Version	4.0
Properties file	`%OMNIHOME%\probes\win32\wineventlog.props`
Configuration file	`%OMNIHOME%\probes\win32\wineventlog.xml`
Rules file	`%OMNIHOME%\probes\win32\wineventlog.rules`

© Copyright IBM Corp. 2006, 2019

Table 3. Summary (continued)	
Requirements	For details of any additional software that this probe requires, refer to the `description.txt` file that is supplied in its download package.
Connection method	Windows API
Multicultural support	Available
Peer-to-peer failover functionality	Available
IP environment	IPv4
Federal Information Processing Standards (FIPS)	IBM Tivoli Netcool/OMNIbus uses the FIPS 140-2 approved cryptographic provider: IBM Crypto for C (ICC) certificate 384 for cryptography. This certificate is listed on the NIST website at http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2004.htm. For details about configuring Netcool/OMNIbus for FIPS 140-2 mode, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide*.

Installing probes

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

The installation process consists of the following steps:

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

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

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

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

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

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

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

Unicode Transformation Format (UTF-8) encoding is a variable length character encoding for Unicode. It can represent any character in the Unicode standard. This controls how the inputs to the probe (for example, the properties file, rules file, and event stream) are encoded and how the probe encodes its output (for example, log files and events). When `-utf8enabled` is set to TRUE, these inputs are UTF8 encoded.

This command option is only available for Netcool/OMNIbus version 7.3.0 and later, and it is a generic Netcool/OMNIbus property. Before V7.3.0, Netcool/OMNIbus only supported UTF-8 encoding on Linux and UNIX operating systems. Netcool/OMNIbus V7.3.0 also supports UTF-8 on Windows operating systems.

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

Running the probe

Currently you can only run the probe in Console Mode.

To run the probe in Console Mode, you must run it from the command line using the following command:

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

Note : Due to a known issue, the probe can only be started from the `%OMNIHOME%\probes\win32` directory. See "Probe can only be started from the %OMNIHOME%\probes\win32 directory" on page 18.

Running the probe under process agent control

The probe can be run under process agent (PA) control. You can control how the process agent interacts with the probe using the PA configuration file.

The default location is `%NCHOME%\omnibus\etc`.

Use the following steps to configure and run the process agent :

1. In the `nco_pa.conf` ensure the current working directory (CWD) is pointing to the path where the `welcsnsprobe.dll` file is located, for example:

   ```
   Command '[CWD=C:\IBM\Tivoli\Netcool\omnibus\probes\win32]%OMNIHOME%\probes
   \win32\nco_p_nonnative.exe nco_p_wineventlog.exe' run as 0.
   ```

   Note : The CWD must be specified using square brackets and without spaces.
2. Ensure the ObjectServer is not already started if you plan to run the ObjectServer under PA control.
3. Use the following command on the command line of the host to manually start a process agent:

   `%OMNIHOME%\bin\nco_pad -name` *`process_agent`*.

   Note : The *process_agent* variable is the name of the process agent as defined in the Netcool/OMNIbus Server Editor or `%NCHOME%\ini\sql.ini` file.
4. To display the service status of process agents you have configured, enter the following command:

`%OMNIHOME%\bin\nco_pa_status`

Note : You will need to enter your windows password.

The following is an example PA configuration file for running 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 'WineventlogProbe'
{
    Command '[CWD=C:\IBM\Tivoli\Netcool\omnibus\probes\win32]
    %OMNIHOME%\probes\win32\nco_p_nonnative.exe
    nco_p_wineventlog.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 'WineventlogProbe' 'MasterObjectServer'
}

# ROUTING TABLE
#
nco_routing
{
    host 'host_ip' 'NCO_PA' 'user' 'password'
}
```

Running multiple probes

You can run multiple instances of the probe.

The following outlines the minimum configuration requirements to run multiple probes:

1. Specify the **Name** property to a unique instance name, for example

   `wineventlog_probe1`
2. Create a new version of the properties file, for example

   `wineventlog_probe1.props`
3. Create a new version of the XML file, for example

   `wineventlog_probe1.xml`
4. Specify the **ConfigFile** property in the properties file to direct the instance to the unique XML file, for example

   `wineventlog_probe1.xml`
5. Change the **RuleFiles** property in the properties file to the default rules file, for example

   `C:\\IBM\\Tivoli\\Netcool\\omnibus\\probes\\win32\\wineventlog.rules`

Run each unique instance separately, using the following command, for example:

- Instance 1 (wineventlog_probe1):

  ```
  nco_p_wineventlog.bat -propsfile "C:\\IBM\\Tivoli\\Netcool\\omnibus\\probes\
  \win32\\wineventlog_probe1.props"
  ```

- Instance 2 (wineventlog_probe2):

  ```
  nco_p_wineventlog.bat -propsfile "C:\\IBM\\Tivoli\\Netcool\\omnibus\\probes\
  \win32\\wineventlog_probe2.props"
  ```

Compatibility with previous versions of the probe

When migrating from the Probe for Windows NT Event Log (nco_p_mhntlog) to the Probe for Windows Event Log (nco_p_wineventlog.bat) you will need to map the old element names with the new element names when making updates to any rules files or filter files that you may have previously configured.

The following table outlines the old element name and the corresponding the new element name:

Table 4. Element name mapping

Old Element Name	New Element Name
$EventCategory	$TaskCategory
$EventDescription	$EventDescription
$EventID	$EventID
$EventType	$Level
$HostName	$Computer
$IPAddress	Not Applicable
$LogFile	$LogName
$LogSource	$Source
$Machine	$Computer
$RecordNumber	$RecordID
$TimeGenerated	$DateTime - Includes both date and time information.
$UserName	$User

Requirements

A few prerequisites need to be set before running the probe.

- Microsoft .Net version 4.0 or later.
- Event Log service must be running.
- The user running the probe must have permission to access each specified Event Log.
- To monitor forwarded event logs the user must configure event forwarding on the windows operating system so that the machine on which this probe runs can read events in the forwarded event log folder.
- Both host machines need to be accessible to each other.
- Make sure that you can remotely access the host machines.

Note : If you make any network changes, you should restart the system.

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 reads an XML configuration file that contains a list of specific log files to monitor. The probe then reads the specified log files and sends the events to the ObjectServer.

Data acquisition is described in the following topics:

Buffer settings

The probe maintains a queue that stores raw events before they are processed by the probe. When an event storm occurs, this queue can grow quickly and consume excessive amounts of memory.

To increase the efficiency of sending alerts to the ObjectServer, the following properties are available:

- **`Buffering`** - When set to 1, this property instructs the probe to send alerts when the internal alert buffer has reached the size specified by the **`BufferSize`** property.
- **`BufferSize`** - This property specifies the size of the buffer that the probe uses to store alerts before sending them to the ObjectServer.

Example buffer settings

The following example shows performance settings from the properties file of a Probe for Windows Event Log:

```
BufferSize          : 100
Buffering           : 1
FlushBufferInterval : 10
```

When the internal alert buffer has 100 alerts waiting to be sent to the ObjectServer, or after 10 seconds have elapsed since the last flush, the probe flushes the alerts in the buffer to the ObjectServer.

Peer-to-peer failover functionality

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

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

Example property file settings for peer-to-peer failover

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

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

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

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

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

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

Event Synchronization

All events visible in the Event Viewer are stored in active event logs. The probe supports active events retrieval by resynchronizing the event logs on start up.

This resynchronization operation can be enabled by setting the <Resync> tag in the XML configuration file (wineventlog.xml). When the **Resync** property is enabled, the probe will retrieve active events every time a new connection is setup.

The probe can be configured, using the **EnableLastEventFilter** property in the wineventlog.props file, to store a marker for the last event received for each log and upon resynchronization it retrieves only the new events created since the last received event. If **EnableLastEventFilter** is set to 0 and the <Resync> tag in wineventlog.xml is set to true, the probe will retrieve all active events for the specific log.

The <ResyncFilter> tag in the XML configuration file needs to be specified for each log.

XML configuration file

The probe is supplied with an XML configuration file (wineventlog.xml) that allows you to specify from which of the Windows Server event logs the probe extracts events, and what filtering criteria the probe uses to limit the events that it receives.

The probe only monitors the events logs for which there is an entry in the XML configuration file. By default, the XML configuration file can be found in the following location:

%OMNIHOME%\probes\win32\wineventlog.xml

The XML configuration file must contain an entry for each event log that you want the probe to monitor. For each Windows log you want the probe to monitor, you must add a <Log> tag within the <WindowsLogs> tag of the XML file. For each application or services log you want the probe to monitor, you must add a <Log> tag within the <AppServiceLogs> tag of the XML file.

When the probe is running, it periodically checks whether the XML configuration file has been updated. This enables you to change the logs that the probe monitors without having to restart the probe. To specify the frequency with which the probe checks for updates to the XML configuration file, specify the **ReadFileInterval** property in the wineventlog.props file.

You can specify a different location for the host configuration file using the **ConfigFile** property in the wineventlog.props file.

Tags and attributes used by the XML configuration file

The following is a list of valid XML tags that can be used in the `wineventlog.xml` configuration file.

Table 5. Supported XML tags

Tag	Description	Attributes	Available sub-tags

`<WindowsLogs> </WindowsLogs>`	This is the top level XML tag for Windows logs.	None	`<log>`
			`<Resync>`
			`<ResyncFilter>`
			`<NotificationFilter>`

`<Log> </Log>`	This defines a log to monitor. The Name attribute specifies the full log name.
	Note : The name can be found in the Log Properties dialog box in Event Viewer, and must be entered into the XML configuration file exactly as it appears there.
		Name	`<Resync>`
			`<ResyncFilter>`
			`<NotificationFilter>`

`<Resync> </Resync>`	This is the XML tag which specifies resynchronization. If this is set to true, the probe will perform a resynchronization at every reconnection. The probe will also perform a resynchronization if you change the value set for this tag. So if it is initially set to `false` and later changed, then the probe will perform a resynchronization.	None	`<ResyncFilter>`
`<ResyncFilter> </ResyncFilter>`	This defines the XPath filter for resynchronization. The filter can be created manually or it can be configured through Filter Current Log dialog box in Windows Event Viewer.	None	None
`<NotificationFilter> </NotificationFilter>`	This defines the XPath filter for listening for new events. The filter can be created manually or it can be configured through Filter Current Log dialog box in Windows Event Viewer.	None	`<ResyncFilter>`

`<AppServiceLogs> </AppServiceLogs>`	This is the top level XML tag for application or service logs.	None	`<log>`
			`<Resync>`
			`<ResyncFilter>`
			`<NotificationFilter>`

XML configuration file syntax

The `wineventlog.xsd` file defines the expected structure and syntax of the XML configuration file. The schema is part of the XML technology used to define the xml file constraints. The user can refer to the `wineventlog.xsd` file for full constraint definitions. The default location is: `%OMNIHOME%\probes\win32\wineventlog.xsd`

The XML file uses the following syntax:

```
<WindowsLogs>
  <Log Name="windows_log_1">
        <Resync>true | false</Resync>
          <ResyncFilter>resynch_filter</ResyncFilter>
        <NotificationFilter>notification_filter</NotificationFilter>
  </Log>

  <Log Name="windows_log_2">
        <Resync>true | false</Resync>
          <ResyncFilter>resynch_filter</ResyncFilter>
        <NotificationFilter>notification_filter</NotificationFilter>
  </Log>
 </WindowsLogs>

<AppServiceLogs>
 <Log Name="app-service_log_1">
    <Resync>true | false </Resync>
    <ResyncFilter>resynch_filter</ResyncFilter>
    <NotificationFilter>Notification_filter</NotificationFilter>
 </Log>

 <Log Name="app-service_log_2">
    <Resync>true | false </Resync>
    <ResyncFilter>resynch_filter</ResyncFilter>
    <NotificationFilter>Notification_filter</NotificationFilter>
 </Log>
</AppServiceLogs>
```

Using the SourceType property

You can use the **SourceType** property to define the libraries that the probe uses when reading data from an event log. If you change the value of this property you need to move bookmark files to a new location to avoid a file conflict.

For each event log that you define in the `wineventlog.xml` configuration file, the probe creates a bookmark file named `wineventlog.bm`. This file is located in the following folder:

`%OMNIHOME%\win32\`*`logtype`*`\`

where *logtype* is the name of the type of event log.

For example, if you have configured the Application, Security, and Setup logs in the configuration file, a file named `wineventlog.bm` is created in each of the following folders:

`%OMNIHOME%\win32\Application\`
`%OMNIHOME%\win32\Security\`
`%OMNIHOME%\win32\Setup\`

If you change the value of the **SourceType** property, you need to move these files to a different location to avoid a file conflict occurring. Do this before changing the value of the property:

1. In the `%OMNIHOME%\win32\` folder create a folder named after the current setting of the **SourceType** property. For example, if the current value of the property is `DotNet`, create the following folder:

   `%OMNIHOME%\win32\DotNet`
2. Move the folders that contain the bookmark files into this new directory.

For example, move the %OMNIHOME%\win32\Application folder to %OMNIHOME%\win32\Dotnet\Application.

3. You can now change the value of the **SourceType** property.

Filter definitions

The resynchronization filter and notification filter allow you to maximize the efficiency of the probe. The filters can be created manually using XML Path Language (Xpath) expressions or they can be configured using the Filter Current Log dialog box in Windows Event Viewer. For information about Xpath syntax, visit the following Web site: http://msdn.microsoft.com/

Forwarded event log

Windows allows events to be forwarded from one host to another and by default, the forwarded event will be stored in the **Windows Logs** > **Forwarded Events** folder but a different folder can be specified.

You can configure the probe to monitor the forwarded events stored in any forwarded event log folder. Therefore, this probe indirectly supports remote event extraction by allowing you to retrieve events from the Forwarded Events folder.

The following configuration example explains how to set up Windows to forward from one host to another. The events forwarded from the source host (Host B) are sent to the collector host (Host A). A subscription is then configured on Host A that allows you to collect the forwarded events.

Set up event forwarding from Host B

First set up event forwarding from Host B:

1. Log on to Host B.
2. Open a command prompt and run the following command:

   ```
   winrm quickconfig
   ```

3. When prompted to make changes to the WinRM listener and Windows Firewall, enter Y.

   You will receive a confirmation that these changes were successful.

Set up event collection on Host A

On the collector host (Host A), you need to set up event collection. To do this, enable and start the collector service on the central server.

1. Log on to Host A.
2. Open a command prompt and run the following command:

   ```
   wecutil qc
   ```

3. When prompted to change the service startup mode, choose Yes.

   You should see a confirmation that the collector service was set up properly.

Create subscriptions to events on Host A

After setting up event forwarding and collection, you need to create subscriptions for the events that you wish to forward to the collector. Subscriptions are set up on the collector host. To set up a subscription, perform the following steps:

1. Open Server Manager by selecting **Start** > **Administrative Tools** > **Server Manager**.
2. Expand the **Diagnostics** > **Event Viewer** nodes.
3. Right-click the **Subscriptions** node and choose **Create Subscription**.
4. In the Subscription Properties window, enter the text `All Critical and Warning Events` in the Subscription Name text box.

5. Choose the **Collector Initiated** option. This option instructs the collector to connect to the source computers to gather events.
6. Click the **Select Computers** button.
7. In the **Computers** window, click the **Add domain computers** button. Enter the name `Host B` and click the **OK** button. Click the **OK** button in the **Computers** window to return to the **Subscription Properties window**.
8. Click the **Select Events** button.
9. Select the **Critical** and **Warning** options and then choose all **Windows Logs**. Then click the **OK** button.
10. Click the **Advanced button** to open the **Advanced Subscription Settings window**.
11. Click **Machine Account**.
12. Enter a username and password with sufficient access to the event logs on the source computer. Then click the **OK** button.
13. Click the **OK** button two more times to close all windows.

    You will now see the subscription active.

Test the subscription

Finally, test the subscription by creating a warning event on Host B:

1. Log on to the source computer, Host B.
2. Open a command prompt and enter the following:

   ```
   EVENTCREATE /T Warning /ID 500 /L Application /D “Testing Subscription”
   ```
3. Log on to Host A and open **Server Manager**.
4. Click on the **Diagnostics** > **Event Viewer** > **Windows Logs** > **Forwarded Events** node.

   The warning event you created on Host B should be displayed in the Forwarded Events log on Host A. You may need to click the **Refresh** button if the event does not appear. There is a short delay between the time an event is logged on a local computer and the time it is forwarded to the collector server.

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 information about default 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**
Buffer *integer*	`-buffer` (This is equivalent to **Buffer** with a value of 1.) `-nobuffer` (This is equivalent to **Buffer** with a value of 0.)	Use this property to specify whether buffering is used when sending events to the ObjectServer. This property takes the following values: 0: The probe does not use event buffering. 1: The probe buffers events before sending them to the ObjectServer. The default is 0. If multithreaded processing is in operation (the default), a separate communication thread is used to send data to each registered target ObjectServer, and a separate text buffer is therefore maintained for each ObjectServer. **Note :** All events sent to the same table are sent in the order in which they were processed by the probe. If alerts are sent to multiple tables, the order is preserved for each table, but not across tables.
BufferSize *integer*	`-buffersize` *integer*	Use this property to specify the number of events that the probe buffers before sending them to the ObjectServer. The default is 10.
ConfigFile *string*	`-configfile` *string*	Use this property to specify the location of the XML configuration file which defines the logs that the probe monitors, and the filters that the probe uses to limit which events it receives. The default is %OMNIHOME%\probes \win32\wineventlog.xml
EnableLastEventFilter *integer*	`-enableLastEventFilter` *integer*	Use this property to enable the last event filter which stores the last event generated so that when the probe restarts it will resynch the event list only as far back as the last stored event. This property takes the following values: 0: The probe does not use the last event filter. 1: The probe uses the last event filter. The default is 1.

Table 6. Properties and command line options (continued)		
Property name	**Command line option**	**Description**
EnableDetailEventDesc *integer*	`-enabledetaileventdesc` *integer*	Use this property to specify whether the probe extracts the full event message, or just the first line of the message. This property takes the following values: `0`: The probe only extracts the first line of the event. 1: The probe extracts the full event description The default is `0`. **Note :** A typical Windows event consists of multiple lines. The first line contains the event ID. When this property is set to 0, the probe attempts to find a period followed by space `". "` as an indicator of the first line. If this terminator is not found, the probe extracts the full description. The terminator is based on observation of the Event Message format in Event Viewer.
EventAttribute *string*	`-eventattribute` *string*	Use this property to specify a comma-separated list of additional attributes that the probe generates for all logs that it monitors. The default is `""`. For a list of the attributes that you can specify using this property, see: "Event Attributes" on page 14
FlushBufferInterval *integer*	`-flushbufferinterval` *integer*	Use this property to specify the interval (in seconds) that the probe waits before flushing the buffer contents to the ObjectServer. The default is `0`.
LastEventMarkerPath *string*	`-lasteventmarkerpath` *string*	Use this property to specify the location of the last event marker file. The default is `%OMNIHOME%\probes\win32`.
ReadFileInterval *integer*	`-readFileInterval` *integer*	Use this property to specify how frequently (in seconds) the probe checks for changes to the XML configuration file specified by the **ConfigFile** property. If the file has changed the probe reloads it. The default is 3.

Table 6. Properties and command line options (continued)		
Property name	**Command line option**	**Description**
Retry *integer*	`-retry` *integer*	Use this property to specify whether or not the probe attempts to reconnect automatically after an error. This property takes the following values: `0`: The probe does not attempt to reconnect automatically. `1`: The probe attempts to reconnect automatically. The default is `1`.
SecondaryLocale *string*	`-secondarylocale` *string*	Use this property to specify a secondary locale to use when processing the `EventDescription` entry in an event. If the entry is not populated using the system locale, the probe uses this locale to try and populate the entry. The default is `en-US`.
SourceType *string*	`-sourcetype` *string*	Use this property to specify the Microsoft library to use when reading data from the event log. This property takes the following values: `DotNet`: Use the Windows .NET libraries. `Native`: Use the Windows native library, `wevtapi.dll`. The default is `DotNet`. This property can help to avoid an issue where the `Event Description` entry is null on East Asian locales when using the DotNet method. In this case, change the value of this property to `Native`.

Event Attributes

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

The probe generates a standard set of event attributes for all logs that it monitors. By default, the probe generates the event attributes described in the following table:

Table 7. Default event attributes	
Event attribute name	**Event attribute description**
`$Computer`	This event attribute contains the name of the host.

Table 7. Default event attributes (continued)	
Event attribute name	**Event attribute description**
$DateTime	This event attribute displays the time when the event log record was generated.
$EventDescription	This event attribute contains the description of the event.
$EventID	This event attribute contains the unique identifier of the event.
$Level	This event attribute displays the category of the event.
$LogName	This event attribute contains the name of the log where the event was recorded.
$Source	This event attribute contains the name of the software that recorded the event.

The probe can also generate additional event attributes for all logs that it monitors by using the **EventAttribute** property. Using this property, you can specify one or more of the event attributes described in the following table:

Table 8. Additional event attributes	
Event attribute name	**Event attribute description**
$Keywords	This event attribute contains the keywords assigned to an event.
$OpCode	This event attribute contains the numeric value that identifies the activity or point within an activity that the application was performing when the event was raised.
$RecordId	This event attribute contains the number of the event log record.
$TaskCategory	This event attribute contains the subcomponent or activity of the event publisher.
$User	This event attribute displays the username of the owner of the event.

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 error messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 9. Error messages		
Error	**Description**	**Action**
`Host file parsing exception`	The probe failed to parse the configuration file. There could be a syntax error.	Review additional message in the log file.
`--- Corrupted refresh state :`	The probe encountered an error when refreshing the changes made in the configuration file.	Try rerunning the probe or Contact IBM Software Support.
`Invalid log configuration :` *`filename`*	The probe encountered an error in the log configuration file during refresh. This log file will be ignored.	Try recreating the log file.
`Exception during resync`	The probe encountered an error during resynchronization. It may not complete the resynchronization operation.	Try rerunning the probe or Contact IBM Software Support.

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 raw ProbeWatch error messages that the probe generates. For information about generic ProbeWatch messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 10. ProbeWatch messages		
ProbeWatch message	**Description**	**Triggers/causes**
`Start monitoring configuration file.`	The probe started to monitor configuration file changes.	The probe started.
`Stop monitoring configuration file.`	The probe stopped monitoring.	The probe stopped.
`Config file has changed, need refresh :`*`configFile`*	The probe detected configuration file changes.	The probe detected that one of the configuration files has changed.
`Start log configuration refresh`	The probe started a real time configuration refresh.	The probe detected that a log file changed and is executing a real time refresh..
`Remove event log monitoring :` *`logfile`*	You should remove the displayed log from the configuration file.	The probe has detected an error in the log file.
`Finished log configuration refresh`	The probe completed processing the changes in the configuration file.	The probe detected that one of the configuration files changed and it executed a refresh.

Table 10. ProbeWatch messages (continued)		
ProbeWatch message	**Description**	**Triggers/causes**
`Start listening for event from : logfile`	The probe starts to monitor for new events in the specified log.	The probe started or a new log had been detected.
`Fail to listen for event from: logfile`	The probe failed to monitor events in the specified log.	This could be a system error. Try running a separate probe to troubleshoot this log.
`Start to resync : logfile`	The probe started a resynchronization operation.	The probe started or a new log had been detected.
`Finished resync : logfile`	The probe completed a resynchronization operation.	The probe started or a new log had been detected.
`Fail to resync : logfile`	The probe resynchronization operation failed.	The probe was unable to resync. Check if there is any event in EventViewer to resync.

Known issues

At the time of release, various issues have been reported that you should be aware of when running the probe.

This section covers the following known issues:

Probe cannot run as a Windows service

The probe is unable to be run as a Windows Service. You must run the probe under process agent (PA) control instead.

For details see "Running the probe under process agent control" on page 3.

Defect in .NET 3.5 causes problems in East Asian locales when receiving high event data volumes

A defect in .NET 3.5 causes the probe to operate incorrectly when running in an East Asian locale and when event data volume reaches 2GB.

Users in East Asian locales must either enable UTF-8 mode in this probe, or upgrade the version of .NET Framework on the host machine running the probe to 4.0. See "East Asian locale settings" on page 3.

Running the probe on a server on which Microsoft .Net Framework 4.0 is installed

If you are running the probe on a server on which Microsoft .Net Framework 4.0 is installed, the `EventDescription` field of events received by the probe might not be populated.

This is due to a known issue within Microsoft .Net Framework caused by a difference between .Net 3.5 and .Net 4.0 in the way in which rendering is performed. If the event provider has not provided an event log description in its system locale, the `EventDescription` field is not populated.

From version 3.0 of the Probe for Windows Event Log, the probe has additional properties to help avoid this issue:

- **`SecondaryLocale`**
- **`SourceType`**

You can use either or both of these properties together.

SecondaryLocale

If the `EventDescription` field of a received event cannot be retrieved, the probe uses the locale defined in this property to attempt to retrieve the description. If the probe successfully retrieves an event description, it continues processing the event in the normal way. If the field is still not populated, the probe generates a warning message.

SourceType

You can use this property to instruct the probe to use native libraries when extracting data for the `EventDescription` field. These libraries help to avoid this issue as they process data differently from .Net 4.0. To use the native libraries, set the value of **`SourceType`** to `Native`.

Probe can only be started from the %OMNIHOME%\probes\win32 directory

The Probe for Windows Event Log can only be started from the `%OMNIHOME%\probes\win32` directory using the console command-line option.

If you attempt to start the Probe for Windows Event Log from any other directory, the probe writes a message similar to the following example to the error log:

```
12/12/2011 3:00:34: Debug: D-CSP-000-000: Parsing Exception :
System.IO.FileNotFoundException: Could not find file 'C:\U
sers\Administrator\wineventlog.xsd'.

File name: 'C:\Users\Administrator\wineventlog.xsd' at
System.IO.__Error.WinIOError(Int32 errorCode, String maybeFullPath)

Correct directory to start probe:C:\IBM\Tivoli\Netcool\omnibus\probes\win32>
nco_p_wineventlog.bat -messagelevel debug -messagelog stdout
```

Missing events during Resynch under high load

Under certain conditions the probe may encounter missing events during a Resynch operation.

The probe may encounter missing events when a Resynch operation is carried out in the following conditions:

- All log files are enabled for resynchronization.
- There is a high load for all the enabled log files.

Error with the retry function

If the Windows Event Source service is shut down or brought back up from an inactive status, the probe will attempt to reconnect to the event source regardless of the setting of the **Retry** property.

Enabling the retry function on a 32-bit Windows machine

If you are running the probe on a 32-bit Windows machine and have set the **SourceType** property to DotNet, to use the retry function, you must create an application configuration file named nco_p_wineventlog.exe.config and copy the file under the directory: %OMNIHOME%/probes/win32

The nco_p_wineventlog.exe.config file must contain the following code:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <runtime>
    <legacyUnhandledExceptionPolicy enabled="1"/>
  </runtime>
</configuration>
```

Summary field gets truncated

Any 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 (the $name element) and its description (the $description or $context_EventDescription element). The Summary field can accept values up to 255 characters long. If the concatenated value exceeds 255 characters, it is truncated.

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

© Copyright IBM Corp. 2006, 2019

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.

SC14-7648-06