Netcool/OMNIbus Flat File Writer Gateway 4.4

Reference Guide
June 26, 2020

IBM

SC23-7941-07

Notice

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

Edition notice

This edition (SC23-7941-07) applies to version 4.4 of IBM Tivoli Netcool/OMNIbus Flat File Gateway and to all subsequent releases and modifications until otherwise indicated in new editions.

The edition replaces SC23-7941-06.

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 Flat File Gateway documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM Tivoli Netcool/OMNIbus 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**
SC23-7941-01	December 19, 2007	First IBM publication.
SC23-7941-02	August 22, 2008	Support for Linux® for zSeries added.
SC23-7941-03	August 19, 2008	Guide rewritten to reflect updated gateway architecture.
SC23-7941-04	November 30, 2012	Guide updated for Netcool/OMNIbus V7.4 release. "Installing the gateway" on page 2 updated.
SC23-7941-05	December 6, 2013	"Summary" on page 1 updated. "Flat File Gateway properties" on page 7 updated.
SC23-7941-06	July 20, 2017	Descriptions of the following deprecated properties have been removed from"Flat File Gateway properties" on page 7 : • **`Ipc.SSLEnable`** • **`Ipc.SSLCertificate`** • **`Ipc.SSLPrivateKeyPassword`**
SC23-7941-07	June 26, 2020	"Summary" on page 1 updated. Descriptions for **`Gate.File.FileName`** and **`Gate.File.FileMode`** updated in "Flat File Gateway properties" on page 7. "Known issues with the Flatfile Gateway" on page 24 added.

Conventions used in this guide

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

Operating system-dependent variables and paths

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

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

```
$OMNIHOME/gates
```

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

```
%OMNIHOME%\gates
```

For gateways supported on UNIX, Linux, and Windows operating systems, gateway guides use the standard UNIX conventions for specifying environment variables and describing directory paths. When using the Windows command line with these gateways, 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.

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/platform/`*`arch`*

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

Note : This gateway 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

Gateways 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. Flat File Gateway

The Flat File Gateway is a unidirectional gateway that reads alerts from the Netcool/OMNIbus ObjectServer, and writes the details to a flat file. The gateway can receive `insert`, `update`, and `delete` notification information from multiple tables within the ObjectServer.

This guide contains the following sections:

- "Summary" on page 1
- "Flat File Gateway properties" on page 7
- "Mapping" on page 22
- "Example configuration file" on page 23
- "Running the gateway" on page 23

Summary

Each gateway works in a different way to interface with the ObjectServer, and therefore has specific features, default values, and changeable attributes. Use this summary information to learn about this gateway.

The following table provides a summary of the Flat File Gateway.

Table 3. Summary

Gateway target	Flat File
Gateway executable name	`nco_g_file`
Package Version	4.4
Gateway supported on	For details of supported operating systems, see the following Release Notice on the IBM Software Support website: http://www-01.ibm.com/support/docview.wss?uid=swg21613144

Configuration files	`$OMNIHOME/gates/file/file.map`
	`$OMNIHOME/gates/file/NCO_GATE.props`
	`$OMNIHOME/gates/file/file.reader.tblrep.def`
	`$OMNIHOME/gates/file/file.startup.cmd`

Requirements	IBM Tivoli Netcool/OMNIbus V8.1 or above NSocket library files (for IPv6 support) `common-libngtk-1_3` (UNIX) `common-libngobjserv-1_6` (UNIX)
Remote connectivity	Available
Failover/failback functionality	Not available
IP environment	IPv4 and IPv6 For details of configuring failover/failback, see "Failback" on page 6.

Table 3. Summary (continued)

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 gateway versions and platforms

The following table shows the supported Flat File Writer Gateway versions and the operating systems on which they run.

Table 4. Supported versions and platforms

Gateway Version	Binary	Operating System
Flat File Gateway running on IBM Tivoli Netcool/OMNIbus V8.1.0 or above.	`nco_g_file`	Solaris 10 AIX 6.1 Linux RHEL 5, SLES10 Linux for zSeries SLES 10, RHEL 5 Windows 2008 Server

Installing the gateway

There are separate procedures for installing the gateway on each version of Tivoli Netcool/OMNIbus.

Follow the procedure for the version of Tivoli Netcool/OMNIbus that your site uses.

Installing probes and gateways on Tivoli Netcool/OMNIbus V8.1

From Tivoli Netcool/OMNIbus V8.1 onwards, Tivoli Netcool/OMNIbus probes and gateways can be installed using the IBM Installation Manager. One of the key features of Installation Manager is that all platforms are shipped in a single ZIP file, which means that you do not have to select the platform that you require; Installation Manager does it for you.

Before you can install a probe or gateway, you must have installed and configured Installation Manager and Tivoli Netcool/OMNIbus. To install probes and gateways, you must make sure that the Core Tivoli Netcool/OMNIbus features **Probe Support** and **Gateway Support** respectively are installed.

Installing probes and gateways using the Command Line Tool

To install the probe or gateway using the Command Line Tool, run the following command:

```
installation_manager_location/eclipse/tools/imcl -c install
com.ibm.tivoli.omnibus.integrations.integration_name -repositories
repository_containing_required_integration -installationDirectory
location_of_netcool_omnibus_install_you_are_installing_into
```

Where *integration_name* specifies the name of the probe or gateway that you want to install.

You will be prompted to agree to the terms and conditions of the license as a prerequisite for installing the integration. If you have already reviewed the license and want to skip the manual acceptance, add the `-acceptLicense` option to the `install` command to silently agree to the license.

The following is an example command used to install the SNMP Probe:

```
imcl -c install com.ibm.tivoli.omnibus.integrations.nco-p-mttrapd -
repositories /home/my_home_dir/nco-p-mttrapd_im_package -
installationDirecory /opt/IBM/tivoli/netcool
```

Where `/home/my_home_dir/nco-p-mttrapd_im_package` contains the unzipped contents of the SNMP Probe Installation Manager package.

Note : The command line tool does not add the repository permanently to the Installation Manager instance. If you subsequently start the Installation Manager GUI, the repositories will not be present in the **Repositories** dialog box.

Uninstalling probes and gateways using the Command Line Tool

To uninstall the probe or gateway using the Command Line Tool, run the following command:

```
installation_manager_location/eclipse/tools/imcl uninstall
com.ibm.tivoli.omnibus.integrations.integration_name -installationDirectory
location_of_netcool_omnibus_install_you_are_uninstalling_from
```

Where *integration_name* specifies the name of the probe or gateway that you want to uninstall.

The following is an example command used to uninstall the SNMP Probe:

```
imcl uninstall com.ibm.tivoli.omnibus.integrations.nco-p-mttrapd -
installationDirecory /opt/IBM/tivoli/netcool
```

Installing probes and gateways using the GUI

To install the probe or gateway using the GUI, use the following steps:

1. Unzip the IM package that contains the probe or gateway into a directory of your choosing. A file called `repository.config` will appear after unzipping the IM package.
2. Start Installation Manager using the following command:

   installer_path/IBMIM

   Where *installer_path* is the path to the Installation Manager directory.
3. Perform the following menu actions to display the repository dialog box:

   Files > **Preferences** > **Repositories.**
4. Use the button **Add Repository** in the repository dialog box to point to the repository that contains the unzipped IM package that contains the probe or gateway. This is the repository that contains the `repository.config` file.
5. Click the **Install software packages** icon.
6. Select the name of the probe or gateway that you want to install.
7. Click **Next**.
8. Click **I accept** when the Licensing panel appears.
9. Highlight **IBM Tivoli Netcool OMNIbus** in the **Package Group Name** field.
10. Click **Next**.
11. Click **Next**.
12. Click **Install**.
13. When the **Install Packages** panel appears indicating that you have successfully installed the probe or gateway, click **Finish**.

Uninstalling probes and gateways using the GUI

To uninstall the probe or gateway, use the following steps:

1. Start Installation Manager using the following command:

   `installer_path/IBMIM`

   Where *installer_path* is the path to the Installation Manager directory.

2. Click the **Uninstall software packages** icon.
3. Select the name of the probe or gateway that you want to uninstall.
4. Click **Next**.
5. Click **Uninstall**.
6. When the **Install Packages** panel appears indicating that you have successfully uninstalled the probe or gateway, click **Finish**.

Configuring IBM Tivoli Netcool/OMNIbus

After installing the gateway, you must configure Tivoli Netcool/OMNIbus.

To configure Netcool/OMNIbus on UNIX and Linux operating systems, create the server NCO_GATE in the interfaces files using `nco_xigen`.

Note : For details about using `nco_xigen`, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

To configure Netcool/OMNIbus on Windows operating systems, create the server NCO_GATE in the interfaces files using **Server Editor**.

Note : For details about using **Server Editor**, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

Gateway quick start guide

This section describes how to start the gateway with the minimum required configuration and assumes that you already have a version of Netcool/OMNIbus installed and configured.

To start the gateway with the minimum required configuration, use the following steps:

1. Install the gateway.
2. Edit the gateway properties file and set the following options:

   ```
   MessageLevel : 'debug'
   Server : 'NCOMS'
   ```

   Where NCOMS is the name of the ObjectServer.

3. Edit the `$NCHOME/etc/omni.dat` file to include the ObjectServer information.
4. Run the `$NCHOME/bin/nco_igen` command to generate the interfaces file.
5. To check that the ObjectServer is available, run the `$OMNIHOME/bin/nco_ping` *server* command.

   For example, `%OMNIHOME/bin/nco_ping` *NCOMS*

6. Start the gateway.
7. Check the log file to confirm that events are being sent.

Note : The quick start guide is only to start the gateway with the minimum configuration necessary. To run the gateway in the most productive manner possible for your environment you should review the gateway configuration and property values.

Features of the Flat File Gateway

The Flat File Writer Gateway has various features that enable you to read alerts from the ObjectServer and to write them to a flat file.

- “Centralized property management” on page 5
- “Setting the format of headers and trailers added to alerts” on page 5
- “Receiving notifications from multiple tables within the ObjectServer” on page 5

- “Writing alert data to the file as name-value pairs” on page 6
- “Specifying start and end characters for all value fields” on page 6
- “Passing table data” on page 6
- “Mapping table data” on page 6
- “Failback” on page 6
- “Process Agent control” on page 6
- “Error handling” on page 7

Centralized property management

The gateway uses centralized property management and separates properties from data processing configuration. To configure the common gateway properties and the Flat File Writer Gateway properties, use the `NCO_GATE.props` properties file.

To configure data processing, use the table replication definition file `$OMNIHOME/gates/file/file.reader.tblrep.def`, and the mapping file `$OMNIHOME/gates/file/file.map`.

Setting the format of headers and trailers added to alerts

Whenever a `delete`, `update`, or `insert` occurs in the ObjectServer fields, the gateway writes header and trailer information with the changed alerts to the file.

Using the properties file, you can define the format in which the gateway adds the header and trailer in the alerts. The following example shows header and trailer format definitions of the `deletes, updates,` and `inserts`:

```
# Gate.File.DeleteHeader : '\nStart of delete line\n'
# Gate.File.DeleteTrailer : '\nEnd of delete line\n'
# Gate.File.UpdateHeader : '\nStart of update line\n'
# Gate.File.UpdateTrailer : '\nEnd of update line\n'
# Gate.File.InsertHeader : '\nStart of insert line\n'
# Gate.File.InsertTrailer : '\nEnd of insert line\n'
```

Where `\`n instructs the start of the next line.

Receiving notifications from multiple tables within the ObjectServer

The gateway can receive details of `updates`, `inserts`, and `deletes` to multiple tables from the ObjectServer. By default, the gateway writes all the `updates` to the file, then all the `inserts`, and then all the `deletes`.

The alert data written to the output file has the following structure:

```
UPDATE: ...
UPDATE: ...
INSERT: ...
INSERT: ...
DELETE: ...
```

The gateway can also write the details of the notifications from multiple tables to the output file with separators between the notifications from each table. To do this, set the **`Gate.File.BatchSeparator`** property to TRUE, and then specify the separators that the gateway adds to the start and to the end of each batch using the **`Gate.File.BatchHeader`** and **`Gate.File.BatchTrailer`** properties respectively. Using the default values for the batch separators, the alert data written to the output file has the following structure:

```
########## BATCH START ##########
Table:alerts.status
UPDATE: ...
UPDATE: ...
########## BATCH END ##########
########## BATCH START ##########
Table:alerts.status
```

```
INSERT: ...
INSERT: ...
########## BATCH END ##########
########## BATCH START ##########
Table:alerts.status
DELETE: ...
########## BATCH END ##########
```

Writing alert data to the file as name-value pairs

By default, the gateway writes the alert data to the file as comma separated entries.

The gateway can also write the alert data to the file as name-value pairs instead. To do this, set the **`Gate.File.OutFieldNames`** property to TRUE and specify the character that the gateway uses to separate each column name from its corresponding value using the **`Gate.File.ColSepartor`** property.

Specifying start and end characters for all value fields

By default, the gateway only adds the start and end characters specified by **`Gate.File.StartString`** and **`Gate.File.EndString`** properties to string fields.

The gateway can also be configured to write start and end characters to all string values. To do this, set the **`Gate.File.ProtectAllFile`** property to TRUE.

Passing table data

The gateway can replicate the data in any ObjectServer table in the destination file. Details of the tables to be replicated are stored in the table replication definition file. Use the **`Gate.Reader.TblReplicateDefFile`** property to specify the location of the table replication definition file.

Mapping table data

The gateway writes the alerts received from the various tables of ObjectServer into the flat file in a format defined by the map definition file. To specify the map definition file, use the **`Gate.MapFile`** property

Failback

The failback function comes into operation when the gateway loses its connection to the primary ObjectServer. When the primary ObjectServer fails, the reader fails back to the backup ObjectServer as configured using `nco_xigen`. To enable failback, set the **`Gate.Reader.FailbackEnabled`** property to TRUE

When the reader has detected that it is now connected to a backup ObjectServer, it periodically polls for the return of the primary ObjectServer for the period specified by the **`Gate.Reader.FailbackTimeout`** property. Connection is checked at startup and whenever an event is sent. When the primary ObjectServer is detected again, the reader automatically fails over to the primary ObjectServer.

On detection of the reader failover or failback, the writer will shut down the gateway and rely on the Process Agent to restart the gateway.

Process Agent control

You can control how the gateway runs by using Process Agent control.

The gateway can be run under PA control. The **`Gate.PAAware`** property indicates whether the gateway is PA aware. The **`Gate.PAAwareName`** property indicates which PA is running the gateway.

Note : These properties are maintained automatically by the PA server and provide information only. Do not change these properties manually.

Error handling

You can troubleshoot problems with the gateway by consulting error messages. To help you do this, the gateway has configurable error handling.

Error handling is provided by the Netcool/OMNIbus Gateway Toolkit (NGTK) library. To specify that the NGTK library logs debug messages, set the **`Gate.NGtkDebug`** property to TRUE.

Configuration

Before using the gateway, you must make various configuration settings.

Use this section to update any configuration settings that you need to make.

- "Flat File Gateway properties" on page 7
- "Startup command file" on page 20
- "Table replication definition file" on page 21
- "Mapping" on page 22
- "Example configuration file" on page 23
- "Running the gateway" on page 23

Flat File Gateway properties

You use the Flat File Gateway properties to specify how the gateway interfaces with the data source. You can override the default values by using the configuration file.

The following tables describes the properties used to create a Flat File Gateway writer.

Note : For details of the data type definitions for the alerts tables forwarded to the ObjectServer, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*, (SC23-6371).

Common Netcool/OMNIbus properties

Table 5. Common gateway properties

Property Name	Command line option	Description
`MaxLogFileSize` *integer*	`-maxlogfilesize` *integer*	Use this property to specify the size (in bytes) that the gateway allocates for the log file. When the log file reaches this size, the gateway renames the log file by appending the name with the suffix .old and creates a new log file. The default value is `1024`
`MessageLevel` *string*	`-messagelevel` *string*	Use this property to specify the reporting level of the log file messages. The default value is `'warn'`
`MessageLog` *string*	`-messagelog` *string*	Use this property to specify the location of the message log file. The default value is `'$OMNIHOME/log/NCO_GATE.log'`

Table 5. Common gateway properties (continued)

Property Name	Command line option	Description
Name *string*	-name *string*	Use this property to specify the name of the current gateway instance. If you want to run multiple gateways on one machine, you must use a different name for each instance. The default value is 'NCO_GATE'
PropsFile *string*	-propsfile *string*	Use this property to specify the location of the gateway properties file. The default value is '$OMNIHOME/etc/NCO_GATE.props'
Help *string*	-help *string*	Use this property to instruct the gateway to display application help information on startup and exit. The default value is FALSE
UniqueLog *string*	-uniquelog *string*	Use this property to specify whether or not log file names are made unique by adding the Process ID (PID) of the gateway to the file name. The default value is FALSE
Version *string*	-version *string*	Use this property to instruct the gateway to display information about the application version on startup and exit. The default value is FALSE

Common IPC properties

Table 6. Common IPC properties

Property Name	Command line option	Description
Ipc.ClientOverrideSybase *string*	-clientoverridesybase *string*	Use this property to specify whether the client overrides the Sybase settings. The default value is FALSE

Table 6. Common IPC properties (continued)

Property Name	Command line option	Description
`Ipc.OpenClientDebug` *string*	`-openclientdebug` *string*	Use this property to specify that the probe logs client debug messages. The default value is `FALSE`
`Ipc.QueueSize` *integer*	`-queuesize` *integer*	Use this property to specify the size of the internal server queues used by the middleware. The default value is `1024` You should only change this value if instructed by IBM Software Support.
`Ipc.SingleThreaded` *boolean*	`-singlethreaded` *boolean*	Use this property to specify that the server middleware runs single threaded. The default value is `FALSE`
`Ipc.StackSize` *integer*	`-stacksize` *integer*	Use this property to specify the size (in bytes) of the internal server thread stacks. The default value is 34816
`Ipc.Timeout` *integer*	`-timeout` *integer*	Use this property to specify the period (in seconds) that the gateway allows for a response from the server before timing out. The default value is `60`
`Ipc.TruncateVendorLogFile` *string*	`-truncatevendorlogfile` *string*	Use this property to specify whether the gateway truncates the middleware vendor log file on startup. The default value is `TRUE`
`Ipc.VendorLogFileSize` *integer*	`-vendorlogfilesize` *integer*	Use this property to specify the maximum size (in bytes) of the middleware vendor file. The default value is `1024`
`Connections` *integer*	`-connections` *integer*	Use this property to specify the maximum number of client connections that can be made to the gateway server. The default value is `30`

Common Gateway Properties

Table 7. Common Gateway Properties

Property Name	Command line option	Description
`Gate.CacheHashTblSize` *integer*	`-cachehtblsize` *integer*	Use this property to specify the number of elements (database rows) that the gateway allocates for the hash table cache. The default value is `5023`.
`Gate.MapFile` *string*	`-mapfile` *string*	Use this property to specify the mapping file for the gateway to use. The default value is `'$OMNIHOME/gates/file/file.map'`.
`Gate.Mapper.Debug` *boolean*	`-mapperdebug` *boolean*	Use this property to specify whether or not the gateway includes mapper debug messages in the debug log. The default value is TRUE
`Gate.Mapper.ForwardHistoricDetails` *string*	`-mapperforhistdtls` *string*	Use this property to specify whether the gateway forwards all historic details on converted update. The default value is FALSE
`Gate.Mapper.ForwardHistoricJournals` *string*	`-mapperforhistjrnl` *string*	Use this property to specify whether the gateway forwards all historic journals on converted update. The default value is FALSE
`Gate.StartupCmdFile` *string*	`-startupcmdfile` *string*	Use this property to specify the location of the startup command file. The default value is `'$OMNIHOME/gates/file/file.startup.cmd'`
`Gate.Reader.Debug` *string*	`-readerdebug` *string*	Use this property to specify whether the gateway includes gateway reader debug messages in the debug log. The default value is TRUE

Table 7. Common Gateway Properties (continued)

Property Name	Command line option	Description
`Gate.Reader.Description` *string*	`-readerdescription` *string*	Use this property to specify the application description for the reader connection. This description is used in triggers and allows you to determine which component of the gateway attempted to perform an action. The default value is `'Gateway Reader'`
`Gate.Reader.DetailsTableName` *string*	`-readerdetailstblname` *string*	Use this property to specify the name of the details table that the gateway reads. The default value is `'alerts.details'`
`Gate.Reader.FailbackEnabled` *string*	`-readerfailbackenabled` *string*	Use this property to specify whether the gateway fails back to the backup ObjectServer when disconnected from the primary ObjectServer. The default value is `TRUE`
`Gate.Reader.FailbackTimeout` *integer*	`-readerfailbacktimeout` *integer*	Use this property to specify the frequency (in seconds) with which the gateway attempts to fail back to the primary system following a system failover. The default value is `30`

Table 7. Common Gateway Properties (continued)

Property Name	Command line option	Description
Gate.Reader.IducFlushRate *integer*	-readeriducflushrate *integer*	Use this property to specify the rate (in seconds) of the granularity of the reader. If you set this property to 0, the reader gets its updates at the same granular rate as that of the ObjectServer to which it is connected. The default value is 0 **Note :** If you set this property to a value greater than 0, the reader issues automatic IDUC flush requests to the ObjectServer with this frequency. This enables the reader to run at a faster granularity than that of the ObjectServer, thus enabling the gateway to capture more detailed event changes in systems where the ObjectServer itself has high granularity settings.
Gate.Reader.JournalTableName *string*	-readerjournaltblname *string*	Use this property to specify the name of the journal table that the gateway reads. The default value is 'alerts.journal'
Gate.Reader.LogOSSql *boolean*	-readerlogossql *boolean*	Use this property to specify whether the gateway logs all SQL commands sent to the ObjectServer in debug mode. The default value is FALSE
Gate.Reader.Password *string*	-readerpassword *string*	Use this property to specify the password associated with the user specified by the **Gate.Reader.Username** property. The default value is '' **Note :** This password must be encrypted using the nco_g_crypt utility. For information about using nco_g_crypt, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

Table 7. Common Gateway Properties (continued)

Property Name	**Command line option**	**Description**
`Gate.Reader.ReconnectTimeout` *integer*	`-readerreconntimeout` *integer*	Use this property to specify the time (in seconds) between each reconnection poll attempt that the gateway makes if the connection to the ObjectServer is lost. The default value is `'30'`
`Gate.Reader.Server` *string*	`-readerserver` *string*	Use this property to specify the name of the ObjectServer from which the gateway reads alerts. The default value is `'NCOMS'`
`Gate.Reader.StatusTableName` *string*	`-readerstatustblname` *string*	Use this property to specify the name of the status table that the gateway reads. The default value is `'alerts.status'`
`Gate.Reader.TblReplicateDefFile` *string*	`-readertblrepdef` *string*	Use this property to specify the location of the table replication definition file. The default value is `'$OMNIHOME/gates/file/file.reader.tblrep.def'`
`Gate.Reader.Username` *string*	`-readerusername` *string*	Use this property to specify the user name that is used to authenticate the ObjectServer connection. The default value is `'root'`
`Gate.Transfer.FailoverSyncRate` *integer*	`-fsyncrate` *integer*	Use this property to specify the rate (in seconds) of the failover synchronization. The default value is `'60'`
`Gate.NGtkDebug` *boolean*	`-ngtkdebug` *boolean*	Use this property to specify whether or not the NGTK library logs debug messages. The default value is `'TRUE'`

Table 7. Common Gateway Properties (continued)

Property Name	Command line option	Description
Gate.PAAware *integer*	-paaware *integer*	This property indicates whether or not the gateway is Process Agent (PA) aware. The default value is '0' (the gateway is not PA-aware). **Note :** This property is maintained by the PA server and is included in the properties file for information only.
Gate.PAAwareName *string*	-paname *string*	This property indicates the name of the PA controlling the gateway. The default value is '' **Note :** This property is maintained by the PA server and is included in the properties file for information only.

Flat File Gateway Properties

Table 8. Flat File Gateway Properties

Property Name	Command line option	Description
Gate.File.BatchHeader *string*	-filebatchhdr *string*	Use this property to specify the format of the header that the gateway adds to indicate the start of each new table written to the file. The default value is '########## BATCH START ##########' **Note :** The gateway only uses the batch header if the **Gate.File.UseBatchSeparator** property is set to TRUE.
Gate.File.BatchTrailer *string*	-filebatchtl *string*	Use this property to specify the format of the trailer that the gateway adds to indicate the end of each table written to the file. The default value is '########## BATCH END ##########' **Note :** The gateway only uses the batch trailer if the **Gate.File.UseBatchSeparator** property is set to TRUE.

Table 8. Flat File Gateway Properties (continued)

Property Name	Command line option	Description
`Gate.File.ColSeparator` *string*	`-filecolseparator` *string*	Use this property to specify the character that the gateway inserts between each column name and its corresponding value. The default value is `':'`
`Gate.File.DateFormat` *string*	`-dateformat` *string*	Use this property to specify the date format that the gateway uses for any date fields written to the socket. The default value is `'%x %X'`
`Gate.File.DeleteHeader` *string*	`-deleteheader` *string*	Use this property to specify the format of the header added to the start of each delete alert written to the socket. The default value is `'DELETE: '`
`Gate.File.DeleteTrailer` *string*	`-deletetrailer` *string*	Use this property to specify the format of the trailer added to the end of each delete alert written to the socket. The default value is `'\n'`
`Gate.File.DoTimeStamp` *boolean*	`-dotimestamp` *boolean*	Use this property to specify whether the gateway prefixes each event with the date and time of its arrival. The default value is `'FALSE'`
`Gate.File.EndString` *string*	`-endstring` *string*	Use this property to specify whether the gateway prefixes each event with the date and time of its arrival. The default value is `'"'`

Table 8. Flat File Gateway Properties (continued)

Property Name	Command line option	Description
`Gate.File.FileMode` *integer*	`-filemode` *integer*	Use this property to specify the file permissions to set for the file to which the gateway writes. The parameter that you specify is contingent on the umask value of the shell in which you start the gateway. The umask value determines the default permissions for any file created by a process. The **`Gate.File.FileMode`** property accepts any permission that is not specified by umask. The default value is `'644'` **Note :** If the value configured for this property is outside of the range of permitted values, the permission will be undetermined. The final file permission will be bound by the operating system. In Windows, there is an OS predisposition to force 644 permission regardless of what is being configured in this property.
`Gate.File.FileName` *string*	`-filefname` *string*	Use this property to specify the name of the file to which the gateway writes alert data from the ObjectServer. The default value is `'output.log'`. **Note:** This property supports date and time stamping using date parameters. For example, if you specify `output.#y-#m-#d-#h`, the gateway appends the date and time at which the file is created to the file name. In this example, `#y` indicates the year (00-99), `#m` indicates the Month as a number (01-12), `#d` indicates the Day of the month (01-31), `#H` indicates the in 24 hour format (00-23), and `#M` indicates the minute (00-59). The gateway is incapable of knowing if this output file is absent, so if this output file gets deleted while the gateway is running, it will not know and will not exit.

Table 8. Flat File Gateway Properties (continued)

Property Name	**Command line option**	**Description**
`Gate.File.InsertHeader` *string*	`-InsertHeader` *string*	Use this property to specify the format of the header added to the start of each insert alert written to the file. The default value is `'INSERT'`
`Gate.File.InsertTrailer` *string*	`-inserttrailer` *string*	Use this property to specify the format of the trailer added to the end of each insert alert written to the file. The default value is `'\n'`
`Gate.File.MaxFileSize` *integer*	`-maxfilesize` *integer*	Use this property to specify the maximum size (in bytes) of the flat file to which the gateway sends output. When the file reaches this size, the gateway sends its output to a new file. The default value is `0`. **Note :** The maximum value that can be assigned to this property is 2,147,000,000 bytes. **Note :** To specify whether the gateway maintains two concurrent files or multiple files, use the **`Gate.File.OutputMode`** property.
`Gate.File.OutputFieldNames` *boolean*	`-fileoutputfldnames` *boolean*	Use this property to specify whether the gateway writes the alert data to the file as name-value pairs. The default value is `FALSE` **Note :** If the name fields appear as `""` in the output for all fields, you must update the mapping file, adding the appropriate field name. For example, to output `"Id":"Heartbeat"`, you must change the entry in the mapping file to: `'Id' = '@Identifier'`.

Table 8. Flat File Gateway Properties (continued)

Property Name	Command line option	Description
`Gate.File.OutputMode` *string*	`-fileoutputmode` *string*	Use this property to specify whether the gateway maintains two concurrent files or a series of successive, incrementally named files. Possible values are: • `HISTORICAL`: The gateway writes to the file specified by the **`Gate.File.FileName`** property; when it reaches the maximum file size, it starts writing a new file named _1. It repeats this each time the maximum file size is reached, incrementing the filename by one each time; for example, the gateway creates files named `FILE, FILE_1, FILE_2`, and so forth • `CURRENT`: The gateway writes to the file specified by the **`Gate.File.FileName`** property; when it reaches the maximum file size, it renames this file `.old` and creates a new file named to continue output; for example, the gateway creates files named `FILE` and `FILE.old`. The default value is `"HISTORICAL"` **Note :** The maximum size for any of these files is specified by the **`Gate.File.MaxFileSize`** property.
`Gate.File.ProtectAllFields` *boolean*	`-fileprotectall` *boolean*	Use this property to specify whether the gateway adds the characters defined by the **`Gate.File.StartString`** and **`Gate.File.EndString`** properties to integer fields as well as string fields. The default value is FALSE
`Gate.File.Separator` *string*	`-separator` *string*	Use this property to specify the character that the gateway inserts between the fields written to the file. The default value is `','`

Table 8. Flat File Gateway Properties (continued)

Property Name	Command line option	Description
`Gate.File.StartString` *string*	`-filestartstring` *string*	Use this property to specify the character that the gateway adds to the start of string values written to the file. The default value is `'"'`
`Gate.File.Sync` *boolean*	`-filesync` *boolean*	Use this property to specify the whether synchronous write is enabled The default value is `FALSE`
`Gate.File.Truncate` *boolean*	`-filetruncate` *boolean*	Use this property to specify the file is truncated on startup, or whether the writer continues to append to the end of the file. The default value is `FALSE`
`Gate.File.UpdateHeader` *string*	`-fileuphdr` *string*	Use this property to specify the value that the gateway adds to the start of each update alert written to the file. The default value is `'UPDATE: '`
`Gate.File.UpdateTrailer` *string*	`-fileupdtl` *string*	Use this property to specify the value that the gateway adds to the end of each update alert written to the file. The default value is `'\n'`
`Gate.File.UseBatchSeparator` *boolean*	`-fileusebatchsep` *boolean*	Use this property to specify whether the gateway writes details of changes from multiple tables to the file in batches, with a separator between each table. The default value is `FALSE`

Properties taking Boolean values

For binary-typed properties (namely boolean properties taking the value TRUE or FALSE) or range-typed properties (namely those taking an integer range of values), using a value that does not conform to the possible settings will result in the use of the property's default value.

Boolean-Type File Gateway properties:

- **`Gate.File.DoTimeStamp`**
- **`Gate.File.OutputFieldNames`**
- **`Gate.File.Truncate`**
- **`Gate.File.DoTimeStamp`**
- **`Gate.File.OutputFieldNames`**

- **Gate.File.ProtectAllFields**
- **Gate.File.Sync**
- **Gate.File.UseBatchSeparator**

Range-Type File Gateway properties:

- **Gate.File.FileMode**
- **Gate.File.MaxFileSize**

Startup command file

The startup command file contains a set of commands that the gateway executes each time it starts.

You can specify the location of the startup command file using the generic Netcool/OMNIbus **Gate.StartupCmdFile** property.

The default startup command file is located in the following directory: `$OMNIHOME/gates/file/file.startup.cmd`

The default startup command file contains example commands. You should make a copy of the default file for future reference.

You can use the following commands within the startup command file:

- SHOW PROPS - Use this command to display the current configuration of the gateway by listing all properties and their values.
- GET PROPERTY '*property_name*' - Use this command to return the value of the property specified in *property_name* from the gateway properties file.
- SET PROPERTY '*property_name*' TO ('*string*' | *integer* | TRUE | YES | FALSE | NO); - Use this command to set the value of the property specified in *property_name* in the gateway properties file.
- SET LOG LEVEL TO - Use this command to set the level of message logging for the gateway. This command can take the following values: `fatal, error, warn, info` or `debug`. The default logging level is `warn`.

These commands can also be entered using the SQL interactive interface (`nco_sql`). For more information about using the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

For more information about the startup command file, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

TRANSFER command

The TRANSFER command transfers data from an ObjectServer table to a target table using a transfer map. The data can also be filtered. This command is most useful is cases where once populated, the target table contents are unlikely to change.

The TRANSFER command takes the following syntax:

TRANSFER FROM '*source*' [TO '*target*']

The TRANSFER command is usually specified in the startup command file. To run the TRANSFER command from the startup command file use the following example:

```
TRANSFER FROM 'source_table' TO 'target_table'
            VIA FILTER 'Colname != \'Severity\''
            WITH DELETE VIA 'Column_Name <> \'Severity\''
            USING TRANSFER_MAP GatewayTablesMap;
```

Note : To replicate data from dynamic secondary tables you must use the **Gate.ResyncTables** property in the table replication definition file.

This command can also be entered using the SQL interactive interface (nco_sql). For more information about using the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

For more information about the startup command file and the TRANSFER command, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table replication definition file

The gateway replicates data between ObjectServer tables and the gateway target. The table replication definition file is used to define which tables and event types are monitored in Tivoli Netcool/OMNIbus and forwarded to the target that the gateway is configured to send data to.

You can specify the location of the table replication definition file using following generic Tivoli Netcool/OMNIbus property.

Gate.Reader.TblReplicateDefFile

The default table replication definition file is in the following directory: $OMNIHOME/gates/file/file.reader.tblrep.def

The default table replication definition file contains example commands. You should make a backup copy of the default file for future reference.

Note : You should use the REPLICATE command to replicate data from the primary tables (alerts.status, alerts.journal, alerts.details) and dynamic secondary tables (if required).

You can add one or more optional clauses to the REPLICATE command to further process the data during replication. The available commands are listed in the following syntax example. Use the optional clauses in the order in which they are listed in the syntax. For example, when using both the

FILTER WITH and AFTER IDUC DO clauses, the FILTER WITH clause must precede the AFTER IDUC DO clause.

```
REPLICATE ALL | (INSERTS, UPDATES, DELETES)
FROM TABLE sourcetable
USING MAP mapname
[FILTER WITH filter]
[INTO targettable]
[ORDER BY order, ... ]
[AFTER IDUC DO afteriduc] ;
```

Table 9. Optional replication commands

Command	Description
FILTER WITH '*filter*'	Filters the database rows selected for replication, where *filter* defines the filter that the gateway uses in the WHERE clause of the SQL SELECT. Filtering is positive by default, which means that only those events that match the filter definition are replicated. You can use a negative filter by putting an exclamation mark (!) before the equals sign (=) in the filter clause. For example, the following filter clause replicates all events whose severity is not 5: FILTER WITH 'Severity !=5'
ORDER BY '*order*'	Order results by the SQL SELECT ORDER BY clause used to get data. A potential use case might be to order by first occurrence, so that alerts are processed in chronological order, in which case the value specified for *order* would be 'FirstOccurrence'.

Table 9. Optional replication commands (continued)

Command	Description
AFTER IDUC DO '*afteriduc*'	Updates replicated rows, where *afteriduc* specifies which field to update with what value. This uses the SQL UPDATE action to execute on rows retrieved by the SQL SELECT action used to get data, e.g. 'SentToCRM=1'.

Mapping

Each gateway uses its own specific mapping.

Mappings for use with the Flat File Gateway must use the following syntax:

```
CREATE MAPPING mappingname
(
’’ = ’@fieldname’ [ON INSERT ONLY],
[ , ’’= ’@fieldname’ [ON INSERT ONLY]]...
) ;
```

Where:

mappingname is the name of the mapping to be created.

fieldname must be the name of a field in the ObjectServer alerts.status table.

The optional ON INSERT ONLY controls the updating of the field during the life of the alert; when omitted, the field is updated for any change in the state of the alert. When included, the field is only set when the alert is created.

Example mapping

The following default mapping is supplied in the G_FILE.conf configuration file:

```
CREATE MAPPING FILE_MAP
(
'' = '@Identifier',
'' = '@Serial',
'' = '@Node',
'' = '@NodeAlias',
'' = '@Manager',
'' = '@Agent',
'' = '@AlertGroup',
'' = '@AlertKey',
'' = '@Severity',
'' = '@Summary',
'' = '@StateChange' CONVERT TO DATE,
'' = '@FirstOccurrence' CONVERT TO DATE,
'' = '@LastOccurrence' CONVERT TO DATE,
'' = '@InternalLast' CONVERT TO DATE,
'' = '@Poll',
'' = '@Type',
'' = '@Tally',
'' = '@Class',
'' = '@Grade',
'' = '@Location',
'' = '@OwnerUID',
'' = '@OwnerGID',
'' = '@Acknowledged',
'' = '@Flash',
'' = '@EventId',
'' = '@ExpireTime',
'' = '@ProcessReq',
'' = '@SuppressEscl',
'' = '@Customer',
'' = '@Service',
'' = '@PhysicalSlot',
'' = '@PhysicalPort',
'' = '@PhysicalCard',
'' = '@TaskList',
'' = '@NmosSerial',
'' = '@NmosObjInst',
'' = '@NmosCauseType',
'' = '@LocalNodeAlias',
'' = '@LocalPriObj',
'' = '@LocalSecObj',
'' = '@LocalRootObj',
'' = '@RemoteNodeAlias',
'' = '@RemotePriObj',
'' = '@RemoteSecObj',
'' = '@RemoteRootObj',
'' = '@X733EventType',
'' = '@X733ProbableCause',
'' = '@X733SpecificProb',
'' = '@X733CorrNotif',
'' = '@ServerName',
'' = '@ServerSerial',
'' = '@URL'
);
```

Example configuration file

The following Flat File writer is supplied in the `G_FILE.conf` configuration file:

```
START WRITER FILE_WRITER
(
TYPE = FILE,
REVISION = 1,
FILE = '/path/to/file',
MAP = FILE_MAP,
INSERT_HEADER = 'INSERT: ',
UPDATE_HEADER = 'UPDATE: ',
DELETE_HEADER = 'DELETE: ',
START_STRING = '"',
END_STRING = '"',
INSERT_TRAILER = '\n',
UPDATE_TRAILER = '\n',
DELETE_TRAILER = '\n'
);
```

Running the gateway

This section describes how to run the Flat File Gateway.

To run the gateway, use the in information in the following topics:

- “Running the gateway on UNIX” on page 23
- “Running the gateway on Windows” on page 23

Running the gateway on UNIX

To run the gateway on UNIX, run the following command:

```
$OMNIHOME/bin/nco_g_file
```

or use the `-config` command line option as follows:

```
$OMNIHOME/bin/nco_g_file -config /path/config_filename
```

Where *`path`* is the location of the configuration file and *`config_filename`* is the name of the configuration file.

Running the gateway on Windows

Perform the following lis to run the Flat File Gateway on Windows.

To run the gateway on Windows, run the following command:

```
%OMNIHOME%\bin\win32\nco_g_file.exe
```

or use the **`-propsfile`** command line option as follows:

```
%OMNIHOME%\bin\win32\nco_g_file.exe -propsfile \path\propsfile
```

Where *path* is the location of the property file and *propsfile* is the name of the property file.

Running the gateway as a Windows service

To run the gateway as a Windows service, use the following steps:

1. Run the following command to register the gateway as a service within the Service Control Manager: `nco_g_file.exe -install`
2. If you are running the gateway on the same machine as the ObjectServer, use the following command to register its dependency upon the ObjectServer service: `nco_g_file.exe -install -depend NCOObjectServer`
3. Select the following option from the **Start** menu: **Control Panel** > **Administrative Tools** > **Services**. The **Services** window is displayed.
4. Highlight the Flat File Gateway in the list of available services.
5. Click on **Start**.

Removing the gateway service

To remove the gateway service, use the following steps:

1. Stop the service using the **Services** window.
2. Run the following command: `nco_g_file.exe -remove`.

Known issues with the Flatfile Gateway

This section explains some known issues with the Flatfile Gateway.

Flat File Gateway fails to error handle superfluous text in the properties file

Currently, the Flat File Gateway cannot identify superfluous text in the its properties file (namely, `NCO_GATES.props`). If the gateway encounters an invalid property configured after superfluous text, the gateway ignores it and the rest of the property configurations from that point onwards, and the default property values will be used. No details about this failure will be logged in the gateway's log file.

The gateway allows invalid values to be set for the `MaxFileSize` property

Currently, when running the gateway on Windows or UNIX, if you set the value of the **`Gate.File.MaxFileSize`** property in the `NCO_GATES.props` properties file to a value that is outside of the range of `0` to `2,147,000,000` bytes, the gateway ignores the setting specified and runs using the value `1000` for the **`Gate.File.MaxFileSize`** property instead.

When running the gateway on Windows, if you set the value of the **`Gate.File.MaxFileSize`** property to a string value, again, the gateway ignores the setting specified and runs using the value `1000` for the `Gate.File.MaxFileSize` property instead. However, when running the gateway on UNIX, if you set the value of the **`Gate.File.MaxFileSize`** property to a string value, the gateway correctly fails to start and writes an error message to the gateway's log file.

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.

Part Number:

(1P) P/N:

SC23-7941-07