

Netcool/OMNIbus
Version 7 Release 3

Installation and Deployment Guide



Netcool/OMNIbus
Version 7 Release 3

Installation and Deployment Guide



Note

Before using this information and the product it supports, read the information in "Notices" on page 643.

This edition applies to version 7, release 3 of IBM Tivoli Netcool/OMNIbus (product number 5724-S44) and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 1994, 2011.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this publication	ix
Intended audience	ix
What this publication contains	ix
XXXX	xi
Accessibility	xii
Tivoli technical training	xiii
Support information	xiii
Conventions used in this publication	xiii

Chapter 1. Introduction to Tivoli Netcool/OMNIBus	1
Tivoli Netcool/OMNIBus components	1
The ObjectServer	2
Probes	3
Gateways	3
Desktop tools	3
Administration tools	3
The Web GUI component	4

Chapter 2. Backing up and restoring Tivoli Netcool/OMNIBus	9
---	----------

Chapter 3. Planning for installation or upgrade	11
Supported operating systems and compatible products	11
Prerequisites for operating systems	11
JRE requirements	14
Web browsers, JREs, and mobile devices for the Web GUI	14
User interface requirements	16
Online help requirements	16
Disk space requirements	17
Networking protocol support	18
Communication protocol	19
Compatibility with previous versions	19
Federal Information Processing Standard 140–2 (FIPS 140–2) support	26
FIPS 140–2 configuration checklist	26
Additional requirements for the Web GUI	28
Deployment considerations for the Web GUI	28
Central user registry	29
Single sign-on	30
Integration with other Tivoli products	31
Installation modes	33
About the launchpad	33
Overview of the Netcool home location	34
The Deployment Engine	34
Directory structure for the Deployment Engine	35
Deployment Engine user modes	36
Tivoli Integrated Portal components	38
Quick reference to getting started	38
Quick reference to upgrading	41

Chapter 4. Installing, upgrading, and uninstalling (UNIX and Linux)	45
Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)	45
Preparing to install or upgrade (UNIX and Linux)	46
Installation or upgrade prerequisites (UNIX and Linux)	47
Obtaining the installation package (UNIX and Linux)	47
Installing on UNIX and Linux	48
Installing using the installation wizard (UNIX and Linux)	49
Installing in console mode (UNIX and Linux)	52
Installing in silent mode (UNIX and Linux)	54
Verifying the Netcool/OMNIBus installation (UNIX and Linux)	59
Viewing the installation log files (UNIX and Linux)	60
Viewing installed packages (UNIX and Linux)	61
Installation directory structure (UNIX and Linux)	61
Upgrading on UNIX and Linux	64
Guidelines for upgrading to FIPS 140–2 mode (UNIX and Linux)	65
Upgrading using the installation wizard (UNIX and Linux)	68
Upgrading in console mode (UNIX and Linux)	72
Upgrading in silent mode (UNIX and Linux)	75
Manually migrating data (UNIX and Linux)	80
Viewing the migration log file (UNIX and Linux)	80
Modifying your V7.3.1 installation (UNIX or Linux)	81
Additional upgrade and migration notes (UNIX and Linux)	81
Notes on upgrading ObjectServer schemas to V7.3.1 schemas (UNIX and Linux)	81
Files migrated for an upgrade (UNIX and Linux)	85
Migrating your digital certificates and keys (UNIX and Linux)	86
IBM Tivoli Enterprise Console BAROC data migration (UNIX and Linux)	90
Performing postinstallation tasks (UNIX and Linux)	94
Setting Tivoli Netcool/OMNIBus environment variables (UNIX and Linux)	95
Checking the shared library paths	99
Configuring settings for online help access (UNIX and Linux)	100
Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)	102
Installing probes and gateways into the Tivoli Netcool/OMNIBus environment (UNIX and Linux)	103
Uninstalling Tivoli Netcool/OMNIBus (UNIX and Linux)	106
Uninstalling using the wizard (UNIX and Linux)	107

Uninstalling in console mode (UNIX and Linux)	108
Uninstalling in silent mode (UNIX and Linux)	108
Uninstalling probes and gateways (UNIX and Linux)	108
Chapter 5. Installing, upgrading, and uninstalling (Windows)	111
Installable Tivoli Netcool/OMNIbus features (Windows)	111
Preparing to install or upgrade (Windows)	112
Installation or upgrade prerequisites (Windows)	113
Obtaining the installation package (Windows)	113
Notes for Windows Vista and Windows 2008 users	114
Installing on Windows	114
Installing using the installation wizard (Windows)	115
Installing in console mode (Windows)	117
Installing in silent mode (Windows)	119
Verifying the Netcool/OMNIbus installation (Windows)	123
Viewing the installation log files (Windows)	124
Viewing installed packages (Windows)	124
Installation directory structure (Windows)	125
Upgrading on Windows	127
Guidelines for upgrading to FIPS 140-2 mode (Windows)	128
Upgrading using the installation wizard (Windows)	130
Upgrading in console mode (Windows)	133
Upgrading in silent mode (Windows)	136
Viewing the migration log file (Windows)	140
Modifying your V7.3.1 installation (Windows)	140
Additional upgrade and migration notes (Windows)	141
Notes on upgrading ObjectServer schemas to V7.3.1 schemas (Windows)	141
Files migrated for an upgrade (Windows)	144
Guidelines for upgrading to UTF-8 encoding (Windows)	146
Migrating your digital certificates and keys (Windows)	148
IBM Tivoli Enterprise Console BAROC data migration (Windows)	151
Performing postinstallation tasks (Windows)	155
Configuring settings for online help access (Windows)	156
Configuring the JRE for FIPS 140-2 mode (Windows)	158
Installing probes and gateways into the Tivoli Netcool/OMNIbus environment (Windows)	159
Setting up Tivoli Netcool/OMNIbus components as Windows services	161
Uninstalling Tivoli Netcool/OMNIbus (Windows)	169
Before you uninstall	170
Uninstalling using the wizard (Windows)	170
Uninstalling in console mode (Windows)	171
Uninstalling in silent mode (Windows)	171
Uninstalling probes and gateways (Windows)	172

Chapter 6. Installing, upgrading, and uninstalling the Web GUI component	173
Preparing to install or upgrade the Web GUI	173
Types of Installation	173
Web GUI installation or upgrade prerequisites	174
Obtaining the installation package	174
Gathering installation information	175
Starting the installation launchpad	178
Installing the Web GUI	178
Using the GUI installer	178
Using the console installer	180
Using the silent installer	181
Running the installer in an existing environment	185
Upgrading and migrating to the Web GUI component	185
Upgrade and migration notes	186
Upgrading from IBM Tivoli Netcool/Webtop version 2.2 or Tivoli Netcool/OMNIbus version 7.3.0 Web GUI to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI	192
Migrating from IBM Tivoli Netcool/Webtop versions 2.0 or 2.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI	194
Migrating from IBM Tivoli Netcool/Webtop version 1.3.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI	199
Rolling back migration	202
Netcool GUI Foundation to Tivoli Integrated Portal migration notes	204
Performing post-installation tasks	208
Logging in	208
Protecting the vault key file	210
Assigning Web GUI roles to the administrative user	210
Changing the passwords of the supplied users	212
Setting up the WAAPI client	212
Enabling multicultural support for the Web GUI	213
Uninstalling the Web GUI	214
Using the GUI uninstaller	215
Using the console uninstaller	215
Using the silent uninstaller	217
Troubleshooting installation	218
Viewing the installation log file	218
Viewing installed packages	220
Troubleshooting a failed installation on Solaris operating systems	221
Troubleshooting a failed uninstallation on Windows	221
Troubleshooting user registries	222
Installation fails at step "Integrated Solutions Console"	223
Installation failure scenario	223
Install fails after deployment engine upgrade	225
Migration fails with "Out of Memory" errors	225
Migration from Netcool/Webtop V2.1 or earlier with a large number of users fails	225
Directory structure	226

Chapter 7. Setting up the Tivoli Netcool/OMNIbus system	229
--	------------

Creating and running the ObjectServer	229
ObjectServer overview	229
Configuring automated failover and failback	230
Creating an ObjectServer	231
Starting an ObjectServer	236
Stopping an ObjectServer	238
Configuring server communication details in the Server Editor	240
Creating and maintaining server entries after installation	240
Configuring server communication information	242
Adding a backup ObjectServer	245
Changing the priority of servers	247
Hiding backup ObjectServers in the Server Editor (UNIX only)	247
Testing a server.	247
Manually editing the connections data file.	247
Setting up distributed installations	248
Step 1: Installing Tivoli Netcool/OMNIBus components	248
Step 2: Configuring component communications	248
Step 3: Distributing the interfaces files (UNIX only)	250
Chapter 8. Configuring and deploying a multitiered architecture	251
Before you begin	251
Overview of the standard multitiered architecture	251
Naming conventions for the multitiered architecture	253
Component resourcing: determining the number of ObjectServers needed	255
Severity handling	256
Multitiered configuration file locations	258
Setting up the standard multitiered environment	259
Configuring server communication information (multitiered architecture)	260
Installing the primary aggregation ObjectServer	261
Installing the backup aggregation ObjectServer	262
Configuring the bidirectional aggregation ObjectServer Gateway	263
Installing the primary collection ObjectServer	263
Configuring the unidirectional primary collection ObjectServer Gateway	264
Installing the backup collection ObjectServer	265
Configuring the unidirectional backup collection ObjectServer Gateway	266
Installing the display ObjectServer 1.	266
Configuring the unidirectional display ObjectServer 1 Gateway	267
Installing the display ObjectServer 2.	268
Configuring the unidirectional display ObjectServer 2 Gateway	269
Installing additional ObjectServers	270
Adding a second pair of collection ObjectServers	270
Adding an additional display ObjectServer	275
Automatic load balancing of event list clients.	278
Creating custom triggers	279
The performance triggers	280
Resynchronization Complete synthetic events	282

Final steps	283
Sample omni.dat files.	283

Chapter 9. Configuring high availability. 285

Failover configuration	285
Configuring controlled failback of clients	287
Reducing event loss on ObjectServer failure during resynchronization	288
Reducing resynchronization time.	289
Configuring controlled shutdown of an ObjectServer.	289
Configuring proxy servers for failover	293

Chapter 10. Configuring FIPS 140–2 support for the server components . . 295

Creating the FIPS configuration file	295
Configuring the server components for FIPS 140–2 mode	295
Configuration requirements for connecting V7.2 or earlier clients to V7.2.1 or later servers in FIPS 140–2 mode	298
Switching your installation to FIPS 140-2 mode	299

Chapter 11. Importing and exporting ObjectServer configurations 301

Exporting and importing ObjectServer configurations by using the nco_osreport utility	302
About the nco_osreport utility.	302
Exporting ObjectServer configurations and cloning ObjectServers.	303
Command-line options for the nco_osreport command	304
Exporting and importing ObjectServer configurations using the nco_confpack utility.	305
Import and export terminology	305
Importable and exportable objects	306
nco_confpack properties and command-line options	307
Creating and editing configuration list files	309
Exporting configurations	313
Viewing configuration package contents	320
Importing configurations	320

Chapter 12. Setting up desktop ObjectServers 327

Desktop ObjectServer architecture	327
Considerations for setting up a desktop ObjectServer architecture	329
Configuring a desktop ObjectServer architecture	329
Creating and configuring a desktop ObjectServer.	329
Configuring the unidirectional ObjectServer Gateway	330
Viewing the results of tool actions using dual-write mode	332
Viewing operator journal entries from a dual server desktop	333
Desktop ObjectServer authentication.	333

Load balanced mode	333
Configuring load balanced mode	334

Chapter 13. Tivoli Netcool/OMNIBus user access security 337

User access security mechanisms	337
Authentication	337
Authorization	338
Secure mode authentication	338
ObjectServer secure mode	339
Proxy server secure mode	339
Connecting securely from probes and gateways	339
Process control security	339
SQL interactive interface password protection in scripts	340
LDAP authentication	340
Prerequisite information for LDAP configuration	340
Configuring Tivoli Netcool/OMNIBus to use LDAP for external authentication	341
LDAP properties	343
PAM authentication (UNIX and Linux)	345
Configuring Tivoli Netcool/OMNIBus to use PAM for external authentication	346
Configuring an ObjectServer as a PAM authentication source	348
Implementing authorization by using groups and roles	350
System and object permissions	351
Default Tivoli Netcool/OMNIBus roles	351
Default Tivoli Netcool/OMNIBus groups	353
Default Tivoli Netcool/OMNIBus users	354
Using restriction filters to filter table information	355
Defining and following an audit trail	355
Property value encryption	356
Generating a key in a key file	356
Specifying the key file as a property	357
Encrypting a string value with the key	357
Adding an encrypted value to a properties file	358
nco_aes_crypt command-line options	358

Chapter 14. Quick reference to setting up SSL 361

Chapter 15. Setting up the omni.dat file for SSL 363

Chapter 16. Generating the interfaces file 365

Chapter 17. Generating the certificates 367

Chapter 18. Adding the signer certificate to the client from the server 369

Chapter 19. Configuring the clients with the available CommonNames property 371

Chapter 20. Gateways 373

Chapter 21. Probes 375

Chapter 22. Using SSL for client and server communications 377

Configuring the Server Editor for SSL connections	377
Using the Server Editor to configure SSL on UNIX	378
Using the Server Editor to configure SSL on Windows	378
Generating the interfaces file on UNIX	379
Distributed installations and SSL	379
Managing digital certificates for SSL communication	380
Notes for SSL connections in FIPS 140–2 mode	380
About the key database files	381
Guidelines for certificate management	382
Starting iKeyman	382
Creating a key database	383
Creating a self-signed certificate	385
Extracting certificates from a key database	386
Adding certificates from CAs	387
Requesting a server certificate from a CA	389
Signing a certificate request file with a signer certificate	390
Receiving server certificates from CAs	391
Specifying the default certificate	392
Viewing certificate details	393
Deleting certificates	393
Changing the key database password	394
Managing certificates from the command line	395

Chapter 23. IPv6 configuration 399

Configuring IPv6 support	399
------------------------------------	-----

Chapter 24. Multicultural support 403

Setting your locale	403
Identifying which locales are supported on your computer	406
Enabling or disabling localized sorting	407
Identifying which locales are supported for the UNIX desktop	407
Configuring fonts for the UNIX desktop	407
Setting up the ObjectServer to use translated user interface text in the desktop	410

Chapter 25. Extending the functionality of Tivoli Netcool/OMNIBus 413

Overview of the \$NCHOME/omnibus/extensions directory	413
Enabling predictive eventing and predictive analytics	416
Configuration setup for predictive events	418
Configuration setup for linear trending	420
Prerequisites for predictive eventing and predictive analytics	422

Tivoli Netcool/OMNIbus configuration resources for predictive events	425
Configuring predictive eventing in your integrated environment	428
Configuring linear trending	430
Configuring baselining	434
Enabling support for TADDM events	435
Configuration setup for TADDM events	436
Tivoli Netcool/OMNIbus configuration files for TADDM events.	438
Configuring support for TADDM events in your integrated environment	438
Managing virtualized environments	443
Configuration setup for a virtual environment	444
Tivoli Netcool/OMNIbus configuration files for managing virtualization	446
Configuring event management of a virtual environment.	447
Deploying probes remotely.	451
Prerequisites for remote deployment	452
Workflow for deploying probes remotely	453
Overview of deployable bundles	453
IBM Tivoli Monitoring tacmd commands used for remote deployment	454
Creating deployable bundles with the default configuration	455
Creating deployable bundles with updated configuration	457
Deploying Tivoli Netcool/OMNIbus and probes to remote computers	460
Monitoring the status of your deployments	466
Running remotely-deployed probes	467
Using the file transfer utility (ncf_cftp) to update files	469
Removing probes from remote computers	475
Chapter 26. Configuring the Web GUI component	477
Changing user registries after installation	477
Removing user registries from the federated repository	477
Adding user registries to the federated repository	478
Switching the user registry to which user credentials are written	479
Enabling user synchronization between a federated repository and the ObjectServer	480
Adding an external LDAP repository	481
Enabling LDAP authentication of ObjectServer users	484
Configuring VMM for the ObjectServer.	485
Configuring encryptions.	486
Configuring access for HTTP and HTTPS	486
Configuring an SSL connection to an LDAP server	487
Configuring an SSL connection to the ObjectServer.	488
Configuring SSL connections for the event feed from the ObjectServer	490
Replacing the default SSL certificate for connections to Web GUI clients	491

Encrypting Web GUI passwords	495
Enabling FIPS 140-2 mode for the Web GUI	496
Configuring Tivoli Access Manager in Tivoli Integrated Portal	501
Setting up the Web GUI for productive usage	512
Changing data source configurations	512
Setting environment variables for charts	528
Configuring and maintaining single sign-on	529
Extending the functionality of the Web GUI	532
Setting up and configuring a load balancing environment.	535
Configuring launch-in-context integrations with Tivoli products	557
Setting user access to the Inline Frame portlet	567
Enabling multiple logins.	568
Installing, configuring, and using Tivoli Common Reporting	568
Restarting the server	568

Chapter 27. Example Tivoli Netcool/OMNIbus installation scenarios (basic, failover, and desktop architectures) 571

Example Tivoli Netcool/OMNIbus basic architecture	571
Deploying the basic architecture	571
Prerequisites for the basic architecture	572
Step 1: Installing the ObjectServer and process agent	572
Step 2: Installing the probes	573
Step 3: Installing the event list.	573
Step 4: Configuring communications	573
Step 5: Creating the ObjectServer	574
Step 6: Testing the system	574
Step 7: Installing and configuring the Syslog probe and the Syslog daemon	574
Step 8: Configuring process control	576
Step 9: Adding columns to the ObjectServer	578
Next steps	578
Example Tivoli Netcool/OMNIbus basic failover architecture	579
Deploying the basic failover architecture	579
Prerequisites for the basic failover architecture	580
Step1: Installing the basic architecture	580
Step 2: Installing the backup ObjectServer and ObjectServer Gateway	581
Step 3: Configuring communications	581
Step 4: Creating and configuring the backup ObjectServer.	582
Step 5: Configuring the bidirectional ObjectServer Gateway	582
Step 6: Configuring the Syslog probe	583
Step 7: Configuring process control on the backup computer	583
Next steps	584
Example Tivoli Netcool/OMNIbus desktop ObjectServer architecture	585
Deploying the desktop ObjectServer architecture	585
Prerequisites for the desktop ObjectServer architecture	586

Step1: Installing the basic failover architectures	587	Installation error messages	613
Step 2: Installing the desktop ObjectServer and unidirectional gateway	587	Troubleshooting the Deployment Engine	619
Step 3: Configuring component communications	587	UnknownHostException.	619
Step 4: Creating and configuring the desktop ObjectServer.	588	Uninstalling the Deployment Engine	621
Step 5: Configuring the unidirectional ObjectServer Gateway	588	Troubleshooting security.	623
Step 6: Configuring process control on the desktop ObjectServer computer	590	root access requirements for Tivoli Netcool/OMNIbus processes	623
Next steps	591	nco_pad fails when using PAM authentication on SUSE Linux	624
Chapter 28. Example installation scenario for the non-Web and Web GUI components of Tivoli Netcool/OMNIbus (Windows).	593	User authentication failure with Pluggable Authentication Modules (PAM)	624
Setting up the test environment	593	Logging into the Web GUI after the LDAP server has failed	625
Installing Tivoli Netcool/OMNIbus and setting up the ObjectServer	595	Troubleshooting multicultural support	626
Installing and configuring the Web GUI	598	Appendix C. Deployment Engine command reference	627
Installing the probe	602	Appendix D. Default port numbers used by Tivoli Netcool/OMNIbus	631
Monitoring events in the Active Event List	603	Appendix E. Web GUI initialization file properties	633
Next steps	604	Notices	643
Appendix A. IBM Support Assistant	605	Trademarks	645
Tivoli Netcool/OMNIbus and the Log and Trace Analyzer	607	Index	647
Appendix B. Troubleshooting	613		
Troubleshooting installation	613		

About this publication

Tivoli Netcool/OMNIBus is a service level management (SLM) system that delivers real-time, centralized monitoring of complex networks and IT domains.

The *IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide* describes how to install, upgrade, configure, and use Tivoli Netcool/OMNIBus.

Intended audience

This publication is intended for administrators who need to install and deploy Tivoli Netcool/OMNIBus.

What this publication contains

This publication contains the following sections:

- Chapter 1, “Introduction to Tivoli Netcool/OMNIBus,” on page 1
Provides an overview of Tivoli Netcool/OMNIBus.
- Chapter 3, “Planning for installation or upgrade,” on page 11
Describes the hardware, operating system, software, and installation requirements for Tivoli Netcool/OMNIBus.
- Chapter 4, “Installing, upgrading, and uninstalling (UNIX and Linux),” on page 45
Describes how to install, upgrade, and uninstall the product on UNIX and Linux operating systems.
- Chapter 5, “Installing, upgrading, and uninstalling (Windows),” on page 111
Describes how to install, upgrade, and uninstall the product on Windows operating systems.
- Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173
Describes how to install, upgrade, and uninstall the Web GUI component on all operating systems.
- Chapter 7, “Setting up the Tivoli Netcool/OMNIBus system,” on page 229
Describes how to create and set up one or more ObjectServers and configure communications information for your Tivoli Netcool/OMNIBus components. It also describes how to set up a distributed Tivoli Netcool/OMNIBus installation across your network.
- Chapter 8, “Configuring and deploying a multitiered architecture,” on page 251
Describes how to deploy Tivoli Netcool/OMNIBus within a multitiered architecture that increases performance and event handling capacity.
- Chapter 9, “Configuring high availability,” on page 285
Describes how to configure Tivoli Netcool/OMNIBus for high availability.
- Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295
Describes how to configure Tivoli Netcool/OMNIBus to run in Federal Information Processing Standard 140–2 (FIPS 140–2) mode.
- Chapter 11, “Importing and exporting ObjectServer configurations,” on page 301

Describes how to import and export ObjectServer configurations with the **nco_confpack** utility.

- Chapter 12, “Setting up desktop ObjectServers,” on page 327
Describes how to configure a desktop ObjectServer architecture to reduce the load on ObjectServers that receive high numbers of events.
- Chapter 13, “Tivoli Netcool/OMNIBus user access security,” on page 337
Describes user access security techniques to protect your Tivoli Netcool/OMNIBus system from accidental or deliberate damage caused by users or potential users of your system.
- Chapter 22, “Using SSL for client and server communications,” on page 377
Describes how to configure communications between Tivoli Netcool/OMNIBus servers and clients using the Secure Sockets Layer (SSL) protocol.
- Chapter 23, “IPv6 configuration,” on page 399
Describes how to configure support for IPv6.
- Chapter 24, “Multicultural support,” on page 403
Describes how to configure multicultural support.
- Chapter 25, “Extending the functionality of Tivoli Netcool/OMNIBus,” on page 413
Contains information about the additional resources that you can use to extend the functionality of Tivoli Netcool/OMNIBus.
- Chapter 26, “Configuring the Web GUI component,” on page 477
Describes how to set up the Web GUI component for productive use.
- Chapter 27, “Example Tivoli Netcool/OMNIBus installation scenarios (basic, failover, and desktop architectures),” on page 571
Describes some example Tivoli Netcool/OMNIBus systems. Each example architecture builds on the previous one.
- Chapter 28, “Example installation scenario for the non-Web and Web GUI components of Tivoli Netcool/OMNIBus (Windows),” on page 593
Describes how to perform a basic installation and configuration of the non-Web components and the Web GUI component of Tivoli Netcool/OMNIBus within a Windows test environment.
- Appendix A, “IBM Support Assistant,” on page 605
Describes how you can use the IBM Support Assistant workbench and an accompanying Tivoli Netcool/OMNIBus plug-in to help you resolve questions and problems.
- Appendix B, “Troubleshooting,” on page 613
Contains Tivoli Netcool/OMNIBus troubleshooting tips.
- Appendix D, “Default port numbers used by Tivoli Netcool/OMNIBus,” on page 631
Provides details of the default port numbers defined for Tivoli Netcool/OMNIBus, and describes how to change these default values.
- Appendix E, “Web GUI initialization file properties,” on page 633
Describes the properties of the *webgui_home_dir/etc/server.init* file, which controls the operations of the Tivoli Netcool/OMNIBus Web GUI.
- Web GUI log files
Describes the location and content of the Web GUI log files, which can be used for troubleshooting purposes.

This section lists publications in the Tivoli Netcool/OMNIBus library and related documents. The section also describes how to access Tivoli publications online and how to order Tivoli publications.

Your Tivoli Netcool/OMNIBus library

The following documents are available in the Tivoli Netcool/OMNIBus library:

- *IBM Tivoli Netcool/OMNIBus Installation and Deployment Guide*, SC14-7604
Includes installation and upgrade procedures for Tivoli Netcool/OMNIBus, and describes how to configure security and component communications. The publication also includes examples of Tivoli Netcool/OMNIBus architectures and describes how to implement them.
- *IBM Tivoli Netcool/OMNIBus Administration Guide*, SC14-7605
Describes how to perform administrative tasks using the Tivoli Netcool/OMNIBus Administrator GUI, command-line tools, and process control. The publication also contains descriptions and examples of ObjectServer SQL syntax and automations.
- *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*, SC14-7606
Describes how to perform administrative and event visualization tasks using the Tivoli Netcool/OMNIBus Web GUI.
- *IBM Tivoli Netcool/OMNIBus User's Guide*, SC14-7607
Provides an overview of the desktop tools and describes the operator tasks related to event management using these tools.
- *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*, SC14-7608
Contains introductory and reference information about probes and gateways, including probe rules file syntax and gateway commands.
- *IBM Tivoli Monitoring for Tivoli Netcool/OMNIBus Agent User's Guide*, SC14-7610
Describes how to install the health monitoring agent for Tivoli Netcool/OMNIBus and contains reference information about the agent.
- *IBM Tivoli Netcool/OMNIBus Event Integration Facility Reference*, SC14-7611
Describes how to develop event adapters that are tailored to your network environment and the specific needs of your enterprise. This publication also describes how to filter events at the source.
- *IBM Tivoli Netcool/OMNIBus Error Messages Guide*, SC14-7612
Describes system messages in Tivoli Netcool/OMNIBus and how to respond to those messages.
- *IBM Tivoli Netcool/OMNIBus Web GUI Administration API (WAAPI) User's Guide*, SC22-5403-00
Shows how to administer the Tivoli Netcool/OMNIBus Web GUI using the XML application programming interface named WAAPI.

Accessing terminology online

The *Tivoli Software Glossary* includes definitions for many of the technical terms related to Tivoli software. The *Tivoli Software Glossary* is available at the following Tivoli software library Web site:

<http://publib.boulder.ibm.com/tividd/glossary/tivoliglossarymst.htm>

The IBM Terminology Web site consolidates the terminology from IBM product libraries in one convenient location. You can access the Terminology Web site at the following Web address:

<http://www.ibm.com/software/globalization/terminology>

Accessing publications online

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli Information Center Web site at:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp>

Note: If you print PDF documents on other than letter-sized paper, set the option in the **File > Print** window that allows Adobe Reader to print letter-sized pages on your local paper.

Ordering publications

You can order many Tivoli publications online at the following Web site:

<http://www.elink.ibm.link.ibm.com/publications/servlet/pbi.wss>

You can also order by telephone by calling one of these numbers:

- In the United States: 800-879-2755
- In Canada: 800-426-4968

In other countries, contact your software account representative to order Tivoli publications. To locate the telephone number of your local representative, perform the following steps:

1. Go to the following Web site:
<http://www.elink.ibm.link.ibm.com/publications/servlet/pbi.wss>
2. Select your country from the list and click **Go**. The Welcome to the IBM Publications Center page is displayed for your country.
3. On the left side of the page, click **About this site** to see an information page that includes the telephone number of your local representative.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully.

With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate some features of the graphical user interface.

Tivoli technical training

For Tivoli technical training information, refer to the following IBM Tivoli Education Web site:

<http://www.ibm.com/software/tivoli/education>

Support information

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

Online

Go to the IBM Software Support site at <http://www.ibm.com/software/support/probsub.html> and follow the instructions.

IBM Support Assistant

The IBM Support Assistant (ISA) is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. The ISA provides quick access to support-related information and serviceability tools for problem determination. To install the ISA software, go to <http://www.ibm.com/software/support/isa>.

Related reference

Appendix A, "IBM Support Assistant," on page 605

Conventions used in this publication

This publication uses several conventions for special terms and actions and operating system-dependent commands and paths.

Typeface conventions

This publication uses the following typeface conventions:

Bold

- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:** and **Operating system considerations:**)
- Keywords and parameters in text

Italic

- Citations (examples: titles of publications, diskettes, and CDs)
- Words defined in text (example: a nonswitched line is called a *point-to-point* line)
- Emphasis of words and letters (words as words example: "Use the word *that* to introduce a restrictive clause."; letters as letters example: "The LUN address must start with the letter *L*.")
- New terms in text (except in a definition list): a *view* is a frame in a workspace that contains data
- Variables and values you must provide: ... where *myname* represents....

Monospace

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

Operating system-dependent variables and paths

This publication uses the UNIX convention for specifying environment variables and for directory notation.

When using the Windows command line, replace *\$variable* with *%variable%* for environment variables, and replace each forward slash (/) with a backslash (\) in directory paths. For example, on UNIX systems, the *\$NCHOME* environment variable specifies the path of the Netcool® home directory. On Windows systems, the *%NCHOME%* environment variable specifies the path of the Netcool home directory. The names of environment variables are not always the same in the Windows and UNIX environments. For example, *%TEMP%* in Windows environments is equivalent to *\$TMPDIR* in UNIX environments.

If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Operating system-specific directory names

Where Tivoli Netcool/OMNIbus files are identified as located within an *arch* directory under *NCHOME*, *arch* is a variable that represents your operating system directory, as shown in the following table.

Table 1. Directory names for the arch variable

Directory name represented by <i>arch</i>	Operating system
<i>aix5</i>	AIX® systems
<i>hpux11</i>	HP-UX PA-RISC-based systems
<i>hpux11hpie</i>	HP-UX Integrity-based systems
<i>linux2x86</i>	Red Hat Linux and SUSE systems
<i>linux2s390</i>	Linux for System z®
<i>solaris2</i>	Solaris systems
<i>win32</i>	Windows systems

Chapter 1. Introduction to Tivoli Netcool/OMNIbus

Tivoli Netcool/OMNIbus is a service level management (SLM) system that delivers real-time, centralized monitoring of complex networks and IT domains.

This information can then be:

- Assigned to operators
- Passed to helpdesk systems
- Logged in a database
- Replicated on a remote Tivoli Netcool/OMNIbus system
- Used to trigger automatic responses to certain events

Tivoli Netcool/OMNIbus can also consolidate information from different domain-limited network management platforms in remote locations. By working in conjunction with existing management systems and applications, Tivoli Netcool/OMNIbus minimizes deployment time and enables employees to use their existing network management skills.

Tivoli Netcool/OMNIbus tracks alert information in a high-performance, in-memory database, and presents information of interest to specific users through filters and views that can be configured individually. Tivoli Netcool/OMNIbus has automation functions that can perform intelligent processing on managed alerts.

Tivoli Netcool/OMNIbus components

The Tivoli Netcool/OMNIbus components work together to collect and manage network event information.

The components of Tivoli Netcool/OMNIbus are:

- The ObjectServer
- Probes
- Gateways
- Desktop tools
- Administration tools
- The Web GUI visualization component

The following figure shows an overview of the Tivoli Netcool/OMNIbus component architecture. Probes send alerts to the local ObjectServer, and a gateway replicates these alerts in an additional ObjectServer in a failover configuration. Alerts that are sent to the ObjectServer can be viewed in the Active Event List in the Web GUI, or in the desktop event list. Additional gateways are also configured to forward alerts to other applications, such as a helpdesk or Customer Relationship Management (CRM) system, and a relational database management system (RDBMS). Netcool/OMNIbus Administrator (and the other administration tools) can also be used to configure and manage the system.

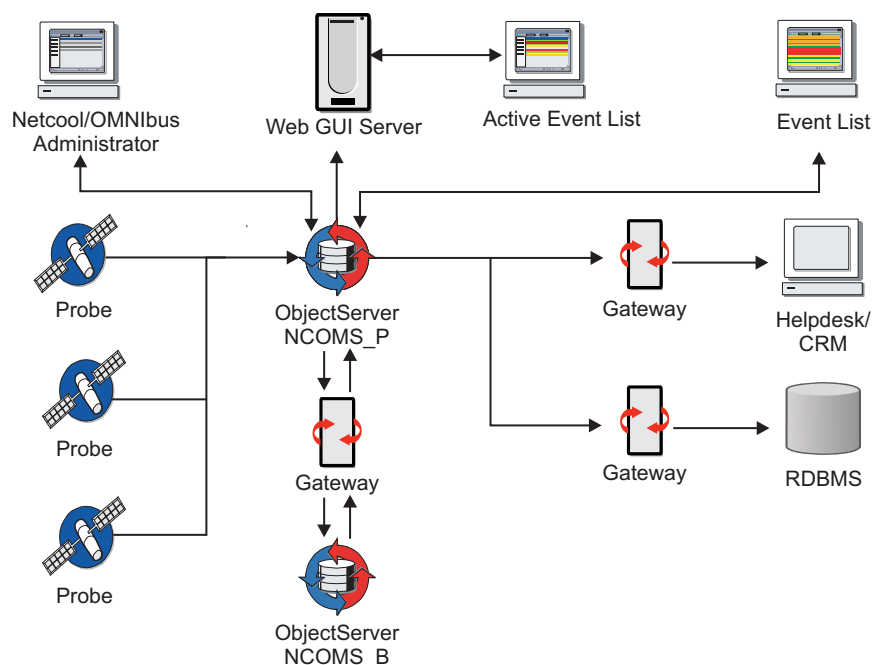


Figure 1. Tivoli Netcool/OMNIBus component architecture

The ObjectServer

The ObjectServer is the in-memory database server at the core of Tivoli Netcool/OMNIBus.

Event information is forwarded to the ObjectServer from external programs such as probes and gateways. This information is stored and managed in database tables, and displayed in the Active Event List (AEL) in the Web GUI, or in the desktop event list.

Deduplication and automation in the ObjectServer

A single device might generate the same error repeatedly until the problem is dealt with. The ObjectServer uses *deduplication* to ensure that event information generated from the same source is not duplicated in the event list. Repeated events are identified and stored as a single event to reduce the amount of data in the ObjectServer. The ObjectServer maintains a count (or tally) of the total number of recurrences of that event.

You can use *automation* to detect changes in the ObjectServer and generate automated responses to these changes. This enables the ObjectServer to process alerts without requiring an operator to take action.

Related tasks

“Creating and running the ObjectServer” on page 229

Probes

Probes connect to an event source, detect and acquire event data, and forward the data to the ObjectServer as events.

Probes use the logic specified in a rules file to manipulate the event elements before converting them into fields of an event in the ObjectServer alerts.status table.

Each probe is uniquely designed to acquire event data from a specific source. Probes can acquire data from any stable data source, including devices, databases, and log files. Probes can also be configured to modify and add to the event data.

Gateways

Tivoli Netcool/OMNIbus gateways enable the exchange of events between ObjectServers and complementary third-party applications, such as databases, and helpdesk or Customer Relationship Management (CRM) systems.

You can use gateways to replicate events or to maintain a backup ObjectServer. Application gateways enable you to integrate different business functions. For example, you can configure a gateway to send event information to a helpdesk system. You can also use a gateway to archive events to a database.

After a gateway is correctly installed and configured, the transfer of events is transparent to operators.

Desktop tools

The desktop is an integrated suite of graphical tools used to view and manage events, and to configure how event information is presented.

Event information is delivered in a format that you can use to quickly determine the availability of services on a network. When an event cause has been identified, you can use the desktop tools to resolve problems quickly.

Most of the features of the desktop are also available in the Web GUI component.

Restriction: Desktop tools are not supported on Linux on System z or HP-UX Integrity systems. You can install and configure the Web GUI component, and then use Web GUI clients to view, manage, and configure events in a similar manner to the desktop tools.

Related concepts

“The Web GUI component” on page 4

Administration tools

Tivoli Netcool/OMNIbus includes tools that administrators can use to configure and manage the system.

The Tivoli Netcool/OMNIbus tools include the Netcool/OMNIbus Administrator, an SQL interactive interface, an import and export utility, and process control.

Netcool/OMNIbus Administrator

Netcool/OMNIbus Administrator is a graphical tool that you can use to configure and manage ObjectServers.

SQL interactive interface

The ObjectServer provides a Structured Query Language (SQL) interface for defining and manipulating relational database objects such as tables and views. You can use the SQL interactive interface (called **nco_sql** on UNIX and Linux, and **isql** on Windows) to connect to an ObjectServer, and use SQL commands to interact with, and control, the ObjectServer. The SQL interactive interface enables you to perform tasks such as creating a new database table or stopping the ObjectServer.

Import and export utility

You can use the Confpack utility (**nco_confpack**) to:

- Import and export Tivoli Netcool/OMNIBus ObjectServer configurations to deploy duplicate Tivoli Netcool/OMNIBus systems in your network
- Extract a subset of configuration items from Tivoli Netcool/OMNIBus ObjectServers (for example, event list menus and automations) and import them into other ObjectServers
- Save Tivoli Netcool/OMNIBus ObjectServer configuration data for backup purposes

Process control

The process control system performs two primary tasks:

- It runs external procedures that are specified in automations. Automations detect changes in the ObjectServer and run automated responses to those changes.
- It manages local and remote processes.

Use process control to configure remote processes in order to simplify the management of Tivoli Netcool/OMNIBus components such as ObjectServers, probes, and gateways. The process control system consists of:

- Process agents, which are programs installed on each host for managing processes
- A set of command-line utilities that provide an interface to process management

Related concepts

Chapter 11, “Importing and exporting ObjectServer configurations,” on page 301

The Web GUI component

The Tivoli Netcool/OMNIBus Web GUI is a Web-based application that processes network events from one or more data sources and presents the event data to users in various graphical formats in a Web browser. Certain data can also be displayed on supported mobile devices.

The Web GUI contains most features of the Tivoli Netcool/OMNIBus native desktop environment. The Web GUI extends the event visualization and management capabilities of Tivoli Netcool/OMNIBus by being both highly configurable and remotely accessible through the Internet.

The Web GUI uses a client-server architecture. The Web GUI server runs inside Tivoli Integrated Portal. Clients connect to Tivoli Integrated Portal to access the Web GUI.

The Web GUI can be configured for integrations with other Tivoli products.

Related concepts

“Desktop tools” on page 3

Features of the Web GUI

A number of different clients can be connected to the Web GUI server. The actions that the clients can conduct after they have logged in are dependent on their client type and user roles. The Web GUI has user roles associated with standard client types.

The Web GUI can simultaneously connect to more than one ObjectServer, and centralize this information on one page. This enables end users to view all relevant alerts (for example, high severity alerts) by examining the different customized displays associated with each ObjectServer.

The following figure shows how the features of the Web GUI interact.

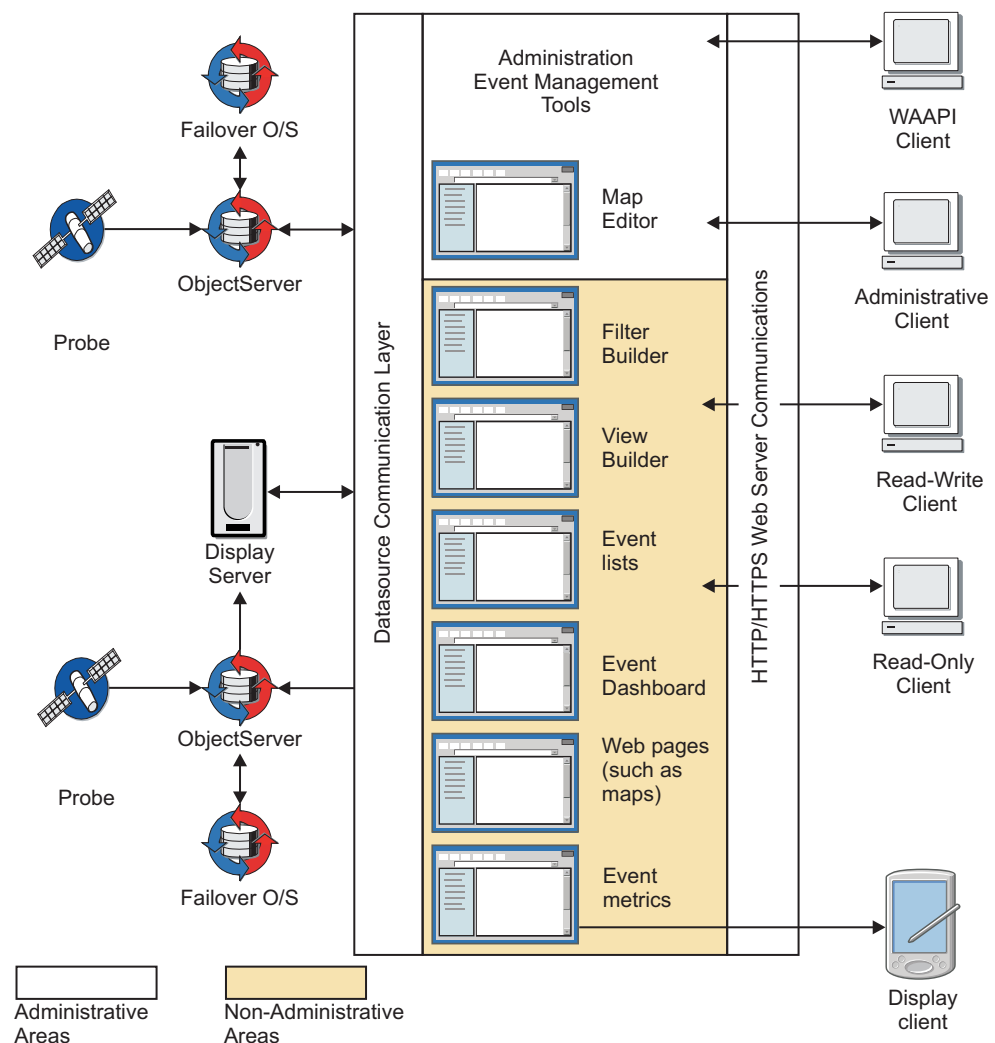


Figure 2. Web GUI communications

The main event display components are as follows:

Event lists

The Web GUI provides the following event lists:

Active Event List (AEL)

A Java applet that runs native desktop actions including acknowledging alerts, viewing alert journals, taking ownership of alerts, and running event management tools.

Lightweight Event List (LEL)

A dynamic HTML event list that provides the data filtering, data sorting, and information drill-down capabilities of the AEL without the event management tools.

Table View

A static HTML event list in the form of a table showing a defined set of alerts. The Table View provides an immediate snapshot of alert status within a monitored system.

The Web GUI provides the following components for configuring a high-level overview of events:

Event Dashboard

An overview of alert information as captured by filters. The Event Dashboard presents the alert information as a series of monitor boxes, from which you can open AELs.

Gauges

An overview of information from Tivoli Netcool/OMNIBus captured by *metrics*. A metric is a type of measurement that is used to determine a quantifiable value from tables or properties in the ObjectServer. The information is displayed in a series of gauges, and can be viewed in a Web browser, or published via a URL to a supported mobile device. Recipients can bookmark the URL so that they can return to the gauges. Click-actions can be associated with the gauges to enable users to drill down into the information.

Maps Administrators can use maps to design visual representations of a network and to create interactive graphical views network performance. The maps can be used to help operators monitor the events occurring on the network.

Chart rendering component

Administrators can create charts that present high-level network alert information to users in a number of graphical formats including bar charts and pie charts.

The Web GUI Administration Application Program Interface (WAAPI) client is a Java-based utility that you can use to remotely administer the Web GUI server.

Tivoli Integrated Portal overview

Web-based products built on the Tivoli Integrated Portal framework share a common user interface where you can launch applications and share information.

Tivoli Integrated Portal helps the interaction and secure passing of data between Tivoli® products through a common portal. You can launch from one application to another and within the same dashboard view research different aspects of your managed enterprise.

Tivoli Integrated Portal is installed automatically with the first Tivoli product using the Tivoli Integrated Portal framework. Subsequent products may install updated versions of Tivoli Integrated Portal.

Tivoli Integrated Portal provides the following features:

- A Web based user interface for individual products and for integrating multiple products.
- A single, task-based navigation panel for multiple products. Users select actions based around the task that they want to complete, not by the product that supports that task.
- Single sign-on (SSO), consolidated user management, and a single point of access for different Tivoli applications.
- Aggregated views that span server instances, such as the Tivoli Netcool/OMNIBus ObjectServer and Tivoli Enterprise Portal Server.
- Inter-view messaging between products to support contextual linkage between applications.
- The ability to create customized pages and administer access to content by user, role, or group.

Related reference

“Tivoli Integrated Portal components” on page 38

Chapter 2. Backing up and restoring Tivoli Netcool/OMNIBus

To prevent the loss of information in the during installation, upgrade, application of a fix pack, or in the event of a disaster, and for recovery of that information, back up your installation of Tivoli Netcool/OMNIBus.

Perform this task:

- If you are installing Tivoli Netcool/OMNIBus or the Web GUI component on a server on which the Deployment Engine (DE) is currently installed, for example, a server that currently hosts another DE-based product.
- If you are upgrading Tivoli Netcool/OMNIBus or the Web GUI.
- If you are applying a fix pack
- During routine system administration and troubleshooting.

You do not need to perform this task if you are installing the product on a clean server.

Note: The following steps are high-level. For detailed instructions on each step, see the links in the “Related” sections. For more information about backing up the Web GUI, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

To back up and restore Tivoli Netcool/OMNIBus:

- To back up the product:
 1. Back up the DE.
 2. To back up the server-side components copy the \$NCHOME directory to a secure location.
 3. To back up the Web GUI configuration data, export the required data from the Web GUI server. The selected data is written to a .zip file that you can copy to a secure location.
- To restore the product:
 1. Restore the DE.
 2. To restore the server-side components, remove or rename the existing \$NCHOME directory, and then replace them with the previously backed up \$NCHOME directory.
 3. To restore the Web GUI, import the previously exported data

Chapter 3. Planning for installation or upgrade

Before installing or upgrading, read about the hardware, operating system, software, and communication requirements for Tivoli Netcool/OMNIBus. Review compatibility with previous versions, and learn about the installation modes and the common installation directory structure for the Network Management portfolio of products.

Tip: For information about sizings for your Tivoli Netcool/OMNIBus deployment, see the *OMNIBus Sizing Guide*, which is available on the IBM Service Management Connect Web site. To obtain this guide, go to <https://www.ibm.com/developerworks/servicemanagement/> and search for *OMNIBus Sizing Guide*.

Tip: IBM provides an unsupported utility, the IBM Tivoli Prerequisite Scanner, that verifies that the required hardware and software for Tivoli Netcool/OMNIBus, excluding the Web GUI component, are present on the host. Note that the IBM Tivoli Prerequisite Scanner is not supported by IBM for use with Tivoli Netcool/OMNIBus, nor is it supplied with the Tivoli Netcool/OMNIBus product. For more information about the IBM Tivoli Prerequisite Scanner, see <http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg21472859>. The IBM Tivoli Prerequisite Scanner is available for download at IBM Service Management Library, at <http://www-01.ibm.com/software/brandcatalog/ismlibrary/>. Search the site for *Prerequisite Scanner* and then log in with your IBM and password to start the download

Supported operating systems and compatible products

Tivoli Netcool/OMNIBus is supported on various versions of UNIX, Linux, and Windows. Use this information to find out which components of the product are supported on which operating systems, and with which Tivoli products and versions you can integrate Tivoli Netcool/OMNIBus.

For the latest information about supported operating systems and compatible products, see <http://www.ibm.com/support/docview.wss?rs=3120&uid=swg21468519>.

Prerequisites for operating systems

Before you install Tivoli Netcool/OMNIBus on your chosen supported operating system, ensure that it meets these requirements.

For the most current information about supported operating systems, see the detailed system requirements document at:

<http://www.ibm.com/support/docview.wss?rs=203&uid=swg21067036>

Important: Ensure that your operating system has all the recommended patches installed, including the latest patch levels.

Requirements for Windows

If installing Tivoli Netcool/OMNIbus on Windows operating systems, Microsoft Windows Installer 3.0, 3.1, or later versions are required on your system before the installation.

Requirements for Linux

On 64-bit Linux systems, Tivoli Netcool/OMNIbus runs in toleration mode. Therefore, you must install the 32-bit versions of the RPM packages on your system. On RHEL 6, no 32-bit versions of the packages are installed by default, so make sure that you have the 32-bit versions of the required packages.

The following table describes the RPM packages required for Linux operating systems.

Table 2. Additional considerations and requirements for Linux operating systems

Linux version	Minimum required RPMs	Explanation and comments
Red Hat Enterprise Linux (RHEL) AS, ES, and WS 5.	<ul style="list-style-type: none">• libXp-1.0.0-8 (required by the native desktop)• openmotif22-2.2.3-18 (required by the native desktop)• libgcc-4.1.2• libXmu-1.0.2-5• libXpm-3.5.5-3• compat-libstdc++-296-2.96-138 (for 32-bit and 64-bit versions of x86 only)• compat-libstdc++-33-3.2.3-61	<p>Install the compat-libstdc++33-3.2.3-61 package before installing the Web GUI component.</p> <p>For more information about obtaining the package, go to the IBM® WebSphere® Application Server Information Center at http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp and search for <i>compat-libstdc++-33-3.2.3-61</i>.</p>

Table 2. Additional considerations and requirements for Linux operating systems (continued)

Linux version	Minimum required RPMs	Explanation and comments
RHEL AS, ES, and WS 6	<ul style="list-style-type: none"> • libXp-1.0.0-8 (required by the native desktop) • openmotif22-2.2.3-18 (required by the native desktop) • pam-1.1.1-4 • libstdc++-4.4.4-13 • libXft • libXtst • libgcc-4.4.4-13 • libXmu-1.0.5 • libXpm-3.5.8 • compat-libstdc++-296-2.96-144 • compat-libstdc++-33-3.2.3-69 • compat-libstdc++-33-3.2.3-47.3 • compat-libstdc++-33.i686 	
SuSE Linux Enterprise Server (SLES) 32-bit	<ul style="list-style-type: none"> • openmotif-libs-2.2.4 (required for the native desktop) • compat-libstdc++-5.0.7-22.2 	
SLES 64-bit	<ul style="list-style-type: none"> • openmotif-libs-32bit-2.2.4 (required for the native desktop) • compat-libstdc++-5.0.7-22.2 • libgcc-4.1.2 	

Also note that the desktop event list is not supported on Linux S390.

Requirements for Linux on System z

If installing Tivoli Netcool/OMNIbus on RHEL AS, ES and WS 5 or 6 on System z operating systems, the following operating system packages must be available on your system before the installation:

- `compat-libstdc++-295-2.95.3-85`
- `libX11`
- `libXtst`

The packages for RHEL AS, ES, and WS 5 and 6 must also be installed on your system.

Requirements for HP-UX Itanium

The desktop event list is not supported on HP-UX Itanium.

JRE requirements

The Netcool/OMNIbus Administrator GUI, Confpack utility (**nco_confpack**), and Accelerated Event Notification component require the Java Runtime Environment (JRE) to be installed on your system.

The IBM JRE 6.0 is included in the installation bundle for all operating systems, and provides support for the Federal Information Processing Standard 140-2 (FIPS 140-2).

Related concepts

“Federal Information Processing Standard 140–2 (FIPS 140–2) support” on page 26

“Installation or upgrade prerequisites (UNIX and Linux)” on page 47

“Installation or upgrade prerequisites (Windows)” on page 113

Web browsers, JREs, and mobile devices for the Web GUI

Web GUI pages are displayed within a number of supported Web browsers on client workstations, which require the Java Virtual Machine (JVM) plug-in to be installed. The JVM plug-in is available as part of the Java Runtime Environment.

Web browser support and JRE support are provided on a subset of the supported server-side operating systems, with the exception of support for Apple on the client side. The following table shows the Web browsers, operating systems, and JRE versions that are supported on Web GUI client workstations. JavaScript must be enabled for all browsers.

Table 3. Supported Web browsers, operating systems, and JRE versions for Web GUI clients

Web browser	Version	Operating system	JVM version (in priority order)
Internet Explorer	7.0, 8.0	Windows 2003, Windows XP, Windows Vista, Windows 2008, and Windows 7	Oracle JRE 6 IBM JRE 6.0 Oracle JRE 5 IBM JRE 5.0

Table 3. Supported Web browsers, operating systems, and JRE versions for Web GUI clients (continued)

Web browser	Version	Operating system	JVM version (in priority order)
Mozilla Firefox	3.5, 3.6	Windows 2003, Windows XP, Windows Vista, Windows 2008 , and Windows 7 SuSE Linux Enterprise Server (SLES) 10, and 11 Red Hat Enterprise Linux (RHEL) 4, 5 and 6 Solaris 9 and 10	Oracle JRE 6 IBM JRE 6.0 (not Solaris) Oracle JRE 5 IBM JRE 5.0 (not Solaris)

Note: You must configure your client Web browsers to accept all cookies.

Mobile devices

To display a Gauges page on a mobile device, the following prerequisites must be met:

- The device must be a smartphone.
- You must have enabled JavaScript for the browser.
- The screen must have a minimum resolution of 320 x 240 pixels.

With this screen resolution, the Gauges page displays two columns of gauges, as shown in the following figure:



Figure 3. Example of the Gauges page on a mobile device

The Gauges page has been demonstrated on smartphones that run on the Blackberry 4.6+ and iOS 4.0+ operating systems.

The drill-down function of the Gauges page uses URL launch actions. If you experience problems with this function on a mobile device, ensure that the browser supports JavaScript. Also ensure that you have not defined any JavaScript actions that might affect the function.

User interface requirements

Tivoli Netcool/OMNIBus supports `openmotif 2.2.4` or higher, or CDE, and Microsoft Windows 2003, 2008, XP, Vista Business edition, and 7.

Online help requirements

The online help for Tivoli Netcool/OMNIBus is deployed using IBM Eclipse Help System (IEHS), which is a Web application. Tivoli Netcool/OMNIBus supports IEHS V3.1.1.

Online help is displayed in a Web browser, and is available in standalone and information center modes. Standalone mode is the default.

In standalone mode, the IEHS application framework and online help files run on a local Web server, and **Local Help System** must be selected as an installable feature in order to obtain the required IEHS components. After installation, you can access the online help, generally without the need for any additional configuration. When you try to access online help, a Web server automatically starts and runs locally until you manually shut it down.

In information center mode, the IEHS application framework and online help files run on a remote Web server that users must connect to in order to access online help. The use of a shared online help server relieves the load on local hosts or workstations. A system administrator must take responsibility for installing, configuring, and managing the IEHS components on this server. The **Local Help System** feature must be installed on this server to obtain the IEHS application framework and online help files. To access the online help on the remote server, users must amend a local IEHS configuration file with the connection details for the remote server. The IEHS configuration file is installed, by default, as part of the standard Tivoli Netcool/OMNIBus installation. The system administrator must then run an IEHS startup script to start the IEHS server and open the connection to users.

For the Tivoli Netcool/OMNIBus desktop component (which is typically accessed by using the Conductor), the default operating system browser is used to display online help in either standalone mode or information center mode. If using Netcool/OMNIBus Administrator or the Accelerated Event Notification client, you must specify a browser for displaying online help.

Supported browsers for IEHS V3.1.1

You might experience limited functionality of your IEHS online help if your browser does not fully support cookies or JavaScript, or if your browser blocks pop-up windows. On some operating systems, certain browsers might be capable of displaying the help system interface in a base mode only. In this mode, only the basic functions of IEHS are available, such as content display and the search function.

The minimum browser versions supported for IEHS V3.1.1 are as follows:

- Internet Explorer 6.0

Note: If Security Settings in Internet Explorer are configured such that the **Allow META REFRESH** option is set to Disable, the help index page might not display. In such a case, set the value of the option to Enable.

- Mozilla 1.7
- Firefox 1.0
- Safari 1.2
- Konqueror (user interface base mode only)

Note: The supported Web browsers that are listed here for IEHS differ from the Web browsers that are supported for the Web GUI.

Related concepts

“Web browsers, JREs, and mobile devices for the Web GUI” on page 14

Related tasks

“Configuring settings for online help access (UNIX and Linux)” on page 100

“Configuring settings for online help access (Windows)” on page 156

Disk space requirements

Ensure that there is enough disk space available on the volume for the operating system on which you are installing Tivoli Netcool/OMNIbus.

Non-Web components

The following table shows the installation disk space required on each operating system. These figures are based on the assumption that a full installation of the features is performed, with the following specifications:

- A few probe installations (approximately four)
- ObjectServer gateways only
- Very small ObjectServer databases with 50-100 events

Table 4. Installation disk space

Operating system	Space required for complete installation
AIX	650 MB
HP-UX	720 MB
HP-UX Integrity	800 MB
Linux	635 MB
Linux on System z	630 MB
Solaris	680 MB
Windows	560 MB

In addition, when installing Tivoli Netcool/OMNIbus on a UNIX operating system, the following disk space must also be available:

- The temp location in /tmp requires 310 MB free space.
- For installation as root user, the common directory in /usr/ibm/common requires 250 MB free space.
- If you are planning to install as a non-root user, the home directory in /home/username requires 240 MB free space.

Web GUI component

A full installation of the Web GUI requires a *minimum* of 2 GB of disk space, and a *minimum* of 1 GB of system memory.

Depending on your installation, you might require at least 2 GB of additional disk space and 2 GB of system memory.

The Web GUI installation directory requires 1 GB of disk space. The temp location in /tmp or C:\temp requires 500 MB free space. If you are planning to install as a non-root user, the home directory in /home/*username* requires 300 MB free space. At the time of installation, the Deployment Engine (DE) directories require 300 MB free space. However, the amount of space required by the DE can increase over time, for example as other Tivoli products are installed on the server.

In addition, you must ensure that you have adequate swap space available on the Web GUI server.

Related concepts

“Installation or upgrade prerequisites (UNIX and Linux)” on page 47

“Installation or upgrade prerequisites (Windows)” on page 113

“Web GUI installation or upgrade prerequisites” on page 174

Networking protocol support

Tivoli Netcool/OMNIbus supports communication over IPv4 and IPv6 networks.

Tivoli Netcool/OMNIbus supports the following configurations for IPv4 and IPv6:

- Installs and runs within a network running in an IPv4 only environment
- Installs and runs within a network running in an IPv6 only environment
- Maintains interoperability with IPv4, such that the product can operate and coexist on a network supporting IPv4 only, IPv6 only, or a dual IPv4 and IPv6 configuration

A dual IPv4 and IPv6 environment can be defined as one in which both protocols can be used simultaneously.

- Supports the processing of events that are generated within both IPv4 and IPv6 network environments within a single ObjectServer and desktop instance
- Operates and coexists on a network supporting IPv4 only, IPv6 only, or a hybrid of IPv4 and IPv6 on all its supported operating systems

IPv6 restrictions

The peer-to-peer functionality of probes is available in a dual IPv4 and IPv6 environment. The V7, or later, ObjectServer gateways (**nco_g_objserv_bi** and **nco_g_objserv_uni**) can also connect to an ObjectServer using IPv6 or IPv4. For details of individual probe and gateway support for IPv6, see the documentation supplied with each probe and gateway.

The IBM Eclipse Help System (IEHS) V3.1.1, which is used for the delivery of online help, does not support IPv6. Therefore, when configuring an IEHS server to use in IEHS information center mode, do not specify an IPv6 address for accessing the server.

Internet Explorer version 6 does not support the use of literal IPv6 addresses in URLs. On Web GUI client workstations, you must use host names instead.

Related concepts

Chapter 23, “IPv6 configuration,” on page 399

Related tasks

“Configuring settings for online help access (UNIX and Linux)” on page 100

“Configuring settings for online help access (Windows)” on page 156

Communication protocol

Tivoli Netcool/OMNIBus components use client/server technology communicating over a TCP/IP network.

You can install the components on a single system or in a distributed environment. For example, it might be appropriate to install a probe on the same system as its event source, while the ObjectServer and desktop are installed on other systems in the network.

The ObjectServer, proxy server, gateways, and process control must be configured to use the Tivoli Netcool/OMNIBus communications protocol.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Compatibility with previous versions

Tivoli Netcool/OMNIBus V7.3.1 probe, gateway, process control, Web GUI, and desktop components are compatible with the components of Tivoli Netcool/OMNIBus versions 7.1, 7.2, 7.2.1, and 7.3. Any exceptions (and workarounds) that apply are documented here.

Probe and gateway dependencies

Ensure that you download the latest version of probe and gateway components for use with V7.3.1. Do not use the V7 probe bundle **PINSTALL** function.

Any dependency patches that are required for probes and gateways are documented in the README.txt and description.txt files that are available with the download packages. This information is also available in the individual probe and gateway publications in the Tivoli Netcool/OMNIBus Information Center at: http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.tivoli.nam.doc/welcome_ob.htm.

Compatibility of ObjectServer Gateways

The Tivoli Netcool/OMNIBus V7.3.1 ObjectServer Gateways contain additional gateway mappings, which are not available in Tivoli Netcool/OMNIBus V7.2.1, or earlier, and therefore cannot be replicated. To use a Tivoli Netcool/OMNIBus V7.3.1 ObjectServer Gateway with an earlier version of the ObjectServer, you must comment out some of the entries in the table replication definition file and map definition file.

For the unidirectional gateway:

- Edit the \$NCHOME/omnibus/gates/objserv_uni/objserv_uni.reader.tblrep.def file by commenting out the following lines, as shown:

```
# REPLICATE ALL FROM TABLE 'iduc_system.iduc_stats'  
# USING map 'IducMap';
```

- Edit the \$NCHOME/omnibus/gates/objserv_uni/objserv_uni.map file by commenting out the following lines, as shown:

In the CREATE MAPPING StatusMap section:

```
# 'ProbeSubSecondId' = '@ProbeSubSecondId',
# 'BSM_Identity' = '@BSM_Identity'
```

Further on in the file:

```
# CREATE MAPPING IducMap
# (
#   'ServerName' = '@ServerName' ON INSERT ONLY,
#   'AppName' = '@AppName',
#   'AppDesc' = '@AppDesc' ON INSERT ONLY,
#   'ConnectionId' = '@ConnectionId' ON INSERT ONLY,
#   'LastIducTime' = '@LastIducTime'
# );
```

For the bidirectional gateway:

- Edit both the \$NCHOME/omnibus/gates/objserv_bi/objserv_bi.objectservera.tblrep.def and \$NCHOME/omnibus/gates/objserv_bi/objserv_bi.objectserverb.tblrep.def files by commenting out the following lines, as shown:

```
# REPLICATE ALL FROM TABLE 'iduc_system.iduc_stats'
# USING map 'IducMap';
```

- Edit the \$NCHOME/omnibus/gates/objserv_bi/objserv_bi.map file by commenting out the following lines, as shown:

In the CREATE MAPPING StatusMap section:

```
# 'ProbeSubSecondId' = '@ProbeSubSecondId',
# 'BSM_Identity' = '@BSM_Identity'
```

Further on in the file:

```
# CREATE MAPPING IducMap
# (
#   'ServerName' = '@ServerName' ON INSERT ONLY,
#   'AppName' = '@AppName',
#   'AppDesc' = '@AppDesc' ON INSERT ONLY,
#   'ConnectionId' = '@ConnectionId' ON INSERT ONLY,
#   'LastIducTime' = '@LastIducTime'
# );
```

Process control compatibility

The V7.3.1 Windows process agent cannot communicate with a V7.2.0, or earlier, Windows process agent. The V7.3.1 Windows process agent also cannot communicate with V7.2.0, or earlier ObjectServers.

nco_postmsg

The **nco_postmsg** utility is compatible with ObjectServer versions 7.1, or later. You can install this utility and then use it to connect to, and send events to, ObjectServer versions 7.1, or later.

Tivoli Event Integration Facility (EIF) toolkit compatibility

The EIF is compatible with earlier versions (such as Tivoli Enterprise Console-based senders and receivers) only when the SOCKET transport type is used. The following conditions apply:

- A new EIF sender cannot send events to the Tivoli Enterprise Console® server by using the SSL transport. However, a new sender can send events to the Tivoli Enterprise Console server by using the SOCKET transport.
- A new EIF receiver cannot receive events from Tivoli Enterprise Console adapters over the SSL transport. However, a new sender can receive events from Tivoli Enterprise Console adapters over the SOCKET transport.
- The Tivoli Enterprise Console adapters and the Tivoli Enterprise Console server are not linked to the new version of the EIF libraries.
- A new EIF receiver can receive events from Tivoli Enterprise Console adapters over IPv4 or IPv6.
- A Probe for Tivoli EIF without the EIF updates cannot receive events over the SSL transport.
- A Probe for Tivoli EIF without the EIF updates can receive events from new EIF senders over IPv6 because the Java implementation already supports IPv6 through the JVM.
- A Probe for Tivoli EIF with the EIF updates can receive events sent over IPv4 from earlier EIF senders such as IBM Tivoli Monitoring and Tivoli Enterprise Console adapters.

Time stamp formats in log files

Time stamps are shown in the ISO 8601 format in log files for the ObjectServer, proxy server, **nco_dbinit** utility, probe, ObjectServer Gateway, and other gateways. For compatibility with earlier versions, you can use the **OldTimeStamp** property to switch to the old time stamp format used in V7.2.1, or earlier. You might find this property useful if you already have tools in place for parsing log files. Note that the **nco_dbinit** log file time stamps cannot be switched to the old format because this utility does not have an **OldTimeStamp** property.

A comparison of the formats is as follows:

Old format in V7.2.1, or earlier	ISO 8601 format
dd/MM/YYYY hh:mm:ss AM dd/MM/YYYY hh:mm:ss PM when the locale is set to en_GB on a Solaris 9 computer For example: 01/05/2009 07:15:04 AM	YYYY-MM-DDThh:mm:ss Where T separates the date and time, hh is in 24-hour clock, and the numbers are shown in Western Arabic digits (0-9). For example: 2001-10-21T13:43:11

Formatting and parsing of dates and times

In Tivoli Netcool/OMNIBus V7.2.1, or earlier, the POSIX `strftime()` function is used in date and time conversions. For the ObjectServer SQL functions (`to_char`, `to_date`, and `to_time`), and the probe rules file functions (`datetotime` and `timetodate`), you can define an output format by specifying a format string that consists of zero or more conversion specifiers. For example, the POSIX format for output can be defined in the ObjectServer `to_time` function as follows:

```
to_time('Thu Dec 11 2003', '%a %b %d %Y')
```

In Tivoli Netcool/OMNIBus V7.3.1, the International Components for Unicode (ICU) libraries use the Locale Data Markup Language (LDML) for date and time

patterns. The characters used in these patterns are defined at <http://userguide.icu-project.org/formatparse/datetime>. Use these date and time patterns wherever possible in your ObjectServer SQL functions and probe rules file functions to obtain your required results.

To maintain compatibility with earlier versions, there is continued support for the POSIX format in the date and time functions for the ObjectServer and probe rules files. Note, however, that the POSIX format is not fully compatible with the parsing technology used for LDML date and time patterns. Some POSIX formats are also not supported. When fully compatible with the parsing technology, identical output is obtained for the POSIX format in V7.3.1 as in earlier versions. When partially compatible, variations can occur in the output obtained for the POSIX format across product versions. For example, the following variations can be obtained for the same date and time:

Result for POSIX %c format in V7.3.1: Monday, July 20, 2009 10:18:43 AM United Kingdom Time

Result for POSIX %c format in earlier versions: Mon Jul 20 10:18:43 2009

Result for POSIX %x format in V7.3.1: Monday, July 20, 2009

Result for POSIX %x format in earlier versions: 07/20/09

The following table provides some guidance on the POSIX formats that are fully-supported or partially-supported in V7.3.1. The first column shows the standard POSIX conversion specifiers that can be used in the date and time functions, and the expected result. The second and third columns indicate whether each conversion specifier is fully supported in V7.3.1 and whether the conversion specifier matches the expected result after parsing. Additionally, the second column lists the results for the POSIX format in the C, en_GB, and en_US locales, while the third column lists the results for the POSIX format in all other locales except Hindi and Arabic.

Note: This information is based on checks that were run on a Solaris 9 host. POSIX output varies across operating systems, so you might observe some variations from the results shown in the table.

Table 5. Compatibility for POSIX format in date and time conversions in V7.3.1

Standard POSIX format supported in date and time parsing (and expected result)	V7.3.1 results for C, en_GB, and en_US locales	V7.3.1 results for all other locales except Hindi, Arabic
%a is replaced by the locale's abbreviated weekday name.	Identical result	Not identical
%A is replaced by the locale's full weekday name.	Identical result	Not identical
%b is replaced by the locale's abbreviated month name.	Identical result	Not identical
%B is replaced by the locale's full month name.	Identical result	Not identical
%c is replaced by the locale's appropriate date and time representation.	Not identical	Not identical

Table 5. Compatibility for POSIX format in date and time conversions in V7.3.1 (continued)

Standard POSIX format supported in date and time parsing (and expected result)	V7.3.1 results for C, en_GB, and en_US locales	V7.3.1 results for all other locales except Hindi, Arabic
%C is replaced by the century number (the year divided by 100 and truncated to an integer) as a decimal number [00-99].	Not supported	Not supported
%d is replaced by the day of the month as a decimal number [01,31].	Identical result	Identical result
%D same as %m/%d/%y.	Identical result	Identical result
%e is replaced by the day of the month as a decimal number [1,31]; a single digit is preceded by a space.	Identical result	Identical result
%h same as %b.	Identical result	Identical result
%H is replaced by the hour (24-hour clock) as a decimal number [00,23].	Identical result	Identical result
%I is replaced by the hour (12-hour clock) as a decimal number [01,12].	Identical result	Identical result
%j is replaced by the day of the year as a decimal number [001,366].	Identical result	Identical result
%m is replaced by the month as a decimal number [01,12].	Identical result	Identical result
%M is replaced by the minute as a decimal number [00,59].	Identical result	Identical result
%n is replaced by a newline character.	Identical result	Identical result
%p is replaced by the locale's equivalent of either a.m. or p.m.	Identical result	Identical result
%r is replaced by the time in a.m. and p.m. notation; in the POSIX locale this is equivalent to %I:%M:%S %p.	Identical result	Not identical
%R is replaced by the time in 24 hour notation (%H:%M).	Identical result	Identical result
%S is replaced by the second as a decimal number [00,61].	Identical result	Identical result
%t is replaced by a tab character.	Identical result	Identical result

Table 5. Compatibility for POSIX format in date and time conversions in V7.3.1 (continued)

Standard POSIX format supported in date and time parsing (and expected result)	V7.3.1 results for C, en_GB, and en_US locales	V7.3.1 results for all other locales except Hindi, Arabic
%T is replaced by the time (%H:%M:%S).	Identical result	Identical result
%U is replaced by the week number of the year (Sunday as the first day of the week) as a decimal number [00,53].	Not supported	Not supported
%u is replaced by the weekday as a decimal number [1,7], with 1 representing Monday.	Identical result	Identical result
%V is replaced by the week number of the year (Monday as the first day of the week) as a decimal number [01,53]. If the week containing 1 January has four or more days in the new year, then it is considered week 1. Otherwise, it is the last week of the previous year, and the next week is week 1.	Identical result	Identical result
%W is replaced by the week number of the year (Monday as the first day of the week) as a decimal number [00,53]. All days in a new year proceeding the first Monday are considered to be in week 0.	Not supported	Not supported
%w is replaced by the weekday as a decimal number [0,6], with 0 representing Sunday.	Not supported	Not supported
%x is replaced by the locale's appropriate date representation.	Not identical	Not identical
%X is replaced by the locale's appropriate time representation.	Not identical	Not identical
%y is replaced by the year without century as a decimal number [00,99].	Identical result	Identical result
%Y is replaced by the year with century as a decimal number.	Identical result	Identical result
%Z is replaced by the timezone name or abbreviation, or by no bytes if no timezone information exists.	Identical result	Identical result
%% is replaced by %.	Identical result	Identical result

Additional notes:

- The following POSIX formats are not supported in Tivoli Netcool/OMNIBus V7.3 or V7.3.1: %U, %w, %W, %C

- For Arabic and Hindi locales, the digits in the formatted output are in the Hindi number format instead of the western Arabic number; so the result is different from the POSIX result.
- Modified conversion specifiers of the POSIX format, which start with E or O are not supported.
- The locale-related formats (%c, %r, %x and %X) can be used individually in a format string, or can be used together only in the following combinations:
 - %x %X
 - %x %r
 Other combinations like %x %C or %X %x result in an error "Invalid date/time format".
- If the locale-related formats (%c, %r, %x and %X) are used with any ordinary characters or other non-locale related formats such as %a or %b, the characters and non-locale related formats are silently ignored. For example:
 - %c YEAR is treated the same way as %c
 - %A %b %x is treated the same way as %x
- V7.2.1, or earlier versions, can only parse time strings that contain local timezone information. The following example shows how a sting that includes timezone information can be parsed in V7.3.1:

String	Output
select to_time('2009-03-28:10:00:00 GMT+08:00', 'yyyy-MM-dd:HH:mm:ss vv') from alerts.status	FUNC ----- 1238205600

Multi-byte character string processing

Support is provided to handle invalid characters during multi-byte character string processing. If an invalid character is encountered, the invalid character is substituted with a question mark (?), and processing continues. A warning message is also recorded in the log file about the invalid character.

Tivoli Integrated Portal versions

The Web GUI is based on Tivoli Integrated Portal V2.1. Adhere to a specific setup if you want the Web GUI to coexist with products that on Tivoli Integrated Portal V1.1. The V1.1 and V2.1 components must be each installed into a unique path and run on unique port numbers. V1.1 and V2.1 can be installed on the same server as the Tivoli Netcool/OMNIBus server-side components and integrated with these components. V1.1 and V2.1 are compatible with Tivoli Netcool/OMNIBus V7.2.1 or later. If you install V1.1 and V2.1 with the same user (root or non-root), the Deployment Engine (DE) is shared across the versions. However, if you install one version as root, and the other as non-root, each version has its own DE.

Related concepts

“Integration with other Tivoli products” on page 31

Federal Information Processing Standard 140–2 (FIPS 140–2) support

Federal Information Processing Standards (FIPS) are standards and guidelines that the National Institute of Standards and Technology (NIST) issues for use in United States federal government computer systems.

The Federal Information Processing Standard 140–2 (FIPS 140–2) defines security requirements for cryptographic modules that are used to protect sensitive information in computer and telecommunication systems.

Tivoli Netcool/OMNIBus uses the FIPS 140–2 approved cryptographic providers, IBMJCEFIPS (certificate 376) or IBMJSSEFIPS (certificate 409), and IBM Crypto for C (ICC) (certificate 384), for cryptography. The certificates are listed on the NIST Web site at <http://csrc.nist.gov/cryptval/140-1/1401val2004.htm>.

The FIPS 140–2 approved cryptographic providers provide both cryptographic functions and Secure Sockets Layer (SSL) data protection, on both client and server applications. When Tivoli Netcool/OMNIBus is running in FIPS 140–2 mode, all encryption and key generation functions are provided by the FIPS 140–2 approved cryptographic modules.

FIPS 140–2 configuration checklist

If you intend to run Tivoli Netcool/OMNIBus in FIPS 140–2 mode, the configuration steps that are required are dependent on your installation environment. Use the following checklist as a guide for configuring FIPS 140–2 mode.

Table 6. FIPS 140–2 configuration checklist

Requirement	Action for FIPS 140–2 operation
<input type="checkbox"/> If required, upgrade to FIPS 140–2 mode.	<ul style="list-style-type: none">• If your existing installation uses DES encryption for passwords, change the encryption scheme for passwords to AES. Perform this task either before you upgrade or after you upgrade. The timing depends on the version from which you are upgrading.• If your existing installation uses property value encryption, or uses the nco_g_crypt and nco_pa_crypt utilities to encrypt passwords:<ol style="list-style-type: none">1. Decrypt the encrypted values before you upgrade.2. Upgrade and then configure your server components for FIPS 140–2 mode.3. Encrypt the values again by using the algorithm and mode of operation defined as AES_FIPS.

Table 6. FIPS 140–2 configuration checklist (continued)

Requirement	Action for FIPS 140–2 operation
<input type="checkbox"/> If you want to switch your V7.3 installation to FIPS 140–2 mode, reconfigure your encryption if required. (The remainder of the entries in this checklist also apply.)	<ul style="list-style-type: none"> • If your installation uses DES encryption for passwords, change the encryption scheme for passwords to AES. • If your installation uses AES property value encryption, or uses the nco_g_crypt and nco_pa_crypt utilities to encrypt passwords: <ol style="list-style-type: none"> 1. Decrypt the encrypted values. 2. Configure your server components for FIPS 140–2 mode. 3. Encrypt the values again by using the algorithm and mode of operation defined as AES_FIPS.
<input type="checkbox"/> Configure the Tivoli Netcool/OMNIBus JRE for FIPS 140–2 mode.	<p>After installing or upgrading:</p> <ul style="list-style-type: none"> • Make configuration changes to the security properties file (java.security). • Additionally download and add policy files to use enhanced encryption algorithms.
<input type="checkbox"/> Configure FIPS 140–2 support for your server components.	<p>After configuring your JRE:</p> <ol style="list-style-type: none"> 1. Create a FIPS configuration file (fips.conf) for FIPS 140–2 initialization. 2. Configure ObjectServers, process agents, proxy servers, and ObjectServer gateways with the required FIPS 140–2 settings.
<input type="checkbox"/> If using SSL, enable FIPS 140–2 mode for SSL connections.	<p>Before creating the key database, which is used to store digital certificates and keys, enable the use of FIPS 140–2 certified cryptography by configuring the relevant properties for the IBM Key Management (iKeyman) utility.</p>
<input type="checkbox"/> Configure the Web GUI for FIPS 140–2 mode.	<p>After configuring Tivoli Netcool/OMNIBus:</p> <ol style="list-style-type: none"> 1. Enable FIPS 140–2 mode on the Tivoli Integrated Portal server. 2. Enable FIPS 140–2 mode on the Web GUI clients. 3. Encrypt passwords using FIPS 140–2 mode. 4. Configure a Secure Socket Layer (SSL) connection for the event data feed between the ObjectServer and the Web GUI. 5. Configure an SSL connection for administering the Web GUI using WAAPI from a remote host. For more information, see the <i>IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide</i>.

Use the links that follow to obtain further information on performing these tasks.

Related concepts

“Guidelines for upgrading to FIPS 140–2 mode (UNIX and Linux)” on page 65

“Guidelines for upgrading to FIPS 140–2 mode (Windows)” on page 128

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

“Enabling FIPS 140-2 mode for the Web GUI” on page 496

Related reference

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102

“Configuring the JRE for FIPS 140–2 mode (Windows)” on page 158

“Switching your installation to FIPS 140-2 mode” on page 299

Additional requirements for the Web GUI

Before you install the Web GUI, read this additional information about the deployment requisites and the Web GUI features.

Deployment considerations for the Web GUI

Your Web GUI deployment depends on the specification of the available hardware, the load to be placed on the system, the requirements of any other Tivoli products, and any failover considerations.

The following list contains items to consider for your Web GUI installation:

Coexistence with IBM Tivoli Netcool/Webtop V2.2

The Tivoli Netcool/OMNIBus V7.3.1 Web GUI cannot coexist on the same instance of Tivoli Integrated Portal as IBM Tivoli Netcool/Webtop V2.2. However, you can run the Tivoli Netcool/OMNIBus V7.3.1 Web GUI and Netcool/Webtop V2.2 on separate instances of Tivoli Integrated Portal on the same computer, or in a distributed environment.

Licensing

The Web GUI does not require a license key. The Web GUI is compatible with IBM Tivoli License Compliance Manager. IBM Tivoli License Compliance Manager allows you to monitor and manage your IBM software usage and license compliance. IBM Tivoli License Compliance Manager is available separately to the Web GUI.

ObjectServer user

To avoid using the ObjectServer root login, you can create a new administrative user and password on the ObjectServer or ObjectServers for Web GUI server connections.

Performance

For performance reasons, the Web GUI and the Tivoli Integrated Portal server should be installed on a separate computer to other Tivoli products.

Storage space

The Tivoli Integrated Portal server system can require a large amount of storage space to accommodate the home page requirements of large numbers of Web GUI users. Ensure that the server has adequate disk capacity, and that the data it holds can be backed up regularly.

System communications

Many organizations use proxy servers to administer firewall, filtering,

connection sharing, and caching activities. If the Tivoli Integrated Portal server needs to communicate with other systems using a proxy server, it might be necessary to add the appropriate data transaction privileges to the intermediary system. In addition, Tivoli Integrated Portal server must be able to communicate with one or more ObjectServers.

System load

Consider the scale of the load the target system is likely to encounter when available, and its position within the topology of your network. If the system is behind a firewall, consider how external clients could establish a connection.

Central user registry

As a post-installation task you can configure a central user registry for user management and authentication. You can configure an LDAP server or Tivoli Netcool/OMNIBus ObjectServer registry (or both).

Note: When you add a new user, you should check that the user ID you specify does not already exist in any of the user repositories to avoid difficulties when the new user attempts to log in.

In a network environment that includes a user registry on an LDAP server or Tivoli Netcool/OMNIBus ObjectServer, you can configure Tivoli Netcool/OMNIBus Web GUI to use either or both types.

Before configuring a central user registry, be sure that the user registry or registries that you plan to identify are started and can be accessed from the computer where you have installed the Tivoli Netcool/OMNIBus Web GUI.

Attention: When Tivoli Netcool/OMNIBus Web GUI is configured with multiple central user repositories, you cannot login if one remote user repository becomes inaccessible from Tivoli Netcool/OMNIBus Web GUI, even if your user ID exists in one of the other repositories. If you need access in this situation, you have to run WebSphere Application Server commands to allow access when all repositories are available, or the federated repositories will not function properly. For more information, refer to the following links:

- <http://www-01.ibm.com/support/docview.wss?uid=swg1PK78677>
- http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.web20fep.multiplatform.doc/info/ae/ae/rxml_atidmgrrealmconfig.html

Synchronization of user repositories

Web GUI users and user groups that are stored in external user repositories (such as LDAP or Active Directory), and which are configured to be part of the federated repository are automatically synchronized with the ObjectServer or ObjectServers that are defined for event retrieval.

If you specify an external user repository during installation, user synchronization with the ObjectServer is enabled by default.

Important: If you add an ObjectServer to the federated repository, users from external user repositories cannot be synchronized with that ObjectServer. This is because, across all user repositories in the federated repository configuration, user IDs must be unique. (In the federated repository, all configured user repositories

are active.) Therefore, if you want user synchronization to be enabled, do not add ObjectServers to the federated repository either during installation, or after installation.

For more information about the federated repository, see the WebSphere Application Server information center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/cwim_fedrepos.html

To be synchronized with the ObjectServer, the users and user groups must have the ncw_admin role or the ncw_user role. For more information about Web GUI users, groups, and roles, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

In an ObjectServer, to distinguish between already-existing users and users that are created by the synchronization process, all users that are created by the synchronization process are assigned to a user group called vmmusers. If an ObjectServer does not already contain this user group, it is created automatically. The name vmmusers is a default, and can be changed.

Tip: In this documentation, the terms *user repository* and *user registry* are used interchangeably.

Related tasks

“Changing user registries after installation” on page 477

Related reference

“Data source configuration file overview” on page 512

Single sign-on

The single sign-on (SSO) capability in Tivoli products means that you can log on to one Tivoli application and then launch to other Tivoli Web-based or Web-enabled applications without having to re-enter your user credentials.

The repository for the user IDs can be the Tivoli Netcool/OMNIBus ObjectServer or a Lightweight Directory Access Protocol (LDAP) registry. A user logs on to one of the participating applications, at which time their credentials are authenticated at a central repository. With the credentials authenticated to a central location, the user can then launch from one application to another to view related data or perform actions. Single sign-on can be achieved between applications deployed to Tivoli Integrated Portal servers on multiple machines.

Single sign-on capabilities require that the participating products use Lightweight Third Party Authentication (LTPA) as the authentication mechanism. When SSO is enabled, a cookie is created containing the LTPA token and inserted into the HTTP response. When the user accesses other Web resources (portlets) in any other application server process in the same Domain Name Service (DNS) domain, the cookie is sent with the request. The LTPA token is then extracted from the cookie and validated. If the request is between different cells of application servers, you must share the LTPA keys and the user registry between the cells for SSO to work. The realm names on each system in the SSO domain are case sensitive and must match exactly. See Managing LTPA keys from multiple WebSphere Application Server cells on the WebSphere Application Server Information Center.

Related concepts

“Integration with other Tivoli products”

 Lightweight Third Party Authentication

Related tasks

“Configuring single sign-on” on page 530

Integration with other Tivoli products

Tivoli Netcool/OMNIBus can extend its functionality through integration with other IBM products and components. This integration enhances the event management capability of Tivoli Netcool/OMNIBus because it enables data exchange across products and supports launch-in-context navigation from Tivoli Netcool/OMNIBus to supporting products for further action.

For the lists of products that you can integrate with each component of Tivoli Netcool/OMNIBus, see “Supported operating systems and compatible products” on page 11.

IBM Tivoli Network Manager IP Edition and IBM Tivoli Business Service Manager (TBSM) require integration with Tivoli Netcool/OMNIBus to become fully operational. The following information lists the versions of this product that can be integrated with the server-side components and the Web GUI component of Tivoli Netcool/OMNIBus V7.3.1:

- The Tivoli Netcool/OMNIBus V7.3.1 server-side components are compatible with IBM Tivoli Business Service Manager (TBSM) V4.2.1. However, if you intend to install the server-side components on a computer that already hosts TBSM V4.2.1, you must make sure that, at a minimum, Fix Pack 01 has been applied to the TBSM installation.
- Compatibility between the Tivoli Netcool/OMNIBus V7.3.1 Web GUI and TBSM V4.2.1 is restricted to launch-in-context.
- The Tivoli Netcool/OMNIBus V7.3.1 server-side components are compatible with IBM Tivoli Network Manager IP Edition V3.8 and V3.9.
- The Tivoli Netcool/OMNIBus V7.3.1 Web GUI component is compatible only with IBM Tivoli Network Manager IP Edition V3.9. The Web GUI is not compatible with IBM Tivoli Network Manager IP Edition V3.8.

Compatibility for launch-in-context integrations

You can configure the Web GUI for launch-in-context integrations with several Tivoli products. Launch-in-context is supported for Tivoli products that run within the Tivoli Integrated Portal framework, and with products that run in other GUI frameworks.

Typically, you must configure both the Web GUI and the integrating product after both products are installed. Depending on the type of integration you require, you must configure one or both products.

The following table describes the Tivoli products that support launch-in context integrations with the V7.3.1 Web GUI, and which versions.

Table 7. Supported products and version for launch-in-context integration with the V7.3.1 Web GUI

Product	Version	Product runs within Tivoli Integrated Portal framework
IBM Tivoli Business Service Manager	4.2.1	Yes Note: Launch-in-context from TBSM to the Web GUI is supported for the Active Event List (AEL). However, the AEL is launched in a separate browser window, not in the Tivoli Integrated Portal framework. For more information, see the <i>IBM Tivoli Business Service Manager Information Center</i> at http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=%2Fcom.ibm.tivoli.itbsm.doc%2Fwelcome.htm and the documentation for Interim Fix 3 of TBSM at https://www-304.ibm.com/support/docview.wss?uid=swg24027603 .
Tivoli Composite Application Manager for Web Services	6.2.0.4	No
Tivoli Composite Application Manager for Transactions	7.1	No
Tivoli Composite Application Manager for Applications	7.1	No
IBM Tivoli Monitoring	6.2.2	No
Tivoli Netcool/Impact	5.1.0	Yes
IBM Tivoli Network Manager IP Edition	3.9	Yes
Tivoli IBM Tivoli Service Request Manager®	7.0.1.0	No
TotalStorage Productivity Center	4.1	Yes

Related concepts

“Enabling predictive eventing and predictive analytics” on page 416

“Managing virtualized environments” on page 443

“Deploying probes remotely” on page 451

“Enabling support for TADDM events” on page 435

“Compatibility with previous versions” on page 19

“The Deployment Engine” on page 34

“Directory structure for the Deployment Engine” on page 35

“Single sign-on” on page 30

Related tasks

“Configuring launch-in-context integrations with Tivoli products” on page 557

Related reference

Appendix A, “IBM Support Assistant,” on page 605

“Deployment Engine user modes” on page 36

Installation modes

The installer supports three modes of operation: installation wizard, console mode, and silent mode. The different modes provide different degrees of user interaction.

The installation wizard uses a graphical user interface, and presents installation options within pages to guide you through the installation process. Run the installer as a wizard where possible. When installing with the wizard, you have the option to either run the installation from an installation launchpad, or to directly run the installer executable file.

The console mode of operation presents installation options within a command shell, using a set of text-based menus and prompts. You might find it useful to run the installer in console mode for remote installations.

The silent mode of operation eliminates the need for user interaction during installation and enables you to rapidly deploy a customized Tivoli Netcool/OMNIbus configuration on multiple workstations. The silent installation uses a predefined set of installation options in a response file.

About the launchpad

The launchpad provides a common user interface for launching the installation programs for the non-Web-based components of Tivoli Netcool/OMNIbus and the Tivoli Netcool/OMNIbus Web GUI component. The launchpad also provides links to installation information on the product information center, and to the product support Web site.

The non-Web components and the Web GUI component are distributed as two separate installation packages, so the launchpad provides a convenient method for installing these packages. The launchpad program needs to be able to detect the installer executable files for the non-Web and the Web GUI components. After obtaining the installation packages, you must extract the contents of the base Tivoli Netcool/OMNIbus package (which provides the non-Web components) into a temporary directory. To run the Web GUI installer from the Tivoli Netcool/OMNIbus launchpad, you must create a directory called WebGUI in one of two locations:

- Create the WebGUI directory in the directory that contains the `launchpad.sh` or `launchpad.exe` program.
- Create the WebGUI directory as a parent of the directory that contains the `launchpad.sh` or `launchpad.exe` program.

Then, extract the Web GUI installation package into the WebGUI directory.

The launchpad supports a wizard installation only; you cannot run the installer in console or silent mode from the launchpad.

Launchpad prerequisites

A browser is required to use the launchpad. The launchpad supports the following browsers:

- Mozilla 1.7, or later
- Firefox 2.0, or later
- Internet Explorer 6.0, or later

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

“Obtaining the installation package (Windows)” on page 113

Overview of the Netcool home location

The Netcool home location is the base directory where Tivoli Netcool/OMNIBus is installed.

The Netcool home location is defined by the `NCHOME` environment variable. On UNIX and Linux, this location defaults to `/opt/IBM/tivoli/netcool`. On Windows, the Netcool home location defaults to `C:\IBM\tivoli\netcool`.

Other Network Management products that use the `NCHOME` environment variable (for example, IBM Tivoli Network Manager IP Edition) can also be installed into the Netcool home location. Each product installs its specific components and files into a dedicated *product* subdirectory in the Netcool home location. Files that are common to all products are additionally installed in a number of shared subdirectories within the Netcool home location.

The Deployment Engine

The Deployment Engine (DE) is an IBM service component that is packaged and installed as part of the Tivoli Netcool/OMNIBus installation.

The DE provides a service on a user or computer basis to record files, components and packages that are installed, and to maintain a database of the installation transactions.

UNIX and Linux

You can install the Tivoli Netcool/OMNIBus components as either a root or non-root user.

If you run the installer as the root user, the DE database is created by default in the `/var/ibm/common/acs` directory, and is globally accessible to subsequent installers for related Tivoli products. Only a single instance of the DE is required and it can then be a shared resource for other related Tivoli products that are

installed on the same computer. Updated versions of the DE are installed automatically with new product versions and patches, but these updates do not affect the operation of any existing installations. The DE can provide a system-wide audit and management of the installed products.

If you run the installer as a non-root user, the DE database is created by default in a `.acsi_hostname` subdirectory in your home directory. If a root instance of the DE already exists, that root instance is be used instead, not only for the installation of Tivoli Netcool/OMNIBus, but also for all DE-based Tivoli products. If Tivoli Netcool/OMNIBus is installed on the same computer by multiple local users, or if several DE-based Tivoli products are installed by multiple local users, multiple instances of DE databases and files can exist on that computer.

Important: When you decide whether to install as root or non-root, you must consider your policy for installations, backups, uninstallations, and so on. If you run the installer as the root user, the root instance of the DE is shared by other users who subsequently install DE based products. These users require write access to the DE installation, and concurrent DE-based operations, for example, product installation or uninstallation, are not possible. A root installation also means that, if the root instance of the DE is uninstalled, for example to troubleshoot a DE-based product, the DE is removed for all other DE-based products, regardless of which user installed them. The preferred option is for the installer to be run by an identified non-root user, and for future installations and uninstallations to be performed by the same user who ran the original installation. If you are installing Tivoli Netcool/OMNIBus as part of a larger solution, take note also that some product installations can be performed only by a non-root user.

Windows

The DE database is created by default in the `C:\Program Files\IBM\Common\acsi` directory.

Run the installer as the same user each time. Whichever user installs the first DE-based Tivoli product must also install, uninstall, or modify every subsequent, related DE-based Tivoli product on that computer.

Related concepts

“Integration with other Tivoli products” on page 31

“Installation or upgrade prerequisites (UNIX and Linux)” on page 47

“Installation or upgrade prerequisites (Windows)” on page 113

Related reference

Appendix C, “Deployment Engine command reference,” on page 627

Directory structure for the Deployment Engine

When the Deployment Engine (DE) is installed it automatically creates an installation and common directory structure. The resulting directory structures depend on the type of operating system and whether DE was installed by an admin or non-admin user.

Restriction: The DE might not function correctly when installed on a Windows operating system that is running in Chinese mode. This is because Windows is unable to interpret the Chinese GB18030 character set correctly.

The following table shows the DE installation directory structure created for a particular operating system and the type of user:

Table 8. DE default installation directory structure

Operating system	Root or admin user	Non-root or non-admin user (DE is installed in the home location for the user)
Windows	%ProgramFiles%\IBM\common\acsi	This option is not supported.
UNIX	/usr/ibm/common/acsi	/home/user_name/.acsi_hostname

UNIX If you are performing the installation as the root user, you can change the installation directory specified in Table 8. However, change installation directory only if you are unable to write to the default location.

The following table shows the DE common directory structure created for a particular operating system and the type of user:

Table 9. DE common directory structure

Operating system	Root or admin user	Non-root or non-admin user (this is identical to the default installation location)
Windows	%ProgramFiles%\IBM\common\acsi	This option is not supported.
UNIX	/var/ibm/common/acsi	/home/user_name/.acsi_hostname

The common directories specified in Table 9 cannot be changed.

Related concepts

“Integration with other Tivoli products” on page 31

Deployment Engine user modes

A single or multi-user mode is set during the initial installation of the Deployment Engine. The type of user mode depends on whether the installation is performed as a non-root or root user.

When the Deployment Engine is installed as part of an application deployment, the authorities of the person performing the installation determines the user mode that is permanently established for the Deployment Engine runtime environment. The selected user-mode is transparent to the user or administrator that is installing the Deployment Engine as part of the new application.

Single-user mode (local Deployment Engine)

A single-user mode of the Deployment Engine exists where the Deployment Engine has been installed on a computer for the purpose of deploying other software.

The single-user mode is established when the user performing the Deployment Engine installation does not have the required authorities for their operating system and there is no existing multi-user deployment of the DE. This type of user is classed as a non-root user and is typically a general user or product administrator.

The following table provides some examples of non-root users:

Table 10. Examples of non-root users

Operating system	Non-root user
Windows	The user does not have all the required system authorities for their operating system.
Unix	The user does not have a user ID of zero (UID=0).

One or more single-user deployments of the Deployment Engine runtime environment can reside on the same computer. However, each user of that computer can deploy only one.

Note: You can have one multi-user and one or more single-user deployments of Deployment Engine running on the same computer. However, the software deployment program does not install a Deployment Engine runtime environment in single-user mode if a compatible multi-user deployment already exists.

Multi-user mode (global Deployment Engine)

A multi-user mode of the Deployment Engine exists where the Deployment Engine has been installed on a computer and is available to all users of that computer for the purpose of deploying other software.

When a multi-user mode of the Deployment Engine has been installed, all users on the computer that currently do not have a single-user mode DE installed, uses the multi-user mode DE when they attempt to install a DE based product such as Tivoli Netcool/OMNIBus. Users require write access to the multi-user mode DE. Therefore, a multi-user mode DE is installed with world write permissions enabled. The `de_security` command can be used to change the system protection settings of the multi-user mode DE to protect it from unauthorized modifications. For more information, see Appendix C, “Deployment Engine command reference,” on page 627.

The multi-user mode is established when the user performing the Deployment Engine installation has the required user authorities for their operating system and is referred to as the root user. A system administrator is typically a root user.

The following table provides some example of root users:

Table 11. Examples of root users

Operating system	Root user
Windows	The user is a member of the Administrator's group (provided the group's default permissions were not changed).
Unix	The user has a user ID of zero (UID=0).

Note: Only one multi-user deployment of the Deployment Engine runtime environment can reside on the same computer.

Related concepts

“Integration with other Tivoli products” on page 31

Tivoli Integrated Portal components

The Tivoli Integrated Portal installation has a core set of components that provide such administrative essentials as network security and database management. Tivoli Integrated Portal is installed during installation of the Web GUI component.

Core components

IBM Deployment Engine

The first core component installed is the deployment engine because it determines what needs to be installed.

Tivoli Integrated Portal Server

The application server is a J2EE lightweight implementation of the WebSphere Application Server. It provides a single sign-on service based on the WebSphere security module and Lightweight Third Party Authentication (LTPA).

Integrated Solutions Console

The Integrated Solutions Console is the administrative console for your applications. It is a Web-based portal component that provides common task navigation for products, aggregation of data from multiple products into a single view, and message passing between views from different products.

IBM HTTP Server

The Web server is installed with the Tivoli Integrated Portal Server.

Common Gateway Interface Server

The CGI server enables external programs to interact with information servers such as HTTP servers. You can write scripts for the CGI.

Optional components

These are the components that you can choose whether to install. It is possible that not every optional component listed here is offered for your product. See your product documentation for more information.

WebSphere federated repository functionality

Environments that have external user registries can participate in a federated repository. You can configure a Lightweight Directory Access Protocol server or Tivoli Netcool/OMNIBus ObjectServer or both as a central user registry. For single sign-on capability, an external authentication source is required.

Related concepts

“Tivoli Integrated Portal overview” on page 6

Quick reference to getting started

Use this information as a quick reference if you are new to Tivoli Netcool/OMNIBus and want to perform a quick installation and configuration to obtain a running ObjectServer.

The steps are as follows:

Table 12. Quick start instructions

Action	More information
<p>1. Prepare for installation by checking the prerequisites and obtaining the installation package.</p> <p>Note: You can obtain the Web GUI installation package at this stage, but do not install the Web GUI.</p>	<p>“Preparing to install or upgrade (UNIX and Linux)” on page 46</p> <p>“Preparing to install or upgrade (Windows)” on page 112</p>
<p>2. Install Tivoli Netcool/OMNIBus by using the wizard, console mode, or silent mode. Accept all the default installable features.</p>	<p>“Installing on UNIX and Linux” on page 48</p> <p>“Installing on Windows” on page 114</p>
<div>UNIX</div> <div>Linux</div> <p>3. If necessary, set the following environment variables:</p> <ul style="list-style-type: none"> • \$NCHOME • \$OMNIHOME • \$PATH • \$LD_LIBRARY_PATH (Solaris or Linux only) • \$LIBPATH (AIX only) • \$SHLIB_PATH (HP-UX only) <p>See the guidance for setting these environment variables.</p>	<p>“Setting Tivoli Netcool/OMNIBus environment variables (UNIX and Linux)” on page 95</p> <p>“Checking the shared library paths” on page 99</p>
<p>4. Create an ObjectServer by running the database initialization utility as follows:</p> <div>UNIX</div> <div>Linux</div> <pre>\$NCHOME/omnibus/bin/nco_dbinit -server <i>servername</i></pre> <div>Windows</div> <pre>%NCHOME%\omnibus\bin\nco_dbinit -server <i>servername</i></pre> <p>Where <i>servername</i> is the ObjectServer name, which must consist of 29 or fewer uppercase letters and cannot begin with an integer.</p> <p>The default database tables and data, users, groups, roles, and properties file are created. (You can use the default user named root, which is created with a blank password, to log in to the ObjectServer.)</p>	<p>“Creating an ObjectServer” on page 231</p>

Table 12. Quick start instructions (continued)

Action	More information
<p>5. Configure server communication information for the ObjectServer on the host computer.</p> <p>UNIX</p> <p>Linux</p> <ol style="list-style-type: none"> 1. Use the Server Editor to add the communication details by running the following command: <code>\$NCHOME/omnibus/bin/nco_xigen</code> Or: 2. Update the ObjectServer communication information by editing the connections data file (<code>\$NCHOME/etc/omni.dat</code>), and generate the interfaces file for Tivoli Netcool/OMNIBus communications by running the following command: <code>\$NCHOME/bin/nco_igen</code> The interfaces file <code>\$NCHOME/etc/interfaces.arch</code> is created, where <i>arch</i> represents the operating system name. <p>Windows</p> <p>Use the Server Editor to add the communication details:</p> <ol style="list-style-type: none"> 1. Click Start > Programs > NETCOOL Suite > System Utilities > Servers Editor. 2. Enter and save communication information for the ObjectServer. The connections data file (<code>%NCHOME%\ini\sql.ini</code>) is updated with these details. 	<p>“Configuring server communication information” on page 242</p>
<p>6. Start the ObjectServer by running the following command:</p> <p>UNIX</p> <p>Linux</p> <pre>\$NCHOME/omnibus/bin/nco_objserv -name servername</pre> <p>Windows</p> <pre>%NCHOME%\omnibus\bin\nco_objserv -name servername</pre> <p>Where <i>servername</i> is the ObjectServer name.</p>	<p>“Starting an ObjectServer manually” on page 237</p>
<p>7. Prepare to install the Web GUI by checking the prerequisites, deciding on the type of installation required, and gathering the required information.</p>	<p>“Web GUI installation or upgrade prerequisites” on page 174</p> <p>“Types of Installation” on page 173</p> <p>“Gathering installation information” on page 175</p>
<p>8. Install the Web GUI by using the wizard, console mode, or silent mode. Use the information gathered in step 7 to specify the parameters of the installation.</p>	<p>“Installing the Web GUI” on page 178</p>

Table 12. Quick start instructions (continued)

Action	More information
9. Log in to the Web GUI, assign Web GUI roles to the administrative user change the passwords for the supplied users.	<p>“Logging in” on page 208</p> <p>“Assigning Web GUI roles to the administrative user” on page 210</p> <p>“Changing the passwords of the supplied users” on page 212</p>
10. Optional: Perform the configuration for the required user registry, for example LDAP, against the Tivoli Integrated Portal installation.	“Changing user registries after installation” on page 477

More information

Read through the following installation scenarios for more information:

- Chapter 27, “Example Tivoli Netcool/OMNIBus installation scenarios (basic, failover, and desktop architectures),” on page 571
- Chapter 28, “Example installation scenario for the non-Web and Web GUI components of Tivoli Netcool/OMNIBus (Windows),” on page 593

Quick reference to upgrading

Use this information as a quick reference to upgrading the non-Web components of Tivoli Netcool/OMNIBus.

The steps are as follows:

Table 13. Quick reference for upgrading the non-Web components

Action	More information
1. Review compatibility issues with earlier versions of the product.	“Compatibility with previous versions” on page 19
2. Prepare for the upgrade by checking the prerequisites and obtaining the installation package.	<p>“Preparing to install or upgrade (UNIX and Linux)” on page 46</p> <p>“Preparing to install or upgrade (Windows)” on page 112</p>
<p>3. If your existing installation is running in non-FIPS 140–2 mode and you intend to upgrade the product to operate in FIPS 140–2 mode, use the FIPS 140–2 configuration checklist as a guide to upgrading to this mode. Based on your existing setup, some configuration steps might be required before you upgrade. Configuration steps will also be required after you upgrade.</p> <p>Pre-upgrade steps are generally required if the user passwords in your system are currently encrypted by using the DES algorithm, or if you are using property value encryption to encrypt string values in properties files.</p>	<p>“FIPS 140–2 configuration checklist” on page 26</p> <p>“Guidelines for upgrading to FIPS 140–2 mode (UNIX and Linux)” on page 65</p> <p>“Guidelines for upgrading to FIPS 140–2 mode (Windows)” on page 128</p>
4. Back up your existing system and then upgrade Tivoli Netcool/OMNIBus by using the wizard, console mode, or silent mode.	<p>“Upgrading on UNIX and Linux” on page 64</p> <p>“Upgrading on Windows” on page 127</p>

Table 13. Quick reference for upgrading the non-Web components (continued)

Action	More information
5. Review the list of files migrated and perform any manual configuration required.	<p>“Files migrated for an upgrade (UNIX and Linux)” on page 85</p> <p>“Files migrated for an upgrade (Windows)” on page 144</p>
<div>UNIX</div> <div>Linux</div> <p>6. Set the following environment variables if necessary:</p> <ul style="list-style-type: none"> • \$NCHOME • \$OMNIHOME • \$PATH • \$LD_LIBRARY_PATH (Solaris or Linux only) • \$LIBPATH (AIX only) • \$SHLIB_PATH (HP-UX only) <p>See the guidance for setting these environment variables.</p>	<p>“Setting Tivoli Netcool/OMNIBus environment variables (UNIX and Linux)” on page 95</p> <p>“Checking the shared library paths” on page 99</p>
<div>UNIX</div> <div>Linux</div> <p>7. Verify your server communication information.</p> <ol style="list-style-type: none"> 1. Either use the Server Editor to review the communication details by running the following command: \$NCHOME/omnibus/bin/nco_xigen Or: 2. Update the ObjectServer communication information by editing the connections data file (\$NCHOME/etc/omni.dat), and generate the interfaces file for Tivoli Netcool/OMNIBus communications by running the following command: \$NCHOME/bin/nco_igen The interfaces file \$NCHOME/etc/interfaces.arch is created, where <i>arch</i> represents the operating system name. <div>Windows</div> <p>Use the Server Editor to review the communication details:</p> <ol style="list-style-type: none"> 1. Click Start > Programs > NETCOOL Suite > System Utilities > Servers Editor. 2. If necessary, update communication information for the ObjectServer. The connections data file (%NCHOME%\ini\sql.ini) is updated with these details. 	<p>“Configuring server communication information” on page 242</p>

Table 13. Quick reference for upgrading the non-Web components (continued)

Action	More information
<p>8. Start the ObjectServer by running the following command:</p> <p>UNIX</p> <p>Linux</p> <pre>\$NCHOME/omnibus/bin/nco_objserv -name servername</pre> <p>Windows</p> <pre>%NCHOME%\omnibus\bin\nco_objserv -name servername</pre> <p>Where <i>servername</i> is the ObjectServer name.</p>	<p>“Starting an ObjectServer manually” on page 237</p>
<p>9. Upgrade your ObjectServer schema to the V7.3.1 schema.</p>	<p>“Notes on upgrading ObjectServer schemas to V7.3.1 schemas (UNIX and Linux)” on page 81</p> <p>“Notes on upgrading ObjectServer schemas to V7.3.1 schemas (Windows)” on page 141</p>
<p>10. If you upgraded from an earlier version that used SSL for client and server communications, and want to continue to use your old certificates, migrate your certificate files and private keys into the key database that is used for certificate management.</p> <p>Certificate migration is supported only in non-FIPS 140–2 mode. If you intend to operate in FIPS 140–2 mode, you must use iKeyman to re-create all the old certificates that you want to reuse.</p> <p>Note: If you upgraded from V7.2.1, your certificates are automatically migrated to V7.3.1, so no further action is required.</p>	<p>“Migrating your digital certificates and keys (UNIX and Linux)” on page 86</p> <p>“Migrating your digital certificates and keys (Windows)” on page 148</p> <p>“Managing digital certificates for SSL communication” on page 380</p>

Chapter 4. Installing, upgrading, and uninstalling (UNIX and Linux)

Use this information to install, upgrade, and uninstall Tivoli Netcool/OMNIBus on UNIX and Linux operating systems.

Note about the Web GUI installation: The Web GUI component is distributed as a separate installation package from the non-Web-based Tivoli Netcool/OMNIBus components. The set of instructions provided here relate only to the non-Web components. For installation, upgrade, and uninstallation instructions about the Web GUI component, see Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173.

Tip: If you intend to install both the Web GUI and non-Web components by using the installation wizard, you can use the launchpad as a starting point for installing the components. If installing in console or silent mode, you must run the installation programs separately. The Web GUI installation requires a running ObjectServer, so ensure that you have installed and started your ObjectServer before installing the Web GUI.

Related concepts

“About the launchpad” on page 33

Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)

You can choose which Tivoli Netcool/OMNIBus features to install on a UNIX or Linux host.

The following table describes the list of Tivoli Netcool/OMNIBus features that you can install. In the Feature column, the first term (for example, Admin) shows the feature name when installing using the wizard or console mode, and the second term (for example, nco_admin_feature) shows the feature name when installing in silent mode.

Table 14. Feature selection

Feature	Description
Desktop or nco_desktop_feature	Desktop GUI Applications Use the Event List to view and manager alerts in your system. This feature includes On-line Help and the Accelerated Event Notification (AEN) client. Use the Netcool/OMNIBus Administrator to configure ObjectServers and manage services and processes under process control.

Table 14. Feature selection (continued)

Feature	Description
Servers or nco_server_feature	<p>Server Applications</p> <p>The ObjectServer is the in-memory database server at the core of Tivoli Netcool/OMNIBus. Use the ObjectServer to store and process alert information. If you do not install the ObjectServer component, you must have an ObjectServer running elsewhere on your network.</p> <p>A proxy server reduces the number of direct connections to the primary ObjectServer. A proxy server can enhance performance when a large number of probes are forwarding alert information directly to the ObjectServer, and a large number of desktop connections are also made to the same ObjectServer.</p> <p>Use the ObjectServer gateways to connect ObjectServers.</p> <p>Use the process control system to configure and manage processes remotely. Process control simplifies the management of Tivoli Netcool/OMNIBus components such as ObjectServers, probes, and gateways. Additionally use the process agent to start processes that are used by external automations from the ObjectServer.</p> <p>Additional tools are also installed. Use the BAROC tool (nco_baroc2sql) to migrate IBM Tivoli Enterprise Console BAROC data into the ObjectServer. Use the Confpack utility (nco_confpack) to import and export parts of ObjectServer configurations. Use the ObjectServer report tool (nco_osreport) to extract entire ObjectServer configuration into SQL files for use in creating new ObjectServers with nco_dbinit.</p>
Probe Support or nco_probe_support_feature	<p>This feature is required for probe installation, and adds the underlying infrastructure for probes.</p> <p>The Probe Rules Syntax Checker (nco_p_syntax) and the nco_postmsg utility are also installed. Use the Probe Rules Syntax Checker to test the syntax of a rules file. Use the nco_postmsg utility to specify name-value pairs for alert data that can be directly sent as a single event to a specified ObjectServer.</p>

Related concepts

“Tivoli Netcool/OMNIBus components” on page 1

“Online help requirements” on page 16

“Installation modes” on page 33

Preparing to install or upgrade (UNIX and Linux)

Before you install or upgrade Tivoli Netcool/OMNIBus, take note of the user and system permissions that are required for the user performing the installation or upgrade, and ensure that these permissions are granted. You also need to obtain the installation package for your operating system.

Installation or upgrade prerequisites (UNIX and Linux)

If you are installing or upgrading Tivoli Netcool/OMNIBus, you must take note of a number of prerequisites.

These prerequisites are as follows:

- Root access to the system, or root privileges, are not required for the installation or upgrade process, and you can install Tivoli Netcool/OMNIBus either as a root user or a non-root user. The default locations to which the Deployment Engine and Tivoli Netcool/OMNIBus are installed are dependent on this user type. If you are installing Tivoli Netcool/OMNIBus as part of a larger solution, see the documentation for the related products to ensure that the correct user is used when installing Tivoli Netcool/OMNIBus. This ensures that permissions of the Deployment Engine database can be maintained for all the related IBM product installations required.
- Sufficient disk space must be available on the volume where you want to install Tivoli Netcool/OMNIBus. If you intend to install other Network Management products, the installation location must also have sufficient space to accommodate these installations.
- You must have write access permissions to the Netcool home directory (NCHOME) where Tivoli Netcool/OMNIBus is installed.
- You must log in as the preferred user as whom you want to perform the installation. Do not log in as root and then switch to the preferred user. For example, logging in as root and then issuing the **su username** command is not supported.

Note: The installation directory path must not include any special or multibyte characters.

You must perform an upgrade from Tivoli Netcool/OMNIBus V7.1, V7.2, or V7.2.1 as the same user who performed the original installation. If this is not possible, ask your system administrator to make you the owner of all files in the installation. If any other user has created any files in the installation, you must have write permission to them.

Related concepts

"Disk space requirements" on page 17

"JRE requirements" on page 14

"The Deployment Engine" on page 34

Obtaining the installation package (UNIX and Linux)

Tivoli Netcool/OMNIBus is distributed as a compressed file that is available on CD, or that you can download from the IBM Passport Advantage® Online Web site.

The boxed product package contains the CD that you can use to install Tivoli Netcool/OMNIBus on your operating system.

If you are downloading the product, proceed as follows:

1. Follow the instructions in the download document for your operating system:

Operating system	Download document location
AIX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026995

Operating system	Download document location
HP-UX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026996
HP-UX Integrity	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026999
Linux	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026997
Linux for System z	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026998
Solaris	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027000

2. Extract the contents of the installation package into a temporary location.

What to do next

After you complete these tasks, you can run the installation program to perform a new installation of Tivoli Netcool/OMNIbus or to upgrade your existing version.

Related concepts

“Installing on UNIX and Linux”

“Upgrading on UNIX and Linux” on page 64

Installing on UNIX and Linux

On UNIX and Linux systems, you can install Tivoli Netcool/OMNIbus by using the installation wizard, or the console or silent installation mode.

The documented instructions apply for a new installation of Tivoli Netcool/OMNIbus as the first product, or as a subsequent, related Tivoli product that is installed in the Netcool home location.

The installation process results in a package installation of the Tivoli Netcool/OMNIbus components. A package is a collection of related configuration files.

After you complete the installation process, you must configure Tivoli Netcool/OMNIbus before attempting to use the system.

You can have multiple Tivoli Netcool/OMNIbus installations (and versions) on the same computer, providing the installations are in different directories. Tivoli Netcool/OMNIbus V7.3.1 will work with earlier versions, but they will not share common components.

Related concepts

“Overview of the Netcool home location” on page 34

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

“Installing using the installation wizard (UNIX and Linux)” on page 49

“Installing in console mode (UNIX and Linux)” on page 52

“Installing in silent mode (UNIX and Linux)” on page 54

Installing using the installation wizard (UNIX and Linux)

Run the wizard to present the installation options in a graphical user interface.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIbus or the Web GUI on a new machine with a version of the DE currently installed.

To install Tivoli Netcool/OMNIbus:

1. From a command prompt, change to the directory where you extracted the contents of the installation package.
2. Either start the installation by using the launchpad, or directly run the command to start the installation program:

- Enter the following command to start the launchpad:

```
./launchpad.sh
```

When the launchpad window opens, select a language, and then click through each of the following options in the left navigation pane in order to review the welcome information, prerequisite information, and installation scenarios: Welcome, Prerequisite Information, and Installation Scenarios.

When you are ready to install Tivoli Netcool/OMNIbus, click the Install Product option in the left navigation pane. Then click the **Start Tivoli Netcool/OMNIbus Installation** button that is associated with the Tivoli Netcool/OMNIbus installation. When the IBM Tivoli Netcool/OMNIbus splash screen is displayed, proceed to the next step.

- Enter the following command to run the installation program:

```
./install.bin
```

The JRE and installation resources are extracted from the installer archive, and the IBM Tivoli Netcool/OMNIbus splash screen is displayed.

Tip: When running the **install.bin** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.bin** command. The **-r** command-line option must be the last option specified.

3. From the drop-down list at the bottom of the splash screen, select a language and click **OK**.
4. If you are installing Tivoli Netcool/OMNIbus as a root user and there is no global Deployment Engine (DE) installed: a message informs you that if you install a global DE, it will be used by any user that attempts to install a DE based product (unless they currently have a single user DE). If you are installing Tivoli Netcool/OMNIbus as a non-root user and you do not have a single user Deployment Engine installed but there is a global Deployment Engine installed: a message informs you that the global DE will be used and that you must have write permissions to the database. You have the option to either terminate or proceed with the installation.
5. From the Introduction page of the installation wizard, click **Next** to proceed to the Software Licence Agreement page.

Tip: While configuring your installation options, you can click **Previous** to revisit previous pages of the wizard and make changes if required.

6. Read the license agreement and the non-IBM terms, and then accept both the IBM and non-IBM terms. You can also click **Print** to obtain a hardcopy of the license agreement.
7. Click **Next**. If you are installing as a root user, an information message is displayed at this stage. To confirm that you want to install as root, click **Yes**. Otherwise, click **No**, in which case the installation is terminated.
8. If you are installing as a root user, the Autonomic Deployment Engine page is displayed. From this page, choose the installation location for the Deployment Engine as follows:
 - Click **Install in recommended location** to accept the default installation location for the Deployment Engine. This is the preferred option. For a root user, the installation location for the Deployment Engine defaults to `/usr/ibm/common/acsi`.

Note: If you are installing as a non-root user, you do not see the Autonomic Deployment Engine page. The Deployment Engine is installed in `/home/username/.acsi_hostname`, where *hostname* is the name of the computer.

- Click **Install into other location** to choose a different location. Only use this option for root installations on Solaris Sparse zones or other restricted systems where root cannot write to the `/usr` directory. To specify a location:
 - a. Click **Next**.
 - b. From the "Autonomic Deployment Engine - Choose a folder" page, specify an alternative installation location. You can click **Restore Default Folder** to revert to the default installation location.

Note: If the Deployment Engine is already installed on your computer, the existing installation location takes precedence over any alternative location that you specify. If you choose an alternative location, the Deployment Engine files are split between two locations: the database files are stored in the `/var/ibm/commom/acsi` directory, and the remaining Deployment Engine resources are stored in your chosen location.

9. Click **Next** and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.
10. If you are installing a global instance of the Deployment Engine as a root user, the Deployment Engine Access Permission page is displayed. Choose the user access security policy you want to apply to the Deployment Engine:
 - **Do not change:** Choose this option to leave the security policy unchanged. Then click **Next**.
 - **Single User (current user only):** Choose this option to restrict the use of this Deployment Engine to the root user only. Then click **Next**.
 - **Group (current user and members of an existing group):** Choose this option to restrict the use of this Deployment Engine to the root user, and members of an existing user group. Then:
 - a. Click **Next**.
 - b. In the Group access entry page, enter the name of an existing user group.

Note: The user group must currently exist. No new user groups can be created.

- c. Click **Next**.
 - **Global (all users)**: Choose this option to allow all users to access the Deployment Engine.
 11. From the Select Destination Folder page, specify an installation location for Tivoli Netcool/OMNIBus. This location becomes your NCHOME location. The installation location defaults to `/opt/IBM/tivoli/netcool`.
 12. Click **Next** to proceed to the Choose Install Set page. Choose either of the installation options by clicking the associated icon:
 - **Typical**: Choose this option to install all the Tivoli Netcool/OMNIBus features. Then click **Next**.
 - **Custom**: Choose this option if you want to select the features that you want to install. Then:
 - a. Click **Next**.
 - b. From the list box, select each feature that you want to install by selecting its associated check box, or deselect a feature by clearing the associated check box.
 - c. Click **Next** after selecting all the features that you want to install.
- Tip:** You can switch from a custom to a typical installation by selecting Typical from the **Install Set** drop-down list. Typical selects all features, whereas Custom reverts to your subset of selected features.
- The Data Migration page is displayed, asking whether you want to migrate data from an existing Tivoli Netcool/OMNIBus installation.
13. Select the **No** option and click **Next**.

After a short interval during which the system is configured, the Pre-Installation Summary page is displayed.
14. Review the installation settings and then click **Install** to start the installation. The Installing Netcool/OMNIBus page shows the progress of the installation. On completion, the Installation Complete page is displayed, confirming that Tivoli Netcool/OMNIBus has been successfully installed.
15. Click **Done** to close the wizard.

If you started the installation program from the launchpad, you can return to the launchpad window, and click **Post-Installation** in the navigation pane to review postinstallation information. Then click **Exit** and confirm that you want to exit the launchpad.

Results

The installation adds a number of files to your system.

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)” on page 45

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173

“Viewing the installation log files (UNIX and Linux)” on page 60

“Performing postinstallation tasks (UNIX and Linux)” on page 94

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

“Installation error messages” on page 613

Installing in console mode (UNIX and Linux)

Run the installation in console mode if you want to complete the installation options by using a series of menus and prompts within a text-based user interface.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIBus or the Web GUI on a new machine with a version of the DE currently installed.

Tip: During the installation, you can enter quit from most of the menu screens to exit the installer. You can also enter back from some of the menu screens to return to the previous screen.

To install Tivoli Netcool/OMNIBus in console mode:

1. From a command prompt, change to the directory where you extracted the contents of the installation package.
2. Enter the following command to run the installation program:

```
./install.bin -i console
```

Wait while the JRE and installation resources are extracted from the installer archive.

Tip: When running the **install.bin** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.bin** command. The **-r** command-line option must be the last option specified.

3. Enter a number that corresponds to the language you want to use for the installation procedure.
4. If you are installing Tivoli Netcool/OMNIBus as a root user and there is no global Deployment Engine (DE) installed: a message informs you that if you install a global DE, it will be used by any user that attempts to install a DE based product (unless they currently have a single user DE). If you are installing Tivoli Netcool/OMNIBus as a non-root user and you do not have a single user Deployment Engine installed but there is a global Deployment

Engine installed: a message informs you that the global DE will be used and that you must have write permissions to the database. You have the option to either terminate or proceed with the installation.

5. Read the Introduction information and press Enter, as prompted.
6. To read the non-IBM terms, press 4, and then press Enter to scroll through the text. After reading the non-IBM terms, press q to return to the license agreement. Press Enter to scroll through the license agreement, and then enter 1 to accept the agreement.
7. If you are installing as a root user, you are asked to confirm whether you want to install as root. Enter 1 as confirmation, and go to step 8. Otherwise, enter 2 to stop the installation so that you can install as a non-root user.
8. If you are installing as a root user, choose the installation location of the Deployment Engine as follows:
 - Enter 1 to accept the default installation location for the Deployment Engine. This is the preferred option. For a root user, the installation location for the Deployment Engine defaults to `/usr/ibm/common/acsi`.

Note: If you are installing as a non-root user, you are not prompted for a location. The Deployment Engine is installed in `/home/username/.acsi_hostname`, where *hostname* is the name of the logged-in user.

- Enter 2 to choose a different location. Then specify an alternative installation location. Only use this option for root installations on Solaris Sparse zones or other restricted systems where root cannot write to the `/usr` directory.

Note: If the Deployment Engine is already installed on your computer, the existing installation location takes precedence over any alternative location that you specify. If you choose an alternative location, the Deployment Engine files are split between two locations: the database files are stored in the `/var/ibm/commom/acsi` directory, and the remaining Deployment Engine resources are stored in your chosen location.

9. Press Enter and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.
10. If you are installing a global instance of the Deployment Engine as a root user, the Deployment Engine Access Permission page is displayed. Choose the user access security policy you want to apply to the Deployment Engine:
 - Enter 1 to leave the security policy unchanged.
 - Enter 2 to restrict the use of this Deployment Engine to the root user only.
 - Enter 3 to restrict the use of this Deployment Engine to the root user, and members of an existing user group. Then:
 - a. In the Group access entry page, enter the name of an existing user group.

Note: The user group must currently exist. No new user groups can be created.

- Enter 4 to allow all users to access the Deployment Engine.
11. Specify an installation location for Tivoli Netcool/OMNIBus. This location becomes your NCHOME location.

The installation location defaults to `/opt/IBM/tivoli/netcool`.
 12. Choose the type of installation to perform:
 - Enter 1 to install all the Tivoli Netcool/OMNIBus features.

- Enter 2 to select the features that you want to install.

In the resulting menu, all features are initially selected, as denoted by [X], so enter a comma-separated list of numbers that correspond to the features you do *not* want to install. (Note that this screen has a toggle function that enables you to type a number and press Enter to deselect a selected feature denoted by [X], or to type a number and press Enter to select a deselected feature denoted by [].)

Your list of selected features is then shown. You can either enter back to return to the previous screen and revise your selection, or press Enter to confirm your selection and continue.

13. Enter 2 to indicate that you do not want to migrate data from an existing Tivoli Netcool/OMNIBus installation.
14. Review the pre-installation summary, and verify that all your required features are selected. Then press Enter to start the installation. On completion, a confirmation message is displayed.
15. Press Enter to exit the installer.

Results

The installation adds a number of files to your system.

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)” on page 45

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

“Viewing the installation log files (UNIX and Linux)” on page 60

“Performing postinstallation tasks (UNIX and Linux)” on page 94

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

“Installation error messages” on page 613

Installing in silent mode (UNIX and Linux)

Run the installation in silent mode if you want to deploy Tivoli Netcool/OMNIBus with identical installation configurations on multiple workstations. In silent mode, the installer suppresses the graphical or text interface and obtains the installation settings from a predefined response file.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIBus or the Web GUI on a new machine with a version of the DE currently installed.

Important: On Solaris Sparse zones, installation as a root user in silent mode will only work on Solaris whole root zones. For root installations on Solaris Sparse zones or other restricted systems where root cannot write to the /usr directory, you must install using the wizard or console mode. Using these modes, you can specify an alternative location for the Deployment Engine, which installs to the /usr directory by default.

The silent mode of installation has two parts:

1. Define your installation settings in a response file.
2. Run the installation program with the settings in this file.

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

Defining your installation settings in a response file (UNIX and Linux)

Before you can run the installation program in silent mode, you must create a response file that defines the features you want to install.

The installation package includes a sample response file that is located in the directory where you extracted the package. The file is called OMNIBus-response.txt. Make a copy of the sample file and use the copy to specify your installation options.

Note: If you previously ran the wizard or console installer with the -r command-line option in order to save your installation settings to an auto-generated installer.properties file, you can use this file as your response file.

To create a response file with your preferred installation options:

1. Copy the OMNIBus-response.txt file and rename it appropriately. You can store this file in the same location as the extracted installation files or in another location.
2. Edit the configuration values in your copy of the response file as follows. Do not add spaces before or after the values that you specify.

INSTALLER_UI

Do not change this configuration value from the default SILENT setting.

LICENSE_ACCEPTED

Set this value to true to indicate your acceptance of the licence agreement. If you run the installer with this value set to false, the installation process terminates.

USER_INSTALL_DIR

Specify the location to which you want to install Tivoli Netcool/OMNIBus.

CHOSEN_INSTALL_SET

Specify the installable features as follows:

- To install all the features, leave the following lines commented out, as given by default:

```
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
```

- To install a subset of the features:
 - a. Uncomment the lines beginning:


```
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
```
 - b. Leave the value of **CHOSEN_INSTALL_SET** as Custom.
 - c. Delete any features that you do not want to install from the list of comma-separated values given for **CHOSEN_INSTALL_FEATURE_LIST**. You must delete the `nco_` value and the comma that follows. Spaces are not required in this list, and the last value does not require a comma.

SKIP_DE_PRECHECKS

Controls whether the installation is terminated if one of the Deployment Engine (DE) prechecks is failed. Possible values are as follows:

- **true**: If the installation fails the DE prechecks, the installation continues.
- **false**: If the installation fails any of the DE prechecks, the installation is terminated and a warning message is sent to the log file.

The DE prechecks might be failed depending on whether you are installing as root or a non-root user, and on whether a root instance of the DE has already been installed. The following table describes the conditions under which a precheck might be failed, depending on which user is installing the product.

Table 15. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

DE_SECURITY_MODE

When you install as root or as an Administrative user, a root instance of the Deployment Engine (DE) is installed on the server. You can select the user access policy to apply to this global instance of the DE by selecting an option for the **DE_SECURITY_MODE** parameter. Alternatively, you can skip this step and change the DE access policy at any time after installation by using the **de_security** script.

Valid options for **DE_SECURITY_MODE** are as follows:

- 0 - No change will be made (default).
- 1 - Single user (current user).
- 2 - Group (current user plus members of an existing user group).
- 3 - Global (all users).

If you use the 'Group' security mode (option 2), you must set the **DE_GROUP_NAME** parameter to a valid user group.

Note: The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

3. Save the response file.

Related concepts

"Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)" on page 45

Running the installation program with the silent mode settings (UNIX and Linux)

After you create the response file that defines which features you want to install, run the installer in silent mode.

Note: No configuration options are displayed during the upgrade. You can cancel the process by pressing Ctrl+C.

To install Tivoli Netcool/OMNIBus in silent mode:

1. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
2. Enter the following command to run the installation program:

```
./install.bin -i silent -f full_path_to_filename
```

The *full_path_to_filename* value defines the full path and file name of the response file that contains your installation settings.
3. Wait for the installation to complete; a message confirms that the installation is complete.

If you have set the value of the **SKIP_DE_PRECHECK** parameter to false in the response file, the installer behaves as described in the following table.

Table 16. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

If you want the installer to perform the action in response to which the installation was terminated, set the value of the **SKIP_DE_PRECHECK** parameter to true and rerun the installation

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related tasks

“Viewing the installation log files (UNIX and Linux)” on page 60

“Performing postinstallation tasks (UNIX and Linux)” on page 94

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

“Installation error messages” on page 613

Verifying the Netcool/OMNIBus installation (UNIX and Linux)

The installation process results in a package installation of the Tivoli Netcool/OMNIBus components. After you have installed the Tivoli Netcool/OMNIBus server-side components, run the **nco_objserv -version** utility to verify that the ObjectServer and its subcomponents have been successfully installed.

To verify that the ObjectServer and its subcomponents have been successfully installed:

1. Go to the \$NCHOME/omnibus/bin directory.
2. Enter the following command to run the utility:

```
./nco_objserv -version
```

The following example shows the output of the command and confirms a successful installation.

Note: The revision and version numbers shown in this example will be specific to your installation.

Netcool/OMNIBus Object Server - Version 7.3.1

(C) Copyright IBM Corp. 1994, 2007

Code Revision: 3.4.5

Library Revisions:
libnetcool: 3.4.5
libncmd: 3.4.5
libnregion: 3.4.5
libnmemstore: 3.4.5
libnoam: 3.4.5
libnsecurity: 3.4.5
libnstore: 3.4.5
libnproc: 3.4.5
libnauto: 3.4.5
libnipc: 3.4.5
libnstk: 3.4.5
libniduc_server: 3.4.5
libnobjserv: 3.4.5
network::ipv6: 5.2010.0311

Compilation Date: Wednesday April 7 18:35:32 BST 2010

Compilation Machine: SunOS 5.9 sparc

Compilation System: sol9-build1

Code Generation: PRODUCTION

Registered debug facilities:
nco_objserv_profiler_timings[ON], signal[ON], thread[ON], cmd[ON],
region[ON], nco_objserv_profiler[ON], ipc_s_rpc[ON], memstore[ON],
nco_objserv[ON], clock[ON], timer[ON], sec_author[ON], prop_mgr[ON],
cb_mgr[ON], arg_mgr[ON], ipc_s_res[ON], noam[ON], nco_objserv_vvtr[ON],
ipc_s_not[ON], store[ON], auto[ON], proc[ON], region_mutation[ON],
sec_audit[ON], module[ON], ipc_s_ini[ON], ipc_s_mut[ON], ipc_s_evt[ON]
2010-05-12T10:03:48: Debug: D-0BJ-105-003: ObjectServer shutdown
handler invoked.

If the command fails to generate the above output, it might indicate one or more components have not installed successfully. Check the installation log files and review the installation messages to identify any problems with the installation.

Related tasks

“Viewing the installation log files (UNIX and Linux)”

Viewing the installation log files (UNIX and Linux)

The installation process generates a set of log files for the Tivoli Netcool/OMNIBus and Deployment Engine installations. You can use these files to verify that you installed Tivoli Netcool/OMNIBus successfully, or to troubleshoot your installation.

The installation log files are saved to different locations depending on the user who installs the product.

To view the installation log files:

- Open the InstallAnywhere log file for Tivoli Netcool/OMNIBus in the following location:

Table 17. InstallAnywhere log file locations

Directory location	Description
/IA-Netcool-OMNIBus-hostname-yyyyymmddThhmmss-0.log	Location where the log file is stored when the product is installed by a root user. This is the home directory of the root user.
/home/username/IA-Netcool-OMNIBus-hostname-yyyyymmddThhmmss-0.log	Location where the log file is stored when the product is installed by a non-root user, where <i>username</i> is the name of the user.

In the file name, *yyyyymmddThhmmss* is the date and time the log file was first generated. Additional log files can be generated on subsequent modifications to the installation.

- Open the Tivoli Netcool/OMNIBus installation log file in the following location:
\$NCHOME/OMNIBus_InstallLog.log
- Open the Deployment Engine log files in the following location:

Table 18. Deployment Engine log file locations

Directory location	Description
/usr/ibm/common/acsi/logs/root	Location where the log files are stored when the product is installed by a root user.
/home/username/.acsi_hostname/logs/username	Location where the log files are stored when the product is installed by a non-root user, where <i>username</i> is the name of the user.

Note: The default logging level is set to DEBUG_MIN. However, in order to provide IBM Support with more detailed logging information, we recommend that the logging level is set to DEBUG_MAX.

To set the logging level to DEBUG_MAX:

- Open the <DE_Common_Dir>/ACULogger.properties file for editing. Where <DE_Common_Dir> is the name of your common directory.
- Then edit the following line, replacing DEBUG_MIN with DEBUG_MAX:
acu.logger.level=DEBUG_MIN

Viewing installed packages (UNIX and Linux)

The installer installs products as packages and you can view the versions of all installed packages at any time. You might be asked for package information by IBM Software Support.

To view installed packages:

1. From a command prompt, enter the following command:

Table 19. Root and non-root locations for the **listIU** command (UNIX and Linux)

User type	Command
If you installed as a root user	/usr/ibm/common/acsi/bin/listIU.sh
If you installed as a non-root user	/home/username/.acsi_username/bin/ listIU.sh In this command, <i>username</i> is the name of the non-root user.

The commands in the preceding table replace the **nco_id** utility. The **nco_id** utility now displays only the version number of Tivoli Netcool/OMNIbus. The list of installed packages and their versions is displayed.

2. To generate more detailed package information, in the form of an XML file, from the same location enter the following command: **de_lsrootiu.cmd**. A file named **package.xml** is created. The file can either be examined or submitted to IBM Software Support.

Results

The list of installed packages, either presented on the command-line interface or written to the **package.xml** file, is formatted as follows: **VV.RR.MM.FF**, where **VV** represents the version number, **RR** represents the release number, **MM** represents the modification number and **FF** represents the fix pack number. Because a fix pack might not update all the installed components, the fix pack levels can vary. The following example shows a sample output of the command:

```
IU UUID: 90A4E4492B83AEBAB0A24E2ADFCDA0E7 Name: SIU-ProbeSupport Version: 7.3.1.2
IU UUID: E852091AEE2609406F9E5EBC6C27BCB3 Name: SIU-Gateways Version: 7.3.1.1
IU UUID: 374A0E3526BE4C3E96F81987322C90F5 Name: SIU-Desktop Version: 7.3.1.0
```

Installation directory structure (UNIX and Linux)

Packages are installed in various locations in the Netcool home directory (\$NCHOME) during the Tivoli Netcool/OMNIbus installation.

Tip: In these tables, *arch* is a variable depicting an operating system directory.

Packages common to products installed in the same \$NCHOME location

The following table describes the directories for common packages that are shared by products installed in the same Netcool home directory.

Table 20. Directories for common packages

Directory location	Description
\$NCHOME/bin	Location of the Netcool portfolio binary files, including the iKeyman utilities, and the nco_run script and links that run common applications.

Table 20. Directories for common packages (continued)

Directory location	Description
\$NCHOME/etc	Location of the configuration files that are generated or used by common applications or third-party products, and the localization configuration file (tds.dat). You can modify these files.
\$NCHOME/etc/default	Location of read-only default reference versions of the localization configuration file (tds.dat) and other configuration files.
\$NCHOME/etc/security	Location of the FIPS 140–2 configuration file (fips.conf) that is required for FIPS 140–2 initialization on Tivoli Netcool/OMNIBus.
\$NCHOME/etc/security/keys	Location of the key database files that are created for managing digital certificates and Secure Sockets Layer (SSL) connections.
\$NCHOME/license	Location of IBM and non-IBM license files.
\$NCHOME/log	Location of the communication log file for the ObjectServer.
\$NCHOME/platform	Location of internal programs and libraries used by Tivoli Netcool/OMNIBus.
\$NCHOME/_uninst	Location of the files for uninstalling Tivoli Netcool/OMNIBus.
\$NCHOME/var	Location of the gateway log files.

Tivoli Netcool/OMNIBus packages

The following table describes the directories that are specific to Tivoli Netcool/OMNIBus.

Table 21. Tivoli Netcool/OMNIBus directories

Directory location	Description
\$NCHOME/omnibus/bin	Location of the nco_run script and links that run Tivoli Netcool/OMNIBus applications. This location also holds the IEHS executable files for starting and stopping an IEHS server that is running locally in standalone mode, or in information center mode.
\$NCHOME/omnibus/db	Location of the ObjectServer database files.
\$NCHOME/omnibus/desktop	Location of the resource files for the Desktop component. Restriction: The desktop directory is not available on a Linux on System z or HP-UX Integrity installation.
\$NCHOME/omnibus/etc	Location of the configuration files that the database initialization utility (nco_dbinit) requires to create an ObjectServer, and configuration files that can be used to upgrade the database schema. This location also holds properties files, and the configuration file for setting the values to run the online help system in information center mode. You can modify these files.
\$NCHOME/omnibus/etc/default	Location of read-only default reference versions of the properties files, and configuration files that are used by the nco_dbinit utility and online help utilities.

Table 21. Tivoli Netcool/OMNibus directories (continued)

Directory location	Description
\$NCHOME/omnibus/etc/initial	Location of the writable copy of the ObjectServer source properties file (NCOMS.props), which is used by nco_dbinit .
\$NCHOME/omnibus/etc/locale	Location of the translated desktop SQL definition files for each supported language. The desktop SQL definition file inserts default values into the desktop tables, including default colors, column visuals, conversions, tools, and menus.
\$NCHOME/omnibus/extensions	Location of resources that you can use to extend the functionality of Tivoli Netcool/OMNibus.
\$NCHOME/omnibus/install	Location of the installation resources for probes and gateways. Also holds the startup script that automatically runs the process control daemon on system startup.
\$NCHOME/omnibus/java	Location of .jar files that support Java applications.
\$NCHOME/omnibus/log	Location of the majority of the ObjectServer log files. (The ObjectServer communication log file is in \$NCHOME/log.)
\$NCHOME/omnibus/patches	Location of data required by the patching system.
\$NCHOME/omnibus/platform	Location of platform-dependent resources such as Tivoli Netcool/OMNibus catalogs, libraries, modules, and IEHS files.
\$NCHOME/omnibus/tsm	Location where TSMs are installed. Required for backward compatibility with the probe installer.
\$NCHOME/omnibus/upgrade	Location of the Tivoli Netcool/OMNibus upgrade script UPGRADE.SH, which migrates configuration data from a previous installation into a V7.3.1 installation.
\$NCHOME/omnibus/utls	Location of the nco_mail and nco_functions utilities. Can be used to store similar utilities used by external automations and tools.
\$NCHOME/omnibus/var	Location where internal runtime information is stored.

Probes

The following table describes the probe directory.

Table 22. Probes directory

Directory location	Description
\$NCHOME/omnibus/probes	Location where probes are installed.

Gateways

The following table describes the gateway directory.

Table 23. Gateways directory

Directory location	Description
<code>\$NCHOME/omnibus/gates</code>	Location where configuration data for new ObjectServer gateways is stored.

Deployment Engine

The Deployment Engine files are saved to different locations depending on the user who installs the product. The following table describes the Deployment Engine directories.

Table 24. Deployment Engine directories

Directory location	Description
<code>/var/ibm/common/acsi</code>	Location where the Deployment Engine data files are stored when the product is installed by a root user.
<code>/usr/ibm/common/acsi</code>	Location where the Deployment Engine scripts are stored when the product is installed by a root user.
<code>/home/username/.acsi_hostname</code>	Location where the Deployment Engine files are stored when the product is installed by a non-root user. The <i>username</i> is the name of the logged-in, non-root user.

Upgrading on UNIX and Linux

On UNIX and Linux systems, you can run the upgrade program for Tivoli Netcool/OMNIBus by using the installation wizard, or the console or silent installation mode.

A Tivoli Netcool/OMNIBus upgrade includes any of the following actions:

- Installing a new version of Tivoli Netcool/OMNIBus V7.3.1 into an environment with a previous installation of V7.1, V7.2, V7.2.1, or V7.3. This includes running the installation program and then migrating your old data into the new installation.

Note: When upgrading from Tivoli Netcool/OMNIBus V7.3.0 to V7.3.1, the files are installed within the existing V7.3.0 directories. To perform the upgrade: stop all the Tivoli Netcool/OMNIBus processes that are currently running; run your normal backup policy on both the Deployment Engine and Tivoli Netcool/OMNIBus; run the V7.3.1 installation process and specify the same file location as the current V7.3.0 installation; make any necessary configuration changes; and then restart the system.

- Modifying your Tivoli Netcool/OMNIBus V7.3.1 installation. This involves adding features to an existing V7.3.1 installation.

Note: If you want to upgrade from V3.6, first upgrade to V7.2.1, and then upgrade to V7.3.1. If you want to upgrade from V7.0, upgrade first to V7.3 and then upgrade to V7.3.1. For information about upgrading from V3.6 to V7.2.1, and from V7.0 to V7.3, see the Tivoli Netcool/OMNIBus V7.2.1 documentation and the V7.3

documentation in the IBM Tivoli Network Management Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>.

You can still continue to run your existing V7.2.1, or earlier, installation, and can run both the old installation and the newly-upgraded system in parallel. You must, however, ensure that the set of ports used in the two installations is different. You can change the ports in the newly-upgraded system by editing the default values in the data connections file named `$NCHOME/etc/omni.dat` file. After changing the ports, run `$NCHOME/bin/nco_igen` to generate the interfaces file that stores server communication information.

Tivoli Netcool/OMNIBus V7.2.1, or later, can operate in FIPS 140–2 mode. If you want to upgrade your current installation so that it runs in this mode, some initial configuration of your existing data might be required either before or after you upgrade.

Related tasks

“Manually editing the connections data file” on page 247

“Generating the interfaces file for multiple platforms (UNIX only)” on page 249

Related reference

Appendix D, “Default port numbers used by Tivoli Netcool/OMNIBus,” on page 631

Guidelines for upgrading to FIPS 140–2 mode (UNIX and Linux)

If you want to upgrade to V7.3.1 in FIPS 140–2 mode, some initial configuration is required if the user passwords in your system are currently encrypted by using the DES algorithm, or if you are using property value encryption to encrypt string values in properties files.

If you do not use DES-encrypted user passwords or property value encryption, you can upgrade as normal and then configure your system for FIPS 140–2 mode.

Upgrading from an installation with DES-encrypted user passwords (UNIX and Linux)

When in FIPS 140–2 mode, the Advanced Encryption Standard (AES) algorithm must be used to encrypt user passwords that are stored in the ObjectServer.

If your existing installation uses DES encryption for passwords, you must change the encryption scheme to AES either before or after upgrading. You can then configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.

If you are running Tivoli Netcool/OMNIBus V7.1 or later, the encryption algorithm is either DES or AES. Check the value of the ObjectServer **PasswordEncryption** property to see whether it is set to DES or to AES.

To upgrade to V7.3.1 in FIPS 140–2 mode, perform the following set of actions *after* upgrading:

When	Actions
Before upgrading (Applies only to V7.0.1.7, or later)	<ol style="list-style-type: none"> 1. In your existing installation, change the setting of the ObjectServer PasswordEncryption property to AES. 2. Ensure that all user passwords are changed or reset. The passwords are now AES encrypted. (See the information that follows this table for guidelines about how to change or reset passwords.) 3. Upgrade to V7.3.1. 4. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.
After upgrading (Applies to all versions earlier than V7.3.1)	<ol style="list-style-type: none"> 1. Upgrade to V7.3.1 2. In the V7.3.1 system, change the setting of the ObjectServer PasswordEncryption property to AES. 3. Ensure that all user passwords are changed or reset. The passwords are now AES encrypted. (See the information that follows this table for guidelines about how to change or reset passwords.) 4. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode. 5. Restart Tivoli Netcool/OMNIBus.

Guidelines for changing or resetting passwords

You can use the SQL interactive interface (**nco_sql**) for changing or resetting passwords.

If you ask users to change their passwords, you must to verify that the changes have been made, and will most likely have to send out reminders. To verify whether all passwords have been changed or to identify which ones still need to be changed, perform either of the following steps:

- Start the SQL interactive interface and then enter the following command:

```
select UserName,Passwd from security.users;
```

Check the length of the encrypted passwords returned. Passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

For information about starting the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.
- From Netcool/OMNIBus Administrator:
 1. Connect to the relevant ObjectServer. Then click the **System** menu button and click **Databases** to open the Databases, Tables and Columns pane.
 2. Select the **security** database and the **users** table, and then click the **Data View** tab in the Databases, Tables and Columns pane to view user data.

In the **Passwd** column, passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

A system administrator can reset user passwords from the SQL interactive interface as follows:

```
alter user 'username' set password 'password';
```

Where *username* is the name of the user and *password* is their new password.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related reference

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102

Upgrading from an installation with property value encryption (UNIX and Linux)

When in FIPS 140–2 mode, property value encryption must be performed by using an algorithm and mode of operation defined as AES_FIPS. Property value encryption is used to encrypt string values in a properties file or configuration file so that the strings cannot be read without a key.

Tip: You can skip this task if you are upgrading from a V7.2.1 system that operates in FIPS 140-2 mode.

If your existing installation uses property value encryption with the AES algorithm, or uses the **nco_g_crypt** and **nco_pa_crypt** utilities to encrypt passwords, these encrypted values do not meet the requirements for FIPS 140–2 operation. (In V7.2, or earlier, the AES algorithm is used by default for property value encryption; in V7.2.1, AES can be explicitly specified.) To run your upgraded system in FIPS 140–2 mode, you must decrypt these values and then encrypt them again by using the AES_FIPS algorithm. You must perform this task for each ObjectServer, proxy server, process agent, probe, and gateway that uses encrypted property values, including passwords.

To upgrade property value (and password) encryption for FIPS 140–2 mode, follow these guidelines:

1. In your existing installation, identify any keys that were generated by using the command-line key generator **nco_keygen**.

Tip: The **nco_keygen** utility stores keys within key files. You should be able to identify any key files used by checking the **ConfigKeyFile** property settings in your properties files.

2. Using the keys in your existing installation, decrypt all encrypted properties and passwords in your properties and configuration files by running the **nco_aes_crypt** utility with the **-d** command-line option.
3. Upgrade to V7.3.1.
4. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.
5. Encrypt the values again by using the **nco_keygen** utility to generate one or more new keys, and then running the **nco_aes_crypt** utility with the relevant key file setting.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related reference

“Property value encryption” on page 356

“nco_aes_crypt command-line options” on page 358

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102

Upgrading using the installation wizard (UNIX and Linux)

Use the wizard to present upgrade options within wizard pages in a graphical user interface. In this mode, you can choose to automatically migrate data from a previous installation during the upgrade process.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIBus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIBus or the Web GUI.

Note: For pre-V7.3.0 installations, if you attempt to upgrade into the same directory as an existing installation, the wizard provides a warning that the directory already exists, and offers to move your existing installation into a backup location. If using the wizard, there is therefore no need for you to take any preliminary action before starting the upgrade.

To upgrade Tivoli Netcool/OMNIBus:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running. If you are upgrading from V7.2 or later, also ensure that you shut down the IBM Eclipse Help System (IEHS) server from the command line as follows:
 - If you are running the IEHS server in standalone mode, enter the following command on the local computer:
`$NCHOME/omnibus/bin/help_end`
 - If you are running the IEHS server in information center mode on a remote server, enter the following command on the server computer:
`$NCHOME/omnibus/bin/IC_end`
2. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
3. Either start the installation by using the launchpad, or directly run the command to start the installation program:
 - Enter the following command to start the launchpad:
`./launchpad.sh`
When the launchpad window opens, select a language, and then click through each of the following options in the left navigation pane in order to review the welcome information, prerequisite information, and installation scenarios: Welcome, Prerequisite Information, and Installation Scenarios.
When you are ready to install Tivoli Netcool/OMNIBus, click the Install Product option in the left navigation pane. Then click the **Start Tivoli Netcool/OMNIBus Installation** button that is associated with the Tivoli Netcool/OMNIBus installation. When the IBM Tivoli Netcool/OMNIBus splash screen is displayed, proceed to the next step.
 - Enter the following command to start the upgrade process:
`./install.bin`
The JRE and installation resources are extracted from the installer archive, and the IBM Tivoli Netcool/OMNIBus splash screen is displayed.

Tip: When running the **install.bin** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent

installations. No value is required for `-r`, and the `installer.properties` file is generated in the directory that contains the **install.bin** command. The `-r` command-line option must be the last option specified.

4. From the drop-down list at the bottom of the splash screen, select a language and click **OK**.
5. If you are installing Tivoli Netcool/OMNIBus as a root user and there is no global Deployment Engine (DE) installed: a message informs you that if you install a global DE, it will be used by any user that attempts to install a DE based product (unless they currently have a single user DE). If you are installing Tivoli Netcool/OMNIBus as a non-root user and you do not have a single user Deployment Engine installed but there is a global Deployment Engine installed: a message informs you that the global DE will be used and that you must have write permissions to the database. You have the option to either terminate or proceed with the installation.
6. From the Introduction page of the installation wizard, click **Next** to proceed to the Software Licence Agreement page.

Tip: While configuring your installation options, you can click **Previous** to revisit previous pages of the wizard and make changes if required.

7. Read the license agreement and the non-IBM terms, and then accept both the IBM and non-IBM terms. You can also click **Print** to obtain a hardcopy of the license agreement.
8. Click **Next**. If you are installing as a root user, an information message is displayed at this stage. To confirm that you want to install as root, click **Yes**. Otherwise, click **No**, in which case the installation is terminated.
9. If you are installing as a root user, the Autonomic Deployment Engine page is displayed. From this page, choose the installation location for the Deployment Engine as follows:
 - Click **Install in recommended location** to accept the default installation location for the Deployment Engine. This is the preferred option. For a root user, the installation location for the Deployment Engine defaults to `/usr/ibm/common/acsi`.

Note: If you are installing as a non-root user, you do not see the Autonomic Deployment Engine page. By default, the Deployment Engine is installed in `/home/username/.acsi_hostname`, where *hostname* is the name of the logged-in user. If you are upgrading from Tivoli Netcool/OMNIBus V7.3.0, the current Deployment Engine will be located in `/home/username/.acsi_username`. Following the upgrade, the Deployment Engine will be located in `/home/username/.acsi_hostname`, and a symbolic link `.acsi_username` will point to the new directory.

- Click **Install into other location** to choose a different location. Only use this option for root installations on Solaris Sparse zones or other restricted systems where root cannot write to the `/usr` directory. To specify a location:
 - a. Click **Next**.
 - b. From the "Autonomic Deployment Engine - Choose a folder" page, specify an alternative installation location. You can click **Restore Default Folder** to revert to the default installation location.

Note: If the Deployment Engine is already installed on your computer, the existing installation location takes precedence over any alternative location that you specify. If you choose an alternative location, the Deployment Engine files are split between two locations: the database files are stored in

the `/var/ibm/commom/acsi` directory, and the remaining Deployment Engine resources are stored in your chosen location.

10. Click **Next** and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.
11. From the Select Destination Folder page, specify an installation location for the Tivoli Netcool/OMNIBus upgrade.

The installation location defaults to `/opt/IBM/tivoli/netcool`.

12. Click **Next** to accept the upgrade directory for Tivoli Netcool/OMNIBus. You obtain one of two results:
 - If you specified the same location as a previous installation, when you click **Next** on the Select Destination Folder page, the installer provides a warning about a detected version. The installer offers to move your old installation into another location, so that Tivoli Netcool/OMNIBus can be installed in the specified location. The NCHOME location of your previous installation directory is renamed; for example, `/opt/netcool` is renamed `/opt/netcool.1`. Click **Next** to indicate your consent, and then confirm that you want to rename your previous installation directory.
 - If you specified a different location from a previous installation, the Choose Install Set page opens.

Note: The Choose Install Set page also opens if you are upgrading from Tivoli Netcool/OMNIBus V7.3.0.

13. From the Choose Install Set page, choose either of the installation options by clicking the associated icon:
 - **Typical:** Choose this option to install all the Tivoli Netcool/OMNIBus features. Then click **Next**.
 - **Custom:** Choose this option if you want to select the features that you want to install. Then:
 - a. Click **Next**.
 - b. From the list box, select each feature that you want to install by selecting its associated check box, or deselect a feature by clearing the associated check box.
 - c. Click **Next** after selecting all the features that you want to install.

Tip: You can switch from a custom to a typical installation by selecting Typical from the **Install Set** drop-down list. Typical selects all features, whereas Custom reverts to your subset of selected features.

The Data Migration page is displayed, asking whether you want to migrate data from an existing Tivoli Netcool/OMNIBus installation.

14. Choose either of the following options:
 - **No:** Click this option if:

You are performing a fresh installation, or want to migrate your existing configuration data at a later stage by manually running an `UPGRADE.SH` script. Then click **Next** to proceed.

You are upgrading an existing Tivoli Netcool/OMNIBus V7.3.0 installation, as the upgrade is performed within the existing directories. Then click **Next** to proceed.
 - **Yes:** Click this option to migrate your data automatically. Then:
 - a. Click **Next**.
 - b. If you are upgrading from Tivoli Netcool/OMNIBus V7.2.1 or a previous installation, your old location was renamed by the wizard. The

renamed location (for example, `/opt/netcool.1`) is shown as the path from which data is migrated. You can alternatively specify a different location, but can also use the **Restore Default** button to switch back to the path identified by the wizard.

If you specified a different upgrade location from a previous installation, and want to migrate configuration data from any existing installation, you can specify the old location from which to migrate the data.

- c. Click **Next** to proceed.

After a short interval during which the system is configured, the Pre-Installation Summary page is displayed.

15. Review the installation settings and then click **Install** to start the installation. The Installing Netcool/OMNIBus page shows the progress of the installation. On completion, the Upgrade Complete - View Results page is displayed if you chose to migrate data from a previous installation. This page contains details of the migrated data. Click **Next** after reviewing the contents.
16. From the Installation Complete page, click **Done** to close the wizard.

If you started the installation program from the launchpad, you can return to the launchpad window, and click **Post-Installation** in the navigation pane to review postinstallation information. Then click **Exit** and confirm that you want to exit the launchpad.

Results

The upgrade adds a number of files to your system.

What to do next

You can open the installation log files to review the installation messages. Also review the migration log file to see if there were any problems. If you chose not to migrate your data, you can manually migrate your existing data into the new installation by running an `UPGRADE.SH` script.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIBus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)” on page 45

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

“Viewing the installation log files (UNIX and Linux)” on page 60

“Viewing the migration log file (UNIX and Linux)” on page 80

“Manually migrating data (UNIX and Linux)” on page 80

“Performing postinstallation tasks (UNIX and Linux)” on page 94

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

“Additional upgrade and migration notes (UNIX and Linux)” on page 81

Upgrading in console mode (UNIX and Linux)

Use the console mode to present the upgrade options as a series of menus and prompts in a text-based user interface. In this mode, you can choose to automatically migrate data from a previous installation during the upgrade process.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIBus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIBus or the Web GUI.

If you attempt to upgrade into the same directory as an existing installation, the wizard provides a warning that the directory already exists, and offers to move the existing installation into a backup location. When using this mode, you do not need to take any preliminary action before starting the upgrade.

Tip: During the upgrade, you can enter quit from most of the menu screens to exit the installer. You can also enter back from some of the menu screens to return to the previous screen.

To upgrade Tivoli Netcool/OMNIBus in console mode:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running.
2. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
3. Enter the following command to start the upgrade process:

```
./install.bin -i console
```

Wait while the JRE and installation resources are extracted from the installer archive.

Tip: When running the **install.bin** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.bin** command. The **-r** command-line option must be the last option specified.

4. Enter a number that corresponds to the language you want to use for the upgrade procedure.
5. If you are installing Tivoli Netcool/OMNIBus as a root user and there is no global Deployment Engine (DE) installed: a message informs you that if you install a global DE, it will be used by any user that attempts to install a DE based product (unless they currently have a single user DE). If you are installing Tivoli Netcool/OMNIBus as a non-root user and you do not have a single user Deployment Engine installed but there is a global Deployment Engine installed: a message informs you that the global DE will be used and that you must have write permissions to the database. You have the option to either terminate or proceed with the installation.
6. Read the Introduction information and press Enter, as prompted.
7. To read the non-IBM terms, press 4, and then press Enter to scroll through the text. After reading the non-IBM terms, press q to return to the license agreement. Press Enter to scroll through the license agreement, and then enter 1 to accept the agreement.
8. If you are upgrading as a root user, you are asked to confirm whether you want to install as root. Enter 1 as confirmation, and go to step 9. Otherwise, enter 2 to stop the installer so that you can upgrade as a non-root user.
9. If you are installing as a root user, choose the installation location of the Deployment Engine as follows:
 - Enter 1 to accept the default installation location for the Deployment Engine. This is the preferred option. For a root user, the installation location for the Deployment Engine defaults to `/usr/ibm/common/acsi`.

Note: If you are installing as a non-root user, you are not prompted for a location. By default, the Deployment Engine is installed in `/home/username/.acsi_hostname`, where *username* is the name of the logged-in user. If you are upgrading from Tivoli Netcool/OMNIBus V7.3.0, the current Deployment Engine will be located in `/home/username/.acsi_username`. Following the upgrade, the Deployment Engine will be located in `/home/username/.acsi_hostname`, and a symbolic link `.acsi_username` will point to the new directory.

- Enter 2 to choose a different location. Then specify an alternative installation location. Only use this option for root installations on Solaris Sparse zones or other restricted systems where root cannot write to the `/usr` directory.

Note: If the Deployment Engine is already installed on your computer, the existing installation location takes precedence over any alternative location that you specify. If you choose an alternative location, the Deployment Engine files are split between two locations: the database files are stored in the `/var/ibm/commom/acsi` directory, and the remaining Deployment Engine resources are stored in your chosen location.

10. Press Enter and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.
11. Specify an upgrade location for Tivoli Netcool/OMNIBus.
The upgrade location defaults to `/opt/IBM/tivoli/netcool`.
12. When you press Enter to accept the upgrade directory, you obtain one of two results:
 - If you specified the same location as a previous installation, you obtain a warning about a detected version in that location. The installer offers to

move your old installation into another location, so that Tivoli Netcool/OMNIbus can be installed in the specified location. The NCHOME location of your previous installation directory is renamed; for example, /opt/netcool is renamed /opt/netcool.1. Press Enter to indicate your consent, and then press Enter again to confirm that you want to rename your previous installation directory.

- If you specified a different location from a previous installation, the Choose Install Set menu is shown.

Note: The Choose Install Set page also opens if you are upgrading from Tivoli Netcool/OMNIbus V7.3.0.

13. Choose the type of installation to perform:

- Enter 1 to install all the Tivoli Netcool/OMNIbus features.
- Enter 2 to select the features that you want to install.

In the resulting menu, all features are initially selected, as denoted by [X], so enter a comma-separated list of numbers that correspond to the features you do *not* want to install. (Note that this screen has a toggle function that enables you to type a number and press Enter to deselect a selected feature denoted by [X], or to type a number and press Enter to select a deselected feature denoted by [].)

Your list of selected features is then shown. You can either enter back to return to the previous screen and revise your selection, or press Enter to confirm your selection and continue.

14. Enter 1 if you want to migrate data from an existing Tivoli Netcool/OMNIbus installation as part of the upgrade process. If you prefer to manually migrate the data after the upgrade process, enter 2.
15. If you chose to migrate data from a previous installation, specify the old location from which to migrate the data. If you chose to upgrade to the same location as a previous installation, and your old location was renamed by the installer, the renamed location (for example, /opt/netcool.1) is shown as the default path from which data will be migrated.
16. When the pre-installation summary is displayed, review the information and then press Enter to start the upgrade. On completion, the Upgrade Complete - View Results screen is displayed if you chose to migrate data from a previous installation. This screen contains details of the data that was migrated. Press Enter after reviewing the contents.
17. Press Enter to exit the installer.

What to do next

You can open the installation log files to review the installation messages. Also review the migration log file to see if there were any problems. If you chose not to migrate your data, you can manually migrate your existing data into the new installation by running an UPGRADE.SH script.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIbus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)” on page 45

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

“Viewing the installation log files (UNIX and Linux)” on page 60

“Viewing the migration log file (UNIX and Linux)” on page 80

“Manually migrating data (UNIX and Linux)” on page 80

“Performing postinstallation tasks (UNIX and Linux)” on page 94

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

“Additional upgrade and migration notes (UNIX and Linux)” on page 81

Upgrading in silent mode (UNIX and Linux)

Run the upgrade in silent mode to propagate one configuration to multiple workstations. In silent mode, the installer suppresses the graphical or text interface and obtains the installation settings from a predefined response file.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIBus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIBus or the Web GUI.

Important: On Solaris Sparse zones, installation as a root user in silent mode works only on Solaris whole root zones. For root installations on Solaris Sparse zones or other restricted systems where root cannot write to the /usr directory, use the wizard or console mode. Using these modes, you can specify an alternative location for the Deployment Engine, which installs to the /usr directory by default.

The silent mode of installation has two parts:

1. Define your installation settings in a response file.
2. Run the installation program with the settings in this file.

Related tasks

“Obtaining the installation package (UNIX and Linux)” on page 47

Defining your upgrade settings in a response file (UNIX and Linux)

Before you can run the upgrade program in silent mode, create a response file that defines the features you want to install.

The installation package includes a sample response file that is located in the directory where you extracted the package. The file is called OMNIBus-response.txt. Make a copy of the sample file and use the copy to specify your installation options.

Note: If you previously ran the wizard or console installer with the -r command-line option in order to save your installation settings to an auto-generated installer.properties file, you can use this file as your response file.

To create a response file with your preferred upgrade options:

1. Copy the `OMNibus-response.txt` file and rename it appropriately. You can store this file in the same location as the extracted installation files or in another location.
2. Edit the configuration values in your copy of the response file as follows. Do not add spaces before or after the values that you specify.

INSTALLER_UI

Do not change this configuration value from the default `SILENT` setting.

LICENSE_ACCEPTED

Set this value to `true` to indicate your acceptance of the licence agreement. If you run the installer with this value set to `false`, the installation process terminates.

USER_INSTALL_DIR

Specify the location to which you want to install Tivoli Netcool/OMNIBus.

CHOSEN_INSTALL_SET

Specify the installable features as follows:

- To install all the features, leave the following lines commented out, as given by default:
`#CHOSEN_INSTALL_SET...`
`#CHOSEN_INSTALL_FEATURE_LIST...`
- To install a subset of the features:
 - a. Uncomment the lines beginning:
`#CHOSEN_INSTALL_SET...`
`#CHOSEN_INSTALL_FEATURE_LIST...`
 - b. Leave the value of **CHOSEN_INSTALL_SET** as `Custom`.
 - c. Delete any features that you do not want to install from the list of comma-separated values given for **CHOSEN_INSTALL_FEATURE_LIST**. You must delete the `nco_` value and the comma that follows. Spaces are not required in this list, and the last value does not require a comma.

SKIP_DE_PRECHECKS

Controls whether the installation is terminated if one of the Deployment Engine (DE) prechecks is failed. Possible values are as follows:

- `true`: If the installation fails the DE prechecks, the installation continues.
- `false`: If the installation fails any of the DE prechecks, the installation is terminated and a warning message is sent to the log file.

The DE prechecks might be failed depending on whether you are installing as root or a non-root user, and on whether a root instance of the DE has already been installed. The following table describes the conditions under which a precheck might be failed, depending on which user is installing the product.

Table 25. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

DE_SECURITY_MODE

When you install as root or as an Administrative user, a root instance of the Deployment Engine (DE) is installed on the server. You can select the user access policy to apply to this global instance of the DE by selecting an option for the **DE_SECURITY_MODE** parameter. Alternatively, you can skip this step and change the DE access policy at any time after installation by using the **de_security** script.

Valid options for **DE_SECURITY_MODE** are as follows:

- 0 - No change will be made (default).
- 1 - Single user (current user).
- 2 - Group (current user plus members of an existing user group).
- 3 - Global (all users).

If you use the 'Group' security mode (option 2), you must set the **DE_GROUP_NAME** parameter to a valid user group.

Note: The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

3. If you want to migrate data from an existing location as part of the upgrade process, add the following configuration setting to the response file:

IAGLOBAL_NCHOME_MIGRATE=omnibus_backup_location

Use *omnibus_backup_location* to specify the path of the existing installation location from which the data is to be migrated:

- If you intend to upgrade to the same location as your existing installation, you need to move the existing Tivoli Netcool/OMNIBus directories and all their contents to a backup location, as described in “Running the upgrade program with the silent mode settings (UNIX and Linux).” The value of *omnibus_backup_location* must be this backup location.
- If you intend to upgrade to a different location from your existing installation, the value of *omnibus_backup_location* must be the path of the existing \$NCHOME location.

Note: If you do not specify the **IAGLOBAL_NCHOME_MIGRATE** setting, you must manually migrate your data after upgrading.

4. Save the response file.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (UNIX and Linux)” on page 45

Running the upgrade program with the silent mode settings (UNIX and Linux)

After you create the response file that defines which features you want to install, run the installer in silent mode.

Note: No configuration options are displayed during the upgrade. You can cancel the process by pressing Ctrl+C.

To upgrade Tivoli Netcool/OMNIBus in silent mode:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running.
2. If you want to use the same location as a previous version, copy the existing Tivoli Netcool/OMNIBus directories and all their contents to a backup location by using the following command: `mv $NCHOME omnibus_backup_location`

In the command, substitute the environment variable with your actual installation location; for example, the default \$OMNIHOME location is /opt/netcool/omnibus and the default \$NCHOME location is /opt/netcool. Also replace *omnibus_backup_location* with your preferred backup location.

Tip: Make a note of the backup location for data migration purposes.

3. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
4. Enter the following command to run the installation program:
`./install.bin -i silent -f full_path_to_filename`
 The *full_path_to_filename* value defines the full path and file name of the response file that contains your installation settings.
5. Wait for the installation to complete; a message confirms that the installation is complete.

If you have set the value of the **SKIP_DE_PRECHECK** parameter to false in the response file, the installer behaves as described in the following table.

Table 26. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

If you want the installer to perform the action in response to which the installation was terminated, set the value of the **SKIP_DE_PRECHECK** parameter to **true** and rerun the installation

Results

The upgrade adds a number of files to your system.

What to do next

After the installation is complete, you can open the installation log files to review the installation messages. If you did not specify the **IAGLOBAL_NCHOME_MIGRATE** setting in the response file, manually migrate your existing data into the new installation by running an **UPGRADE.SH** script.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIbus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related tasks

“Viewing the installation log files (UNIX and Linux)” on page 60

“Manually migrating data (UNIX and Linux)”

“Performing postinstallation tasks (UNIX and Linux)” on page 94

Related reference

“Installation directory structure (UNIX and Linux)” on page 61

Manually migrating data (UNIX and Linux)

After upgrading Tivoli Netcool/OMNIBus, you must migrate your existing data into the new V7.3.1 installation if not already done. This task is relevant only if you ran the upgrade program in console or silent mode, or if you chose not to automatically migrate your data when running the wizard.

Note: The following procedure is not required if you are upgrading from Tivoli Netcool/OMNIBus V7.3.0 and the V7.3.1 installation files are extracted to the original V7.3.0 installation directory. If you have extracted the V7.3.1 installation files to a different location, you must run the `UPGRADE.SH` command in order to migrate the V7.3.0 data to the V7.3.1 installation directories.

To migrate old data into Tivoli Netcool/OMNIBus V7.3.1:

1. Go to the `$NCHOME/omnibus/upgrade` location.
2. Run the following command. The square brackets depict an optional command-line option.

```
UPGRADE.SH -old OLD_PATH -new NEW_PATH [-log LOG_FILE_PATH]
```

In this command, `OLD_PATH` is the location to which you backed up your old installation, and `NEW_PATH` is the new `$NCHOME/omnibus` location. If you include the `-log` command-line option, then `LOG_FILE_PATH` is the location to which you want the upgrade log file to be saved. If you omit the `-log` command-line option, then the log file details are output directly to the screen.

Related reference

“Additional upgrade and migration notes (UNIX and Linux)” on page 81

Viewing the migration log file (UNIX and Linux)

After upgrading Tivoli Netcool/OMNIBus, and migrating your existing data into the new installation, you can review the migration log file to verify whether the process was successful, or for troubleshooting purposes.

If you chose to automatically migrate your data during the upgrade process, you can view a copy of the log that is stored in `$NCHOME/omnibus/log/migrate.log`.

If you ran the upgrade program, and then manually migrated your data by using the `UPGRADE.SH` script with the `-log` command-line option, the log file is stored in the location that is specified by the `-log` command-line option.

Modifying your V7.3.1 installation (UNIX or Linux)

You can modify your Tivoli Netcool/OMNIBus V7.3.1 installation if you want to install additional components.

Note: To add features to your installation, run the installation again on your NCHOME location and choose which features you want to add. You cannot remove existing features.

To change the set of Tivoli Netcool/OMNIBus V7.3.1 features in an existing installation:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running.
2. Back up the current \$NCHOME directory in case you want to revert to that installation.
3. Run the installation program in GUI, console, or silent mode. If running in silent mode, update the response file with the features to be added before running the installer.
4. Review the installation log file.

Results

Where relevant:

- Any existing packages that are of a lower version are replaced with an equivalent higher version.
- Packages for any new features are installed.

What to do next

Before attempting to run Tivoli Netcool/OMNIBus, you might need to perform some post-installation tasks, depending on the features added.

Additional upgrade and migration notes (UNIX and Linux)

Read these notes for additional information about the Tivoli Netcool/OMNIBus upgrade and migration process, and any actions you might be required to perform.

Notes on upgrading ObjectServer schemas to V7.3.1 schemas (UNIX and Linux)

After upgrading to Tivoli Netcool/OMNIBus V7.3.1, upgrade your ObjectServer schemas to the V7.3.1 schema.

Important: Follow these instructions on each ObjectServer instance that is upgraded from V7.0 (through an upgrade to V7.1), V7.1, V7.2, V7.2.1, or V7.3. In each case, ensure that the ObjectServer is running.

Four .sql import files are provided in Tivoli Netcool/OMNIBus V7.3.1 that contain the required schema changes:

- update70to71.sql: This file upgrades a V7.0 ObjectServer schema to a V7.1 schema
- update71to72.sql: This file upgrades a V7.1 ObjectServer schema to a V7.2 schema. Note that the V7.2 and V7.2.1 schemas are identical.
- update72xt073.sql: This file upgrades a V7.2 or V7.2.1 ObjectServer schema to a V7.3 schema.

- `update73to731.sql`: This file upgrades a V7.3 ObjectServer schema to a V7.3.1 schema.

These files are located in the `$NCHOME/omnibus/etc` directory.

Note that database initialization is *not* required after upgrading.

V7.0 to V7.1 schema upgrade

To upgrade a V7.0 ObjectServer schema to a V7.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.0 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'tmp/70to72Upgrade/NCOMS';
```

3. Review the `update70to71.sql` file:
 - Ensure that it is not altering configuration that you have already added or customized for your business.
 - Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer).

In particular, the `connection_watch_disconnect` and `connection_watch_connect` automations were changed in the V7.1 installation package.

After you upgrade the schema, the new triggers are imported as `connection_watch_disconnect2` and `connection_watch_connect2`, and are disabled by default. If you want to use the new triggers, enable them, and then disable the original `connection_watch_disconnect` and `connection_watch_connect` triggers that were available in V7.0.

4. After you have resolved all the conflicts between the import file and the upgraded ObjectServer instance, import the file to the ObjectServer using the `nco_sql` command. For example:

```
$NCHOME/omnibus/bin/nco_sql -user username -password password -server servername < update70to71.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.

5. After the import process is completed successfully, review the ObjectServer log file for any errors. If errors exist, identify the cause, and resolve the conflicts.
6. After all the conflicts are resolved, apply the V7.1 to V7.2 schema upgrade as described in the next section. Review the ObjectServer log file again for errors and determine whether the configuration of the system is acceptable. If not, revert to the backed-up image, make the necessary changes to the `update70to71.sql` file and reapply the file.

V7.1 to V7.2 or V7.2.1 schema upgrade

To upgrade a V7.1 ObjectServer schema to a V7.2 or V7.2.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.0 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'tmp/70to72Upgrade/NCOMS';
```

3. Review the update71to72.sql file:

- Ensure that it is not altering configuration that you have already added or customized for your business.
- Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer).

If you have created your own tools and added them to menus, check the tools.* tables for conflicts.

Several schema changes and new automations support functions in IBM Tivoli Network Manager IP Edition V3.7 (formerly Netcool Precision IP). If you are already using Network Manager, the changes might already have been added to the ObjectServer. If this is the case, remove the duplicated configuration from the update71to72.sql file.

4. After you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the **nco_sql** command. For example:

```
$NCHOME/omnibus/bin/nco_sql -user username -password password -server  
servername < update71to72.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.

5. After the import process is complete, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the update71to72.sql file and reapply the file.
6. After all the conflicts are resolved, apply the V7.2 to V7.3 schema upgrade, or V7.2.1 to V7.3 schema upgrade, as described in the next section. Review the ObjectServer log file again for errors and determine whether the configuration of the system is acceptable. If not, revert to the backed-up image, make the necessary changes to the update71to72.sql file and reapply the file.

V7.2 or V7.2.1 to V7.3 schema upgrade

To upgrade a V7.2 or V7.2.1 ObjectServer schema to a V7.3 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.2 or V7.2.1 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'tmp/70to72Upgrade/NCOMS';
```

3. Review the update72xto73.sql file:

- Ensure that it is not altering configuration that you have already added or customized for your business.
- Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer).

In particular, the deduplication and new_row automations were changed in the V7.3 installation package.

After you upgrade the schema, the new triggers are imported as deduplication_73 and new_row_73, and are disabled by default. If you want to use the new triggers, enable them, and then disable the original deduplication and new_row triggers that were available in V7.2 or V7.2.1.

4. When you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the `nco_sql` command. For example:

```
$NCHOME/omnibus/bin/nco_sql -user username -password password -server  
servername < update72xt073.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.

5. After the import process is complete, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the `update72xt073.sql` file and reapply the file.

V7.3 to V7.3.1 schema upgrade

To upgrade a V7.3 ObjectServer schema to a V7.3.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.3 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'tmp/70to72Upgrade/NCOMS';
```

3. Review the `update73to731.sql` file:
 - Ensure that it is not altering configuration that you have already added or customized for your business.
 - Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer).

In particular, the `disconnect_iduc_missed` trigger has been updated to increase the maximum number of `iduc_missed` signals to 100 before the client is disconnected. This trigger replaces the 7.3 `disconnect_iduc_missed` trigger. The 7.3 trigger should be disabled and the updated trigger enabled before it can be used.

4. After you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the `nco_sql` command. For example:

```
$NCHOME/omnibus/bin/nco_sql -user username -password password -server  
servername < update73xt0731.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.

5. After the import process is completed, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the `update73xt0731.sql` file and reapply the file.

Files migrated for an upgrade (UNIX and Linux)

When you upgrade your UNIX or Linux installation, a number of files are migrated from the old installation backup directory to the new V7.3.1 installation.

Important:

- Review all the migrated properties files:
 - Where file paths are specified for a property, update the path (if necessary) so that it references the correct location in the new installation, rather than the old installation from which the file was migrated.
 - If you used the \$OMNIHOME or \$NCHOME environment variable (rather than the expanded value of the variable), you do not need to make any changes because the environment variable automatically resolves to the new location.
- If your previous installation contained ObjectServer files, which were created as storage objects for log or report data, these logical files were stored in the ObjectServer database with a reference to the full directory path of the physical location. If your upgrade path is different from the path of the previous installation, check to see whether you have any file objects that reference the old location, and update the paths so that they reference the new location. You can check the paths from the catalog.files table. Alternatively, from the Netcool/OMNIBus Administrator window, select the **System** menu button, and then click **Log Files** to see the file details. For further information about the catalog.files table and Netcool/OMNIBus Administrator, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

The following table lists the migrated files and their locations in the new installation.

Table 27. Migrated file locations

File type	Migrated location
Connections data file	\$NCHOME/etc/omni.dat
Configuration files	\$NCHOME/omnibus/etc/*.conf \$NCHOME/omnibus/*/*.conf The upgrade copies only configuration files that use default names; for example, nco_pa.conf and *GATE.conf. Any other configuration files must be copied manually to the equivalent \$NCHOME/omnibus location.
ObjectServer gateway configuration files	\$NCHOME/omnibus/etc/*GATE.props \$NCHOME/omnibus/etc/*.tblrep.def \$NCHOME/omnibus/etc/*.map \$NCHOME/omnibus/etc/*.startup.cmd
Database files	\$NCHOME/omnibus/db
Netcool/OMNIBus Administrator properties file	\$NCHOME/omnibus/etc/nco_config.props
Policy file	\$NCHOME/omnibus/etc/admin.policy

Table 27. Migrated file locations (continued)

File type	Migrated location
Exclusions file	<p>\$NCHOME/omnibus/etc/exclusions.old.xml</p> <p>If you had previously made changes to the exclusions file in your old installation, you must copy these changes from the migrated exclusions.old.xml file into the \$NCHOME/omnibus/etc/exclusions.xml file in your new installation.</p>
Confpack properties file	\$NCHOME/omnibus/etc/nco_confpack.props
Desktop files	<p>\$NCHOME/omnibus/desktop/default.elc</p> <p>The V7.3 UNIX or Linux event list uses locale-specific default.elc and minimal.elc files in the location \$NCHOME/omnibus/desktop/locale/arch/locale.</p> <p>Note: Images and backdrops are no longer required and are not migrated.</p>
Key database files for SSL (V7.2.1)	\$NCHOME/etc/security/keys
FIPS 140-2 configuration file	\$NCHOME/etc/security/fips.conf
Utilities	\$NCHOME/omnibus/utlils
Probe properties and rules files (*rules and *.props)	<p>\$NCHOME/omnibus/probes/migrated</p> <p>Note: All probes must be reinstalled, and the old data migrated into the directory above must be copied to the new probe location.</p>
TSM configuration files	\$NCHOME/omnibus/tsm/migrated

Migrating your digital certificates and keys (UNIX and Linux)

Tivoli Netcool/OMNIBus V7.2.1, or later, uses the IBM Key Management (iKeyman) utility as the certificate management tool for Secure Sockets Layer (SSL) communication. If you upgraded from an earlier version that used SSL for client and server communications, and want to continue to use your old certificates in V7.3, you must migrate your certificate files and private keys into the Certificate Management System (CMS) key database that is used for certificate management in V7.3.1

Note: The old certificates were encrypted with non-FIPS 140–2 certified algorithms, so certificate migration is supported only in non-FIPS 140–2 mode. If you intend to operate in FIPS 140–2 mode, you must use iKeyman to re-create all the old certificates that you want to reuse.

The key database is a file that stores digital certificates and keys. In V7.2.1, or later, a key database must be created on each server computer where an ObjectServer, process agent, or proxy server is configured for SSL, and on each client computer that uses SSL connections. The key database also requires a password for access control; this password must be stored in a stash file (omni.sth) in the same location as the key database. On UNIX, the file name and location of the key database is \$NCHOME/etc/security/keys/omni.kdb.

After upgrading to V7.3,1 follow these guidelines to migrate your existing certificates and keys, assuming no key database currently exists:

1. Use the iKeyman GUI to create the key database and its stash file.
2. Run the certificate migration utility (**nco_ssl_migrate**).

V7.2.1 certificates are automatically migrated to V7.3.1

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

“Managing digital certificates for SSL communication” on page 380

Related reference

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102

Running the certificate migration utility (UNIX and Linux)

You can run the certificate migration utility (**nco_ssl_migrate**) on any server or client computer that has a trusted certificate database or server certificates to be migrated.

The certificate migration tool has two modes of operation, which can be used together, or separately:

- Automatic import: The utility locates the `omni.dat` file in the specified location from which you want to migrate certificates, and then locates the properties files for each of the SSL servers that are defined in the `omni.dat` file. The SSL properties are read to determine the location of the SSL certificates, which are then migrated, followed by the certificates in the trusted certificate database. For example, for a V7.2 to V7.3.1 upgrade, the following certificates are migrated into the key database:
 - Server certificates: `$NCHOME/etc/servername.crt`
 - Trusted certificate database: `$NCHOME/platform/arch/config/trusted.txt`, where *arch* represents the operating system directory
- Manual import: This mode of operation provides an additional method for importing certificates that the automatic import process cannot find; for example, process agent certificates, which are defined on the command line.

If you upgraded from V7.1 or V7.2, the upgrade process should have installed the new files into the same location as your previous installation. You can migrate your certificates by using the following methods:

- Automatic import: Run the **nco_ssl_migrate** utility with the `-auto` and `-fromnchome` options.
- Manual import: Migrate the server certificates by running the **nco_ssl_migrate** utility with the `-manual` option, and either or both of the `-servercerts` and `-trusted` options.

To import your SSL certificates and keys into the key database:

1. Create a key database (if one does not already exist). This must be a dedicated key database with the file name `omni.kdb`.
2. From the command line, run the following command to migrate the existing certificates into the key database:

```
$NCHOME/omnibus/bin/nco_ssl_migrate options
```

In this command, *options* represents the command-line options, which are described in the following table.

Table 28. Command-line options for `nco_ssl_migrate`

Command-line option	Description
<code>-auto</code>	Specifies that the existing certificates and keys must be automatically imported from a specific location into the key database. Use the <code>-auto</code> option with the <code>-fromnchome</code> or <code>-fromomnihome</code> option. Either <code>-auto</code> , or <code>-manual</code> , or both, must be specified.
<code>-dumpprops</code>	Displays all the system and <code>nco_ssl_migrate</code> properties and exits. Use this option to verify that command-line settings are being parsed correctly.
<code>-force</code>	Migrates all certificates regardless of their validity. Expired certificates and certificates that are due to become valid at a future date are migrated. Also, certificates in the key database are overwritten if they have the same name as a certificate being migrated. If <code>-force</code> is not specified, only currently valid certificates are imported.
<code>-fromnchome string</code>	Specifies the NCHOME location from which certificates are to be migrated. Use the <code>-fromnchome</code> option with the <code>-auto</code> option to migrate certificates from V7.1 or V7.2. Set the value of <code>-fromnchome</code> to your previous NCHOME location, which should be the same as your new NCHOME location (if you chose to upgrade to the same location). You can set the <code>-fromnchome</code> option to the <code>\$NCHOME</code> environment variable or its expanded value. For example: <code>-fromnchome "\$NCHOME"</code> <code>-fromnchome "/opt/netcool"</code> Either <code>-fromnchome</code> or <code>-fromomnihome</code> must be specified. If both are specified, the <code>-fromnchome</code> option overrides the <code>-fromomnihome</code> option.
<code>-fromomnihome string</code>	Specifies the OMNIHOME location from which certificates are to be migrated. Use the <code>-fromomnihome</code> option with the <code>-auto</code> option to migrate certificates from V7.0. Either <code>-fromnchome</code> or <code>-fromomnihome</code> must be specified. If both are specified, the <code>-fromnchome</code> option overrides the <code>-fromomnihome</code> option.
<code>-help</code>	Displays help information about the command-line options and exits.
<code>-manual</code>	Specifies that the existing certificates and keys must be manually imported into the key database. Either <code>-auto</code> , or <code>-manual</code> , or both, must be specified. If <code>-manual</code> is specified, the <code>-servercerts</code> option, or <code>-trusted</code> option, or both, must also be specified to identify the certificates to be imported.

Table 28. Command-line options for `nco_ssl_migrate` (continued)

Command-line option	Description
<code>-messagelevel string</code>	<p>Specifies the message logging level. Possible values are: debug, info, warn, error, and fatal. The default level is warn.</p> <p>Messages that are logged at each level are as follows:</p> <ul style="list-style-type: none"> • fatal: fatal only • error: fatal and error • warn: fatal, error, and warn • info: fatal, error, warn, and info • debug: fatal, error, warn, info, and debug <p>These values can be uppercase, lowercase, or mixed case.</p> <p>Messages are logged to <code>\$NCHOME/omnibus/log/nco_ssl_migrate.0.log</code>, with a maximum limit of 1024 KB. When the file reaches this limit, it is closed and renamed <code>nco_ssl_migrate.1.log</code>, and a new <code>nco_ssl_migrate.0.log</code> file is started. When the new file reaches the maximum size, it is renamed <code>nco_ssl_migrate.1.log</code>, overwriting any existing file, and the process continues.</p>
<code>-nowarn</code>	Indicates that you do not want to be prompted for confirmation of actions. Prompts for passwords will still be displayed if required.
<code>-password string</code>	Only use this command-line option if you want to use a different password to open the key database. The password that you specify overrides the stash file password. If you run <code>nco_ssl_migrate</code> without this command-line option, the stash file is used to open the key database so that the files can be migrated into it.
<code>-servercerts string1:string2:string3, ...</code>	<p>Only use this option in conjunction with the <code>-manual</code> option.</p> <p>Specifies a comma-separated list of server certificates to import, where:</p> <ul style="list-style-type: none"> • <i>string1</i> is the name of the server. • <i>string2</i> is the file path and name of the certificate. • <i>string3</i> is the encrypted private key password, which was encrypted with the <code>nco_g_crypt</code> utility. If you do not specify the encrypted password here, you are prompted for the password later and will have to enter it in plain text. <p>A colon (:) is required to separate the server name, certificate, and password. In the following example, an encrypted password is provided for the first certificate entry, but no password is specified for the second entry.</p> <p>"NCOMS:\$NCHOME/etc/NCOMS.crt:EHEDAIBFAPFM,NCOMSB:\$NCHOME/etc/NCOMSB.crt"</p>
<code>-trusted string,...</code>	<p>Only use this option in conjunction with the <code>-manual</code> option.</p> <p>Specifies a comma-separated list of trusted signer certificates to import. If all your trusted certificates were stored in the <code>trusted.txt</code> file, specify only this file here; for example:</p> <p>"\$NCHOME/platform/arch/config/trusted.txt"</p> <p>Where <i>arch</i> represents your operating system directory.</p>

Table 28. Command-line options for `nco_ssl_migrate` (continued)

Command-line option	Description
<code>-version</code>	Displays version information about the <code>nco_ssl_migrate</code> utility and exits.

Results

When you run `nco_ssl_migrate`, automatic import occurs first, followed by the manual import.

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

“Managing digital certificates for SSL communication” on page 380

IBM Tivoli Enterprise Console BAROC data migration (UNIX and Linux)

Tivoli Netcool/OMNIBus provides integration with Tivoli Enterprise Console.

The Tivoli Enterprise Console product is a rules-based event management application that integrates system, network, database, and application management to help ensure the optimal availability of the IT services of an organization.

In Tivoli Enterprise Console, an event is an object that is created based on data that is obtained from a source that is monitored by an event adapter. Each event is identified by a class name, which the event adapter defines. Class names are used to label events, but each event contains additional information that helps define and locate a potential problem. Event classes can be subclassed to facilitate the further breakdown of information so that more detailed rules can be applied to the information. An adapter formats event information into attributes that contain a name and value, and sends this information to the event server for further processing.

An adapter uses various files for its operations. One of these files is the Basic recorder of objects in C (BAROC) file, which describes the classes of events that the adapter supports, to the event server. The event server must load this file before it can understand events received from the adapter. A BAROC file has a `.baroc` extension.

In Tivoli Netcool/OMNIBus, the ObjectServer stores and processes events in a 'flat' normalized representation, which is not compatible with the class hierarchy and extended attribute format that is adopted for Tivoli Enterprise Console events.

To support the migration of Tivoli Enterprise Console event data, Tivoli Netcool/OMNIBus provides a BAROC tool for converting the data. The ObjectServer schema also provides the following objects to support Tivoli Enterprise Console data migration:

- A `master.class_membership` table is used to store details of all Tivoli Enterprise Console classes with the class ID, name and parent ID. The BAROC tool populates this table.
- An `ExtendedAttr` column of data type `varchar(4096)` within the `alerts.status` table, stores multiple name-value pairs in one column, in a format compatible with Tivoli Enterprise Console event strings.

- SQL and probe rule functions
 - An `instance_of` sql function : Returns true if class is a subclass of `parent_class` or they are equal, using the hierarchy defined in the `master.class_membership` table.
 - An `nvp_exists()` sql function: Verifies whether a name-value pair exists.
 - An `nvp_get()` sql function: Retrieves the value of a specific name-value pair.
 - An `nvp_set()` sql function: Adds or replaces keys from a name-value pair string and returns the new name-value pair string.
 - An `nvp_add()` probe rule function: Adds or replaces variables and their values to a name-value pair list or creates a name-value pair list of all variables.
 - An `nvp_remove()` probe rule function: Removes keys from a name-value pair string and returns the new name-value pair string.

About the BAROC conversion tool (`nco_baroc2sql`) (UNIX and Linux)

To support data migration from Tivoli Enterprise Console to Tivoli Netcool/OMNIbus, a tool is provided in Tivoli Netcool/OMNIbus for converting Tivoli Enterprise Console BAROC files to ObjectServer SQL files, which you can then import into the database.

The BAROC tool (`nco_baroc2sql`) is installed when you select the **Servers** feature during the Tivoli Netcool/OMNIbus installation. This tool is located in the `$NCHOME/omnibus/bin` directory. You must first run the tool on your `.baroc` load file to create SQL INSERT statements that are compliant with ObjectServer SQL. These statements are saved to a file that you specify. After generating the SQL output, you must import the data that is defined in the INSERT statements into the ObjectServer database.

When you run the `nco_baroc2sql` tool, it writes an INSERT statement for the ObjectServer `master.class_membership` table for each class-to-parent relationship that exists in the BAROC file. Where the BAROC class has a multiple inheritance relationship to its parent classes, the `nco_baroc2sql` tool writes an INSERT statement for each class/parent relationship that exists in the BAROC file. The format of the INSERT statement that the `nco_baroc2sql` tool generates is:

```
insert into master.class_membership ( Class, ClassName, Parent ) values (int, 'string', int );
```

Where:

- The `Class` value contains a unique numeric identifier for the class. The generated class identifiers start from 76000, unless you specify a different start value when running the tool from the command line.
- The `ClassName` value contains the name of the class as it appears in the BAROC file.
- The `Parent` value contains the numeric class value of the parent class. If no parent class is defined in the BAROC file, an INSERT statement for the class is created, which has the `Parent` field set to -1. (These entries are known as root nodes.)

For example:

```
insert into master.class_membership (Class, ClassName, Parent ) values ( 76000, 'ABC_Base', 76001);
```


The **nco_baroc2sql** tool also creates a class conversion entry for each class in the .baroc files. This enables class-specific tools to be written for the Tivoli Netcool/OMNIBus event list. The format of the INSERT statement that the tool generates is:

```
insert into alerts.conversions values ('Class+ClassID', 'Class', ClassID, 'ClassName' );
```

For example:

```
insert into alerts.conversions values ( 'Class76000', 'Class', 76000, 'ABC_Base');
```

Note: The master.class_membership table does not permit duplicate mappings of class names to class numbers. The table also does not permit multiple entries with either the same class name or class number.

Both types of INSERT statements are saved to the same output file.

Note: The **nco_baroc2sql** tool does not perform any validation to check whether the class identifiers that it allocates are available on the target system. The tool also does not put an upper limit on the class identifier values.

The Tivoli Enterprise Console classes are mapped to ObjectServer classes, and 10,000 class identifiers are reserved for this mapping, ranging from 76000 to 86000.

To map incoming Tivoli Enterprise Console events to ObjectServer class identifiers, the **nco_baroc2sql** tool can optionally generate a lookup table file that can be inserted into the rules file of a probe. The lookup table contains the Tivoli Enterprise Console ClassName values mapped to ObjectServer classes. The lookup table has the following format:

```
ClassName ClassID
```

Each class is defined on a separate line, with one definition for each row that is added to the alerts.conversions table by the SQL that the **nco_baroc2sql** tool generates.

Migrating BAROC data (UNIX and Linux)

Before migrating Tivoli Enterprise Console data, you must prepare a load file that defines the BAROC files to be processed. The BAROC files specified in the load file must be located in the same directory as the load file.

When you run the migration, the **nco_baroc2sql** tool reads the specified load file and processes the BAROC files in the order in which they are presented in the load file.

To run the migration:

1. Enter the following command from the command line:

```
$NCHOME/omnibus/bin/nco_baroc2sql -baroc baroc_load_file -sql  
output_file -lookup lookup_file
```

Where:

- *baroc_load_file* represents the file path and name of the BAROC load file
- *output_file* represents the file path and name of the output file, which is generated as an SQL file
- *lookup_file* optionally represents the file path and name of the lookup table file to which the mapping of ClassName values to ObjectServer classes is written

The **nco_baroc2sql** command has the following command-line options. Either the **-sql** command-line option or the **-lookup** option must be specified, or both. If neither command-line option is specified, the **nco_baroc2sql** tool fails.

Table 29. Command-line options for the **nco_baroc2sql** command

Command-line option	Description
-baroc <i>file</i>	The path to the BAROC load file, which lists the BAROC files to be processed.
-sql <i>file</i>	The path to which the SQL output file will be written.
-help	Displays help text.
-version	Displays version information about the tool.
-classno <i>int</i>	The base class number to use for class conversions. The default is 76000. Only change this value if you have existing conversions in the range of 76000 to 86000.
-lookup <i>file</i>	Optional: The name of the lookup table file to which the mapping of ClassName values to ObjectServer classes is written.

Wait for processing to complete.

- Log on to the SQL interactive interface and then import the SQL output file to the ObjectServer as follows:

```
$NCHOME/omnibus/bin/nco_sql -server servername -username root -password
""< output_file
```

Where *servername* represents the name of the ObjectServer to which data will be imported, and *output_file* represents the file path and name of the SQL output file.

Results

Processing messages are output to the screen and are not generated to a log file. You can redirect the messages to a log file if required.

The Probe for Tivoli EIF provides event flow integration between Tivoli Enterprise Console and Tivoli Netcool/OMNIBus. Information about this probe is available at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.netcool_OMNIBus.doc/probes/tivoli_eif/tivoli_eif/wip/concept/tveif_intro.html

What to do next

If you need to change the master.class_membership table after you ran the **nco_baroc2sql** tool, proceed as follows:

- To add new entries to the master.class_membership table, determine the highest number for class conversions that the table currently contains. Then, rerun the **nco_baroc2sql** tool and use the **-classno** option to specify a base class number for class conversions that is greater than the currently-highest number.
- To change the mapping of class name to class number, delete the existing entries in the master.class_membership table. Then, rerun the **nco_baroc2sql** tool and use the **-classno** option to specify a different base class number to use for class conversions.

You can now reimport the SQL output file into the ObjectServer by repeating step 2 on page 93 of this task.

If you used the `-lookup` command-line option, you can now insert the generated lookup table file into the rules file of the required probe. The following example shows how to define the lookup table `tec_class` in the rules file:

```
table tec_class = "lookup_table"
default = "Unknown"
```

Where `lookup_table` is the path to the lookup table that is generated by the **nco_baroc2sql** tool. The following example shows how to use the `lookup` function to populate the `Class` element with the Tivoli Enterprise Console class name:

```
$Class = lookup($ClassName,tec_class)
```

Performing postinstallation tasks (UNIX and Linux)

After installing or upgrading Tivoli Netcool/OMNIBus, you must perform a number of postinstallation tasks and then configure your system.

Perform one or more of these tasks, depending on the features that you installed:

- Set a number of environment variables (if required).
- Configure the IEHS server for online help access.
- If you want to operate in FIPS 140–2 mode, configure your JRE for FIPS 140–2. Also configure FIPS 140–2 support for the server components.
- Install probes and gateways.
- If you installed the Administrator feature, you can select a browser for the Netcool/OMNIBus Administrator online help.
- If you installed the Gateways, Servers, or Process Control feature, configure server communications.
- If you installed the Servers feature, create an ObjectServer.
- If you are creating a distributed installation, see the additional instructions in *Distributed Installations*.

What to do next

To obtain additional support with Tivoli Netcool/OMNIBus and to aid with problem determination, you can also install the IBM Support Assistant.

To use IBM Tivoli Monitoring to monitor and manage Tivoli Netcool/OMNIBus resources, install the IBM Tivoli Monitoring agent for Tivoli Netcool/OMNIBus. For further information about this agent, see the *IBM Tivoli Monitoring for Tivoli Netcool/OMNIBus Agent User's Guide*.

Related concepts

“Configuring server communication details in the Server Editor” on page 240
Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related tasks

“Creating an ObjectServer” on page 231
“Setting up distributed installations” on page 248

Related reference

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102
Appendix A, “IBM Support Assistant,” on page 605

Setting Tivoli Netcool/OMNIBus environment variables (UNIX and Linux)

When the installation or upgrade is complete, you might need to set a number of environment variables.

The following table describes the environment variables that you might need to set. Note that the POSIX syntax format is deprecated.

Table 30. Netcool/OMNIBus environment variables

Environment variable	Description
NCHOME	<p>This is the home location for Tivoli Netcool/OMNIBus.</p> <p>It is not mandatory that you manually set this environment variable before running the Tivoli Netcool/OMNIBus installation program. Provision is made for this environment variable to be automatically set when you run a Tivoli Netcool/OMNIBus program using one of the nco_* wrapper scripts that are typically located in \$NCHOME/bin or \$NCHOME/omnibus/bin.</p> <p>Note: If you are using more than one Netcool installation, do not set this variable because it can cause programs in one installation to use files in the other installation. However, it is useful to set this variable when using some utilities that are not part of an installation (for example, the installer itself, or patch bundles).</p>
OMNIHOME	<p>This environment variable was previously used to specify the location of a Tivoli Netcool/OMNIBus installation, and is now used to provide legacy support for scripts, third-party applications, and probes that continue to use the \$OMNIHOME environment variable.</p> <p>When using such applications with Tivoli Netcool/OMNIBus V7.3.1. wrapper scripts automatically change the value of \$OMNIHOME to \$NCHOME/omnibus.</p>

Table 30. Netcool/OMNIBus environment variables (continued)

Environment variable	Description
PATH	<p>This environment variable specifies the path to executable files.</p> <p>If you want to run Tivoli Netcool/OMNIBus programs without entering their full path each time, you can add the directory locations of these programs to your PATH environment variable. Path values that you can add include: \$NCHOME/omnibus/bin and \$NCHOME/omnibus/probes.</p>
<p>LD_LIBRARY_PATH on all supported versions of Solaris and Linux</p> <p>LIBPATH on all supported versions of AIX</p> <p>SHLIB_PATH on all supported versions of HP-UX</p>	<p>This environment variable is operating system-specific, and specifies the path to shared libraries that are used to provide a smaller total distribution size for Tivoli Netcool/OMNIBus.</p> <p>The run-time dynamic loader module automatically looks for shared libraries in the default directory \$NCHOME/platform/arch/lib, where <i>arch</i> is the directory that corresponds to your operating system. For example, the default installation location of the shared libraries for Solaris is /opt/IBM/tivoli/netcool/platform/solaris2/lib.</p> <p>On Solaris, AIX, and HP-UX, you do not typically need to check or modify this environment variable setting. However, if another user or application has modified the environment variable, Tivoli Netcool/OMNIBus can function incorrectly or fail. In this situation, check the shared library paths.</p> <p>On Linux, the Netcool/OMNIBus installation provides dynamic (rather than static) links to the Open Motif 2.2.3 library. If this library is installed within its default system directory (for example, /usr/X11R6/lib/libXm.so.3), the LD_LIBRARY_PATH environment variable is automatically set to this path. However, if the library is installed in a different location, you must manually update the environment variable with this location. As with other UNIX operating systems, you can check the shared library paths.</p>
NDE_LOGFILE_MAXSIZE	<p>This environment variable controls the maximum size of log files, in bytes, for the ObjectServer, the ObjectServer Gateway, proxy servers, the nco_postmsg utility, and the nco_bridgeserve utility.</p> <p>When the log file reaches the maximum size specified, the application (for example the ObjectServer) rename log file, for example <i>servername.log</i>, to <i>servername.log_old</i> and starts a new <i>servername.log</i> file. When the new <i>servername.log</i> file reaches the maximum size, <i>servername.log_old</i> is overwritten, and so on.</p> <p>As an alternative to the NDE_LOGFILE_MAXSIZE environment variable, you can use the NDE_LOGFILE_ROTATION_FORMAT and NDE_LOGFILE_ROTATION_TIME environment variables to enforce log file rotation.</p>

Table 30. Netcool/OMNibus environment variables (continued)

Environment variable	Description
NDE_LOGFILE_ROTATION_FORMAT	<p>This environment variable specifies whether a log file rotation takes place for the ObjectServer, the ObjectServer Gateway, proxy servers, the nco_postmsg utility, and the nco_bridgeserve utility. This environment variable also specifies a timestamp that is appended to the old log file after rotation has taken place. Use this environment variable in conjunction with the NDE_LOGFILE_ROTATION_TIME environment variable.</p> <p>If you set the NDE_LOGFILE_ROTATION_FORMAT variable to a non-null value, a daily log rotation is enforced. If you specify a value in POSIX format syntax, or Local Data Markup Language (LDML) syntax format, a timestamp is appended to the old log file after rotation. The timestamp ensures that each old log file has a unique name and so is not overwritten. A timestamp generates a log file name as per the following example: <i>objectservername.log_201004301356</i>. For more information about LDML format syntax, see http://userguide.icu-project.org/formatparse/datetime.</p> <p>If you do not require a timestamp, you can set the value of the NDE_LOGFILE_ROTATION_FORMAT variable to literal characters, for example "rotation." After rotation, this value is appended to the old log file, and generates a log file name as per the following example: <i>objectservername.log_rotated</i>. If you set the variable to literal values then the old rotated files are overwritten by newly rotated files, at the time specified by the NDE_LOGFILE_ROTATION_TIME environment variable.</p> <p>Important: Literal characters must be escaped by single quotes (') as described in http://userguide.icu-project.org/formatparse/datetime.</p> <p>If you set the NDE_LOGFILE_ROTATION_FORMAT environment variable, the NDE_LOGFILE_MAXSIZE environment is ignored.</p>
NDE_LOGFILE_ROTATION_TIME	<p>This environment specifies the time at which a log file rotation occurs, for the ObjectServer, the ObjectServer Gateway, proxy servers, the nco_postmsg utility, and the nco_bridgeserve utility, if you have set the NDE_LOGFILE_ROTATION_FORMAT environment variable to enforce a log file rotation.</p> <p>The NDE_LOGFILE_ROTATION_TIME variable is set to indicate at what time of day the rotation occurs in hours and minutes (specified as hhmm, where hh is in the 24-hour time format).</p>

The following examples show how to manually set the NCHOME, OMNIHOME, PATH, NDE_LOGFILE_MAXSIZE, and NDE_LOGFILE_ROTATION_FORMAT and NDE_LOGFILE_ROTATION_TIME environment variables. These examples assume that the Netcool home directory /opt/IBM/tivoli/netcool is used for Solaris, HP-UX, and Red Hat Linux, and /usr/IBM/tivoli/netcool is used for AIX.

Example: Setting NCHOME, OMNIHOME, and PATH on Solaris, HP-UX, and Red Hat Linux

Each csh user can add the following lines to their \$HOME/.login file:

```
setenv NCHOME /opt/IBM/tivoli/netcool
setenv OMNIHOME $NCHOME/omnibus
setenv PATH $NCHOME/omnibus/bin:$PATH
```

Each ksh and sh user can add the following lines to their \$HOME/.profile file:

```
NCHOME=/opt/IBM/tivoli/netcool;export NCHOME
OMNIHOME=$NCHOME/omnibus;export OMNIHOME
PATH=$PATH:$NCHOME/omnibus/bin;export PATH
```

Example: Setting NCHOME, OMNIHOME, and PATH on AIX

Each csh user can add the following lines to their \$HOME/.login file:

```
setenv NCHOME /usr/IBM/tivoli/netcool
setenv OMNIHOME $NCHOME/omnibus
setenv PATH $NCHOME/omnibus/bin:$PATH
```

Each ksh and sh user can add the following lines to their \$HOME/.profile file:

```
NCHOME=/usr/IBM/tivoli/netcool;export NCHOME
OMNIHOME=$NCHOME/omnibus;export OMNIHOME
PATH=$PATH:$NCHOME/omnibus/bin;export PATH
```

Example: Setting NDE_LOGFILE_MAXSIZE

The following example shows how to set the maximum file size for the ObjectServer to 102,400 bytes:

```
setenv NDE_LOGFILE_MAXSIZE 102400
nco_objserv
```

Example: Setting NDE_LOGFILE_ROTATION_FORMAT and NDE_LOGFILE_ROTATION_TIME using POSIX

The following example shows how to rotate the log files at midnight each day, and append the old log file name with the year, month, day of month, hour and minute by using POSIX format syntax:

```
setenv NDE_LOGFILE_ROTATION_FORMAT %Y%m%d-%H%M
setenv NDE_LOGFILE_ROTATION_TIME 0000
```

Example: Setting NDE_LOGFILE_ROTATION_FORMAT and NDE_LOGFILE_ROTATION_TIME using LDML

The following example shows how to rotate the log files at midnight each day, and append the old log file name with the year, month, day of month, hour and minute by using LDML format syntax:

```
setenv NDE_LOGFILE_ROTATION_FORMAT yyyyMMdd-HHmm
setenv NDE_LOGFILE_ROTATION_TIME 0000
```

Example: Setting NDE_LOGFILE_ROTATION_FORMAT and NDE_LOGFILE_ROTATION_TIME with a literal string

The following example shows how to rotate the log files at midnight each day, and append the old log file name with the literal character string “old”:

```
setenv NDE_LOGFILE_ROTATION_FORMAT \'old\'
setenv NDE_LOGFILE_ROTATION_TIME 0000
```

Related tasks

“Checking the shared library paths”

“Notes for SSL connections in FIPS 140–2 mode” on page 380

Checking the shared library paths

The Tivoli Netcool/OMNIBus directory structure uses shared libraries to provide a smaller total distribution size. The operating system-specific environment variable (LD_LIBRARY_PATH on Solaris and Linux, LIBPATH on AIX, or SHLIB_PATH on HP-UX) is used to specify the location of these libraries.

On Solaris, AIX, and HP-UX, you do not typically need to check or modify this environment variable setting. However, if another user or application has modified the environment variable, Tivoli Netcool/OMNIBus can function incorrectly or fail. In this situation, you must check that all the shared libraries can be found, using the following commands:

- **ldd** on Solaris and Linux systems
- **chatr** on HP-UX systems
- **dump -H** on AIX systems

These commands list the dynamic dependencies of executable files.

The following table shows examples of how to use these commands to list all the dependencies for all installed binaries that are located in the \$NCHOME/omnibus/platform/*arch*/bin/ directory, where *arch* represents the operating system directory.

Table 31. Checking shared library paths

Operating system	Command	Output description
Solaris and Linux	<code>ldd \$NCHOME/omnibus/platform/<i>arch</i>/bin/nco_*</code>	The output of this command lists the dynamic dependencies and indicates which libraries cannot be found.
HP-UX	<code>chatr \$NCHOME/omnibus/platform/<i>arch</i>/bin/nco_*</code>	The output of this command shows the shared library path and whether the environment variable is enabled or disabled. To enable the environment variable if it is disabled, enter the command: <code>chatr +s enable \$NCHOME/omnibus/platform/<i>arch</i>/bin/nco_*</code>
AIX	<code>dump -H \$NCHOME/omnibus/platform/<i>arch</i>/bin/nco_*</code>	The output of this command lists the dynamic dependencies and indicates which libraries cannot be found.

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

Related reference

“Setting Tivoli Netcool/OMNIBus environment variables (UNIX and Linux)” on page 95

Configuring settings for online help access (UNIX and Linux)

After installing Tivoli Netcool/OMNIBus, you might need to configure your system for online help access. Online help is deployed using IBM Eclipse Help System (IEHS) and can be accessed in standalone mode or information center mode. On UNIX, it might also be necessary for you to configure environment variable settings for your browser.

Configure the online help settings in the following table, as relevant for your system.

Table 32. Help configuration settings

Setting	Description
Environment variables for browser on UNIX	<p>Add the directory location of your browser to the following environment variables (if not already added):</p> <ul style="list-style-type: none">• PATH• LD_LIBRARY_PATH (Solaris and Linux), LIBPATH (AIX), or SHLIB_PATH (HP-UX)• Browser-specific environment variables; for example, MOZILLA_FIVE_HOME <p>The operating system default browser is used.</p>
Standalone mode Local Help System feature installed on a client workstation (Online help files installed on a local IEHS Web server)	<p>Configuration is only necessary if the default IEHS port number of 8888 is being used by another local service.</p> <p>If this is the case, edit the IEHS configuration file <code>\$NCHOME/omnibus/etc/nco_IEHS.cfg</code> as follows:</p> <ul style="list-style-type: none">• IEHSMODE: 0• IEHSHost: <i>leave blank</i>• IEHSPort: <i>an unused port number</i> <p>Note: After changing the port number in the configuration file, you must shut down the local IEHS server for your changes to take effect. Run the <code>\$NCHOME/omnibus/bin/help_end</code> command to shut down the server. The server automatically restarts when you access online help. (If you change your environment variable settings for the browser, you must also shut down the IEHS server for your changes to take effect.)</p>

Table 32. Help configuration settings (continued)

Setting	Description
Information center mode	On the computer designated as the IEHS server, edit the IEHS configuration file <code>\$NCHOME/omnibus/etc/nco_IEHS.cfg</code> as follows:
Local Help System feature installed on a remote server	<ul style="list-style-type: none"> • IEHSMode: 1 • IEHSHost: <i>leave blank, or specify the IP address or host name of the IEHS server</i> <p>Note: IEHS V3.1.1 does not support IPv6 addresses.</p> <ul style="list-style-type: none"> • IEHSPort: <i>an unused port number for the IEHS server</i> <p>The default port number is 8888. If necessary, update your firewall settings to open the port.</p>
(Online help files installed on a remote IEHS Web server, which is configured for client access; typically managed by a system administrator)	<p>The host name on which the IEHS server is running can be obtained from the <code>\$NCHOME/omnibus/platform/arch/nco_IEHS/eclipse/workspace/.metadata/.connection</code> file. Note that this file is available only when the IEHS server is running. The file is deleted when you shut down the IEHS server.</p> <p>On each client workstation, edit the IEHS configuration file <code>\$NCHOME/omnibus/etc/nco_IEHS.cfg</code> as follows:</p> <ul style="list-style-type: none"> • IEHSMode: 1 • IEHSHost: <i>IP address or host name of the IEHS server</i> • IEHSPort: <i>port number on which the IEHS server is running</i> <p>Important: You must instruct users to perform this task.</p>

Related concepts

“Online help requirements” on page 16

Running the IEHS server (UNIX and Linux)

In information center mode, you must manually start the IEHS server by using the **IC_start** command. In standalone mode, the local IEHS server starts automatically.

If you have configured your help system to use the information center mode, you must start the IEHS server to make it available to users who need to access online help. To start the IEHS server on the configured computer, enter the following command at the command line:

```
$NCHOME/omnibus/bin/IC_start
```

To stop the IEHS server, enter the following command at the command line:

```
$NCHOME/omnibus/bin/IC_end
```

In standalone mode, the local IEHS server automatically starts the first time that you make a help request. The local IEHS server continues to run until you stop it by using the following command:

```
$NCHOME/omnibus/bin/help_end
```

Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)

To configure the Tivoli Netcool/OMNIBus JRE for FIPS 140–2 operation, change the configuration of the security properties file. You can also download and add policy files to use enhanced encryption algorithms.

Configuration file changes

Make the following configuration changes to the security properties file:

1. Open the `$NCHOME/platform/arch/jre_1.6.7/jre/lib/security/java.security` file for editing, where *arch* represents your operating system directory; for example, `solaris2`.
2. Edit the file as follows:
 - In the List of providers and their preference orders section, add the following lines:
`security.provider.1=com.ibm.fips.jsse.IBMJSSEFIPSPProvider` and
`security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS`. For all other providers, increment the number by two, as shown in the following table, for your operating system:

Operating system	Required entries
AIX and Linux	<code>security.provider.1=com.ibm.fips.jsse.IBMJSSEFIPSPProvider</code> <code>security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS</code> <code>security.provider.3=com.ibm.jsse2.IBMJSSEProvider2</code> <code>security.provider.4=com.ibm.crypto.provider.IBMJCE</code> <code>security.provider.5=com.ibm.security.jgss.IBMJGSSProvider</code> <code>security.provider.6=com.ibm.security.cert.IBMCertPath</code> <code>security.provider.7=com.ibm.security.sasl.IBMSASL</code> <code>security.provider.8=com.ibm.xml.crypto.IBMXMLCryptoProvider</code> <code>security.provider.9=com.ibm.xml.enc.IBMXMLEncProvider</code> <code>security.provider.10=org.apache.harmony.security.provider.PolicyProvider</code> <code>security.provider.11=com.ibm.security.jgss.mech.spnego.IBMSPNEGO</code> <code>security.provider.12=com.ibm.security.cmskeystore.CMSProvider</code>
Solaris and HP-UX	<code>security.provider.1=com.ibm.fips.jsse.IBMJSSEFIPSPProvider</code> <code>security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS</code> <code>security.provider.3=com.ibm.security.jgss.IBMJGSSProvider</code> <code>security.provider.4=sun.security.provider.Sun</code> <code>security.provider.5=com.ibm.crypto.provider.IBMJCE</code> <code>security.provider.6=com.ibm.jsse2.IBMJSSEProvider2</code> <code>security.provider.7=com.ibm.security.cert.IBMCertPath</code> <code>security.provider.8=com.ibm.security.sasl.IBMSASL</code> <code>security.provider.9=com.ibm.xml.crypto.IBMXMLCryptoProvider</code> <code>security.provider.10=com.ibm.xml.enc.IBMXMLEncProvider</code> <code>security.provider.11=com.ibm.security.jgss.mech.spnego.IBMSPNEGO</code> <code>security.provider.12=com.ibm.security.cmskeystore.CMSProvider</code>

- Set the default key and trust manager factory algorithms for the `javax.net.ssl` package:
`ssl.KeyManagerFactory.algorithm=IbmX509`
`ssl.TrustManagerFactory.algorithm=IbmX509`
 - Set the default `SSLConnectionFactory` and `SSLServerSocketFactory` provider implementations for the `javax.net.ssl` package:
`ssl.SocketFactory.provider=com.ibm.jsse2.SSLConnectionFactoryImpl`
`ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl`
3. Save and close the file.

Enhanced encryption algorithms

To enable strong encryption, you need to download and install policy files that allow this feature, from IBM developerWorks®. This involves acceptance of licensing terms.

The steps to enable strong encryption are as follows:

1. Go to the developerWorks Java Technology Security Web page at <http://www-106.ibm.com/developerworks/java/jdk/security/>.
2. Click the **Java SE 6** link. (The files are the same for JRE 1.5.n.)
3. Scroll down on the resulting page and click the **IBM SDK Policy files** link.
4. If you already have an IBM ID and password, click the **Sign in** link. Otherwise, click the **Register here** link to create an ID.
5. On the "Sign in" page, supply your IBM ID and password. This takes you to the "Unrestricted JCE policy files for SDK 1.4" page.
6. Select **Unrestricted JCE Policy files for SDK for all newer versions** and click **Continue**.
7. Scroll down to the License section of the resulting page and click the **View license** link to see the licensing terms for the download.
8. If the licensing terms are acceptable, select **I agree** and click the **I confirm** link. If the terms are not acceptable, you will not be able to enable strong encryption and should click **I cancel**.
9. Click the **Download now** link to download the unrestricted.zip file.
10. Extract the local_policy.jar and US_export_policy.jar files from the unrestricted.zip archive.
11. Save these two files to the \$NCHOME/platform/arch/jre_1.6.7/jre/lib/security directory, replacing the existing files of the same names.
12. Update the policy files on each computer, and optionally run tests.

Related reference

"Switching your installation to FIPS 140-2 mode" on page 299

Installing probes and gateways into the Tivoli Netcool/OMNIBus environment (UNIX and Linux)

Probes and gateways are part of the Tivoli Netcool/OMNIBus suite, and are available as download packages on the Passport Advantage Online Web site.

You can install probes and gateways into a new Tivoli Netcool/OMNIBus environment, or upgrade probes and gateways after upgrading Tivoli Netcool/OMNIBus.

Probes are generally installed on a separate workstation from the ObjectServer.

Tip: Probes can be deployed to remote computers by using the remote deployment mechanism provided by IBM Tivoli Monitoring.

Gateways are generally installed on either the primary server or other servers. You can install ObjectServer gateways as part of the Tivoli Netcool/OMNIBus installation. Other gateways are installed separately by using the download package for individual gateways.

Related concepts

“Deploying probes remotely” on page 451

Installing probes and gateways into a new Tivoli Netcool/OMNIbus environment (UNIX and Linux)

For a new installation of Tivoli Netcool/OMNIbus V7.3.1, download and install each probe and gateway that you require. You can run the installer as a wizard, or in console or silent mode, in a similar manner used for installing Tivoli Netcool/OMNIbus.

Attention: Download and use only repackaged or new probes and gateways in your V7.3.1 installation. In V7.3.1, probes and gateways are installed using the **nco_install_integration** command. Earlier probe and gateway packages, which have not been repackaged, cannot be installed into your V7.3.1 installation by using the **nco_install_integration** command.

Before you begin

The computer on which you install the probe must have the Tivoli Netcool/OMNIbus **Probe Support** feature installed prior to probe installation. Any feature can be installed to obtain the infrastructure required for gateways.

Proceed as follows:

1. To download probes from the Passport Advantage Online Web site, follow the instructions that are available in the Tivoli Netcool/OMNIbus Information Center at:
http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.tivoli.nam.doc/welcome_ptsm.htm
2. To download gateways from the Passport Advantage Online Web site, follow the instructions that are available in the Tivoli Netcool/OMNIbus Information Center at:
http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.tivoli.nam.doc/welcome_og.htm
3. After downloading the UNIX or Linux installation package for a probe or gateway, extract the contents of the package to a temporary location.
4. Install the probe or gateway by running the following command:
`$NCHOME/omnibus/install/nco_install_integration option`

The value of *option* depends on your installation mode as follows:

Installation mode	Instruction
Installation wizard	<p>No value is required for <i>option</i>.</p> <p>When the wizard runs, follow the prompts to:</p> <ol style="list-style-type: none">1. Specify the location of the probe or gateway to be installed. This location is the directory containing the README.txt file in the extracted package.2. Accept the license conditions.

Installation mode	Instruction
Console mode	<p>Specify <i>option</i> as:</p> <pre>-i console</pre> <p>When the text-based installer runs, follow the prompts to:</p> <ol style="list-style-type: none"> 1. Specify the location of the probe or gateway to be installed. This location is the directory containing the README.txt file in the extracted package. 2. Accept the license conditions.
Silent mode	<p>Specify <i>option</i> as:</p> <pre>-i silent -f full_path/response.txt</pre> <p>Where:</p> <ul style="list-style-type: none"> • <i>full_path</i> specifies the full path to a response file named response.txt that you are required to create. • response.txt is a text file that you create with the following contents: <pre>LICENSE_ACCEPTED=true PROBE_OR_GATE_LOCATION=README_directorypath</pre> <p><i>README_directorypath</i> is the path to the directory that contains the README.txt file.</p>

Results

On completion:

- Probes are installed to the following directory:
\$NCHOME/omnibus/probes
- Gateways are installed to the following directories:
\$NCHOME/omnibus/bin: Gateway binaries
\$NCHOME/omnibus/gates: Gateway configuration files

Installing probes or gateways into an upgraded Tivoli Netcool/OMNIBus environment (UNIX and Linux)

If you have upgraded your version of Tivoli Netcool/OMNIBus V7.3.1 from a version that is earlier than V7.3, reinstall all your probes and gateways and then import the old probe and gateway configuration data. You can run the installer as a wizard, or in console or silent mode, in a similar manner used for installing Tivoli Netcool/OMNIBus. If you have upgraded from Tivoli Netcool/OMNIBus V7.3, you do not have to perform this task.

To install probes or gateways and import existing configuration data:

1. Follow the instructions for installing probes and gateways into a new Tivoli Netcool/OMNIBus environment.
2. After installing, import the configuration data.

The UPGRADE.SH script, which was used to migrate your old Tivoli Netcool/OMNIBus data into the new V7.3.1 installation, would have copied your old probe and gateway configuration files into the following locations:

- Probe configuration files: \$NCHOME/omnibus/probes/migrated
- Gateway configuration files: \$NCHOME/omnibus/etc

3. Copy the migrated probe configuration files in the \$NCHOME/omnibus/probes/ migrated directory into the appropriate locations in \$NCHOME/omnibus/probes.

Related tasks

“Installing probes and gateways into a new Tivoli Netcool/OMNIbus environment (UNIX and Linux)” on page 104

Uninstalling Tivoli Netcool/OMNIbus (UNIX and Linux)

You can uninstall Tivoli Netcool/OMNIbus by using the installation wizard, or the console or silent installation mode.

The installer records the mode that was used for the installation. When the **uninstall** command is invoked, the mode used for installing is, by default, used for uninstalling. The **uninstall** command provides command-line options that you can use to set the uninstallation mode irrespective of the mode used at installation time.

Attention: Do not attempt to uninstall Tivoli Netcool/OMNIbus by deleting files or directories unless you intend to delete the entire installation of related Tivoli products that are installed in the \$NCHOME location. You will also need to delete the Deployment Engine files. This will affect any other IBM Tivoli products that have been installed using the Deployment Engine.

When you uninstall Tivoli Netcool/OMNIbus, the uninstallation process removes all files except for the following files:

- Files used by the installer program, such as the installer plan and log files, and the installer database files
- Common packages that are required by other products installed in the same NCHOME location
- Tivoli Netcool/OMNIbus configuration files that have been modified
- Probes and non-ObjectServer gateways - these have their own uninstaller

Note: Deployment Engine files are retained only if they are still required by another product on the same server.

To completely remove Tivoli Netcool/OMNIbus and any other related Tivoli products from the \$NCHOME location:

1. Uninstall Tivoli Netcool/OMNIbus and the other products.
2. Run the following command to remove the configuration files and common files that were retained in the Netcool home location:

```
rm -rf $NCHOME
```
3. Also manually delete the following InstallAnywhere files and Deployment Engine directories (if present):

User type	Directory location
If you installed Tivoli Netcool/OMNIBus as a root user	<p>/IA-Netcool-OMNIBus-hostname- yyyymmddThhmmss-0.log (and any other IA-*.log files)</p> <p>/usr/ibm/common/acsi</p> <p>/var/ibm/common/acsi</p> <p>Where <i>yyyymmddThhmmss</i> is the date and time the log file was first generated.</p>
If you installed Tivoli Netcool/OMNIBus as a non-root user	<p>/home/username/IA-Netcool-OMNIBus- hostname-yyyymmddThhmmss-0.log (and any other IA-*.log files)</p> <p>/home/username/.acsi_hostname</p> <p>Where <i>username</i> is the name of the logged-in user who performed the installation, and <i>yyyymmddThhmmss</i> is the date and time the log file was first generated.</p>

Uninstalling using the wizard (UNIX and Linux)

You can use the uninstall wizard to guide you through the uninstallation process for Tivoli Netcool/OMNIBus.

To uninstall Tivoli Netcool/OMNIBus by using the wizard:

1. Stop all processes that are currently running.
2. From a command prompt, enter the following command:
`$NCHOME/_uninst/OMNIBus/uninstall -i gui`
The Uninstall Wizard starts and displays the Uninstall OMNIBus page.
3. Click **Uninstall** to proceed with the uninstallation, and wait while the features are removed. On completion, the wizard confirms that Tivoli Netcool/OMNIBus was successfully uninstalled.
4. Click **Done** to close the wizard.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling in console mode (UNIX and Linux)

Use the console mode to uninstall Tivoli Netcool/OMNIbus from the command-line interface.

To uninstall Tivoli Netcool/OMNIbus in console mode:

1. Stop all processes that are currently running.
2. From a command prompt, enter the following command:
`$NCHOME/_uninst/OMNIbus/uninstall -i console`
3. When prompted, press Enter to proceed. On completion, you are returned to the command prompt.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling in silent mode (UNIX and Linux)

Use the silent mode to uninstall Tivoli Netcool/OMNIbus with no user interaction.

To uninstall Tivoli Netcool/OMNIbus in silent mode:

1. Stop all processes that are currently running.
2. From a command prompt, enter the following command:
`$NCHOME/_uninst/OMNIbus/uninstall -i silent`
On completion, you are returned to the command prompt.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling probes and gateways (UNIX and Linux)

Probes and non-ObjectServer gateways are separate packages and are not removed from your system when you remove Tivoli Netcool/OMNIbus. You must uninstall probes and gateways individually.

To uninstall a probe or gateway after uninstalling Tivoli Netcool/OMNIbus:

1. From a command prompt, change to the following directory:
`$NCHOME/_uninst/name`
Where *name* is a subdirectory named after the probe or gateway.
2. Enter the following command:
`./uninstall`

The uninstaller runs in the mode in which the probe or gateway was installed.

Chapter 5. Installing, upgrading, and uninstalling (Windows)

Use this information to install, upgrade, and uninstall Tivoli Netcool/OMNIBus on Windows operating systems.

Note about the Web GUI installation: The Web GUI component is distributed as a separate installation package from the non-Web-based Tivoli Netcool/OMNIBus components. The set of instructions provided here relate only to the non-Web components. For installation, upgrade, and uninstallation instructions about the Web GUI component, see Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173.

Tip: If you intend to install both the Web GUI and non-Web components by using the installation wizard, you can use the launchpad as a starting point for installing the components. If installing in console or silent mode, you must run the installation programs separately. The Web GUI installation requires a running ObjectServer, so ensure that you have installed and started your ObjectServer before installing the Web GUI.

Related concepts

“About the launchpad” on page 33

Installable Tivoli Netcool/OMNIBus features (Windows)

You can choose which Tivoli Netcool/OMNIBus features to install on a given Windows host.

The following table describes the list of Tivoli Netcool/OMNIBus features that you can install. In the Feature column, the first term (for example, Admin) shows the feature name when installing using the wizard or console mode, and the second term (for example, nco_admin_feature) shows the feature name when installing in silent mode.

Table 33. Feature selection

Feature	Description
Desktop or nco_desktop_feature	Desktop GUI Applications Use the Event List to view and manager alerts in your system. This feature includes On-line Help and the Accelerated Event Notification (AEN) client. Use the Netcool/OMNIBus Administrator to configure ObjectServers and manage services and processes under process control.

Table 33. Feature selection (continued)

Feature	Description
Servers or nco_server_feature	<p>Server Applications</p> <p>The ObjectServer is the in-memory database server at the core of Tivoli Netcool/OMNIBus. Use the ObjectServer to store and process alert information. If you do not install the ObjectServer component, you must have an ObjectServer running elsewhere on your network.</p> <p>A proxy server reduces the number of direct connections to the primary ObjectServer. A proxy server can enhance performance when a large number of probes are forwarding alert information directly to the ObjectServer, and a large number of desktop connections are also made to the same ObjectServer.</p> <p>Use the ObjectServer gateways to connect ObjectServers.</p> <p>Use the process control system to configure and manage processes remotely. Process control simplifies the management of Tivoli Netcool/OMNIBus components such as ObjectServers, probes, and gateways. Additionally use the process agent to start processes that are used by external automations from the ObjectServer.</p> <p>Additional tools are also installed. Use the BAROC tool (nco_baroc2sql) to migrate IBM Tivoli Enterprise Console BAROC data into the ObjectServer. Use the Confpack utility (nco_confpack) to import and export parts of ObjectServer configurations. Use the ObjectServer report tool (nco_osreport) to extract entire ObjectServer configuration into SQL files for use in creating new ObjectServers with nco_dbinit.</p>
Probe Support or nco_probe_support_feature	<p>This feature is required for probe installation, and adds the underlying infrastructure for probes.</p> <p>The Probe Rules Syntax Checker (nco_p_syntax) and the nco_postmsg utility are also installed. Use the Probe Rules Syntax Checker to test the syntax of a rules file. Use the nco_postmsg utility to specify name-value pairs for alert data that can be directly sent as a single event to a specified ObjectServer.</p>

Related concepts

“Tivoli Netcool/OMNIBus components” on page 1

“Online help requirements” on page 16

“Installation modes” on page 33

Preparing to install or upgrade (Windows)

Before you install or upgrade Tivoli Netcool/OMNIBus, take note of the user and system permissions that are required for the user performing the installation or upgrade, and ensure that these permissions are granted. You also need to obtain the installation package for your operating system.

Installation or upgrade prerequisites (Windows)

If you are installing or upgrading Tivoli Netcool/OMNIBus, you must take note of a number of prerequisites.

These prerequisites are as follows:

- You must have Administrator privileges on the system.
- Sufficient disk space must be available on the volume where you want to install Tivoli Netcool/OMNIBus. If you intend to install other Network Management products, the installation location must also have sufficient space to accommodate these installations.
- Some Tivoli Netcool/OMNIBus components require the Java Runtime Environment (JRE) to be installed on your system.
- You must have write access permissions to the Netcool home directory (NCHOME) where Tivoli Netcool/OMNIBus is installed.

Note: The installation directory path must not include any special or multibyte characters.

Related concepts

"Disk space requirements" on page 17

"JRE requirements" on page 14

"The Deployment Engine" on page 34

Obtaining the installation package (Windows)

Tivoli Netcool/OMNIBus is distributed as a compressed file that is available on CD, or that you can download from the IBM Passport Advantage Online Web site.

The boxed product package contains the CD that you can use to install Tivoli Netcool/OMNIBus on your operating system.

If you are downloading the product, proceed as follows:

1. Follow the instructions in the download document available at <http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027001>.
2. Extract the contents of the installation package into a temporary location.

What to do next

After you complete these tasks, you can run the installation program to perform a new installation of Tivoli Netcool/OMNIBus or to upgrade your existing version.

Related concepts

"Installing on Windows" on page 114

"Upgrading on Windows" on page 127

Notes for Windows Vista and Windows 2008 users

When installing or upgrading, you can choose the location to which you want to install Tivoli Netcool/OMNIbus. If you are using Windows Vista or Windows 2008, take note that the C:\Program Files directory is protected, and applications cannot write to it unless explicitly run as Administrator.

If you install Tivoli Netcool/OMNIbus in the C:\Program Files directory, certain configuration files will be redirected to another Program Files directory in your local profile. By default, this location is:

C:\Users\Username\AppData\Local\VirtualStore\Program Files

Where *Username* is the name of the logged-in user.

Configuration files that are continually being read from and written to are saved to this location. On Windows Vista, which supports only the desktop component of Tivoli Netcool/OMNIbus, this includes properties files and log files. On Windows 2008, which supports both the desktop and server components, this includes configuration files such as the ObjectServer database files, properties files, and log files.

If you choose to install to the C:\Program Files directory, be aware that your files are held in two separate Program Files locations, and familiarize yourself with the contents of the directories. The settings in your properties files remain unchanged; as a consequence, although your log and properties file settings (such as **MessageLog** and **PropsFile**) reference locations in the C:\Program Files directory, these files are physically located in C:\Users\Username\AppData\Local\VirtualStore\Program Files.

Alternatively, you can install Tivoli Netcool/OMNIbus in a location outside the C:\Program Files directory. For example, if you accept the default installation path (C:\IBM\tivoli\netcool), the files are installed in a single location, as is the standard with previous versions of Windows.

Installing on Windows

On Windows systems, you can install Tivoli Netcool/OMNIbus by using the installation wizard, or the console or silent installation mode.

The documented instructions apply for a new installation of Tivoli Netcool/OMNIbus as the first product, or as a subsequent, related Tivoli product that is installed in the Netcool home location.

The installation process results in a package installation of the Tivoli Netcool/OMNIbus components. A package is a collection of related configuration files.

After you complete the installation process, you must configure Tivoli Netcool/OMNIbus before attempting to use the system.

Related concepts

“Overview of the Netcool home location” on page 34

Related tasks

“Obtaining the installation package (Windows)” on page 113

“Installing using the installation wizard (Windows)”

“Installing in silent mode (Windows)” on page 119

Chapter 2, “Backing up and restoring Tivoli Netcool/OMNIbus,” on page 9

Installing using the installation wizard (Windows)

Run the wizard to present the installation options in a graphical user interface.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIbus or the Web GUI on a new machine with a version of the DE currently installed.

To install Tivoli Netcool/OMNIbus:

1. From Windows Explorer, navigate to the directory where you extracted the contents of the downloaded package.
2. Either start the installation by using the launchpad, or directly run the command to start the installation program:

- Double-click the following file to start the launchpad:

launchpad.exe

When the launchpad window opens, select a language, and then click through each of the following options in the left navigation pane in order to review the welcome information, prerequisite information, and installation scenarios: Welcome, Prerequisite Information, and Installation Scenarios.

When you are ready to install Tivoli Netcool/OMNIbus, click the Install Product option in the left navigation pane. Then click the **Start Tivoli Netcool/OMNIbus Installation** button that is associated with the Tivoli Netcool/OMNIbus installation. When the IBM Tivoli Netcool/OMNIbus splash screen is displayed, proceed to the next step.

- Double-click the following file to run the installation program:

install.exe

The JRE and installation resources are extracted from the installer archive, and the IBM Tivoli Netcool/OMNIbus splash screen is displayed.

Tip: When running the **install.exe** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.exe** command. The **-r** command-line option must be the last option specified.

3. From the drop-down list at the bottom of the splash screen, select a language and click **OK**.
4. From the Introduction page of the installation wizard, click **Next** to proceed to the Software Licence Agreement page.

Tip: While configuring your installation options, you can click **Previous** to revisit previous pages of the wizard and make changes if required.

5. Read the license agreement and the non-IBM terms, and then accept both the IBM and non-IBM terms. You can also click **Print** to obtain a hardcopy of the license agreement.
6. Click **Next** and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.

The installation location defaults to C:\Program Files\IBM\Common\acsi.

The location of the Deployment Engine will be different if installed by a non-admin user. For more information, see “Directory structure for the Deployment Engine” on page 35.

7. From the Deployment Engine Access Permission page, choose the user access security policy you want to apply to the Deployment Engine:

- **Do not change:** Choose this option to leave the security policy unchanged. Then click **Next**.
- **Single User (current user only):** Choose this option to restrict the use of this Deployment Engine to an Administrator group. Then click **Next**.
- **Group (current user and members of an existing group):** Choose this option to restrict the use of this Deployment Engine to an Administrator group, and members of an existing user group. Then:
 - a. Click **Next**.
 - b. In the Group access entry page, enter the name of an existing user group.

Note: The user group must currently exist as new user groups will not be created by the Deployment Engine. The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

- c. Click **Next**.
- **Global (all users):** Choose this option to allow all users to access the Deployment Engine.
8. From the Select Destination Folder page, specify an installation location for Tivoli Netcool/OMNIBus. This location becomes your NCHOME location. The installation location defaults to C:\IBM\tivoli\netcool.
9. Click **Next** to proceed to the Choose Install Set page. Choose either of the installation options by clicking the associated icon:
 - **Typical:** Choose this option to install all the Tivoli Netcool/OMNIBus features. Then click **Next**.
 - **Custom:** Choose this option if you want to select the features that you want to install. Then:
 - a. Click **Next**.
 - b. From the list box, select each feature that you want to install by selecting its associated check box, or deselect a feature by clearing the associated check box.
 - c. Click **Next** after selecting all the features that you want to install.

Tip: You can switch from a custom to a typical installation by selecting Typical from the **Install Set** drop-down list. Typical selects all features, whereas Custom reverts to your subset of selected features.

After a short interval during which the system is configured, the Pre-Installation Summary page is displayed.

10. Review the installation settings and then click **Install** to start the installation. The Installing Netcool/OMNIBus page shows the progress of the installation. On completion, the Installation Complete page is displayed. This page confirms that the installation was successful and informs you that the system needs to be restarted to complete the installation. Either choose to restart now or later.
11. Click **Done** to close the wizard.
If you started the installation program from the launchpad, and chose to restart later, you can return to the launchpad window and click **Post-Installation** in the navigation pane to review postinstallation information. Then click **Exit** and confirm that you want to exit the launchpad.

Results

The installation adds a number of files to your system.

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related concepts

"Installable Tivoli Netcool/OMNIBus features (Windows)" on page 111

Related tasks

"Obtaining the installation package (Windows)" on page 113

Chapter 6, "Installing, upgrading, and uninstalling the Web GUI component," on page 173

"Viewing the installation log files (Windows)" on page 124

"Performing postinstallation tasks (Windows)" on page 155

"Uninstalling the Deployment Engine" on page 621

Related reference

"Installation directory structure (Windows)" on page 125

"Installation error messages" on page 613

Installing in console mode (Windows)

Run the installation in console mode if you want to complete the installation options by using a series of menus and prompts within a text-based user interface.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIBus or the Web GUI on a new machine with a version of the DE currently installed.

Tip: During the installation, you can enter quit from most of the menu screens to exit the installer. You can also enter back from some of the menu screens to return to the previous screen.

To install Tivoli Netcool/OMNIBus in console mode:

1. From a command prompt, change to the directory where you extracted the contents of the installation package.
2. Enter the following command to run the installation program:
`install.exe -i console`

Tip: When running the **install.exe** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.exe** command. The **-r** command-line option must be the last option specified.

A new command window opens from which you can specify your installation settings.

3. Enter a number that corresponds to the language you want to use for the installation procedure.
4. Read the Introduction information and press Enter, as prompted.
5. To read the non-IBM terms, press 4, and then press Enter to scroll through the text. After reading the non-IBM terms, press q to return to the license agreement. Press Enter to scroll through the license agreement, and then enter 1 to accept the agreement.
6. Press Enter to install the Deployment Engine, or to update an existing version if present.

The installation location defaults to C:\Program Files\IBM\Common\acsi.

The location of the Deployment Engine will be different if installed by a non-admin user. For more information, see "Directory structure for the Deployment Engine" on page 35.

7. From the Deployment Engine Access Permission page, choose the user access security policy you want to apply to the Deployment Engine:
 - Enter 1 to leave the security policy unchanged.
 - Enter 2 to restrict the use of this Deployment Engine to an Administrator group.
 - Enter 3 to restrict the use of this Deployment Engine to an Administrator group, and members of an existing user group. Then:
 - a. In the Group access entry page, enter the name of an existing user group.

Note: The user group must currently exist as new user groups will not be created by the Deployment Engine. The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

- Enter 4 to allow all users to access the Deployment Engine.
8. Specify an installation location for Tivoli Netcool/OMNIBus. This location becomes your NCHOME location.
The installation location defaults to C:\IBM\tivoli\netcool.
 9. Choose the type of installation to perform:

- Enter 1 to install all the Tivoli Netcool/OMNIBus features.
- Enter 2 to select the features that you want to install.

In the resulting menu, all features are initially selected, as denoted by [X], so enter a comma-separated list of numbers that correspond to the features you do *not* want to install. (Note that this screen has a toggle function that enables you to type a number and press Enter to deselect a selected feature denoted by [X], or to type a number and press Enter to select a deselected feature denoted by [].)

Your list of selected features is then shown. You can either enter back to return to the previous screen and revise your selection, or press Enter to confirm your selection and continue.

10. Review the pre-installation summary, and verify that all your required features are selected. Then press Enter to start the installation. On completion, a confirmation message is displayed. You are also informed that your computer must be restarted for the upgrade process to complete.
11. Press Enter to exit the installer.
12. Reboot your computer to complete the process.

Results

The installation adds a number of files to your system.

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (Windows)” on page 111

Related tasks

“Obtaining the installation package (Windows)” on page 113

“Viewing the installation log files (Windows)” on page 124

“Performing postinstallation tasks (Windows)” on page 155

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation directory structure (Windows)” on page 125

“Installation error messages” on page 613

Installing in silent mode (Windows)

Run the installation in silent mode if you want to deploy Tivoli Netcool/OMNIBus with identical installation configurations on multiple workstations. In silent mode, the installer suppresses the graphical or text interface and obtains the installation settings from a predefined response file.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database before installing Tivoli Netcool/OMNIbus or the Web GUI on a new machine with a version of the DE currently installed.

The silent mode of installation has two parts:

1. Define your installation settings in a response file.
2. Run the installation program with the settings in this file.

Related tasks

“Obtaining the installation package (Windows)” on page 113

Defining your installation settings in a response file (Windows)

Before you can run the installation program in silent mode, you must create a response file that defines the features you want to install.

The installation package includes a sample response file that is located in the directory where you extracted the package. The file is called OMNIbus-response.txt. Make a copy of the sample file and use the copy to specify your installation options.

Note: If you previously ran the wizard or console installer with the -r command-line option in order to save your installation settings to an auto-generated installer.properties file, you can use this file as your response file.

Note: When specifying the installation location, use two backslashes (\\) as the path separator because a single backslash (\) is interpreted as an escape character. For example: C:\\IBM\\tivoli\\netcool.

To create a response file with your preferred installation options:

1. Copy the OMNIbus-response.txt file and rename it appropriately. You can store this file in the same location as the extracted installation files or in another location.
2. Edit the configuration values in your copy of the response file as follows. Do not add spaces before or after the values that you specify.

INSTALLER_UI

Do not change this configuration value from the default SILENT setting.

LICENSE_ACCEPTED

Set this value to true to indicate your acceptance of the licence agreement. If you run the installer with this value set to false, the installation process terminates.

USER_INSTALL_DIR

Specify the location to which you want to install Tivoli Netcool/OMNIbus.

CHOSEN_INSTALL_SET

Specify the installable features as follows:

- To install all the features, leave the following lines commented out, as given by default:
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
- To install a subset of the features:
 - a. Uncomment the lines beginning:

```
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
```

- b. Leave the value of **CHOSEN_INSTALL_SET** as Custom.
- c. Delete any features that you do not want to install from the list of comma-separated values given for **CHOSEN_INSTALL_FEATURE_LIST**. You must delete the nco_ value and the comma that follows. Spaces are not required in this list, and the last value does not require a comma.

SKIP_DE_PRECHECKS

Controls whether the installation is terminated if one of the Deployment Engine (DE) prechecks is failed. Possible values are as follows:

- **true**: If the installation fails the DE prechecks, the installation continues.
- **false**: If the installation fails any of the DE prechecks, the installation is terminated and a warning message is sent to the log file.

The DE prechecks might be failed depending on whether you are installing as root or a non-root user, and on whether a root instance of the DE has already been installed. The following table describes the conditions under which a precheck might be failed, depending on which user is installing the product.

Table 34. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

DE_SECURITY_MODE

When you install as root or as an Administrative user, a root instance of the Deployment Engine (DE) is installed on the server. You can select the user access policy to apply to this global instance of the DE by

selecting an option for the **DE_SECURITY_MODE** parameter. Alternatively, you can skip this step and change the DE access policy at any time after installation by using the **de_security** script.

Valid options for **DE_SECURITY_MODE** are as follows:

- 0 - No change will be made (default).
- 1 - Single user (current user).
- 2 - Group (current user plus members of an existing user group).
- 3 - Global (all users).

If you use the 'Group' security mode (option 2), you must set the **DE_GROUP_NAME** parameter to a valid user group.

Note: The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

3. Save the response file.

Related concepts

"Installable Tivoli Netcool/OMNIBus features (Windows)" on page 111

Running the installation program with the silent mode settings (Windows)

After you create the response file that defines which features you want to install, run the installer in silent mode.

Note: No configuration options are displayed during installation.

To install Tivoli Netcool/OMNIBus in silent mode:

1. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
2. Enter the following command to run the installation program:

```
install.exe -i silent -f full_path_to_filename
```

The *full_path_to_filename* value defines the full directory path and file name of the response file that contains your installation settings. If the path includes spaces, enclose it in quotation marks " ".

Wait for the installation to complete.

3. Reboot your computer to complete the process.

Results

The installation adds a number of files to your system.

What to do next

You can view the installation log files to review the installation messages. Before attempting to run Tivoli Netcool/OMNIBus, you must perform some postinstallation tasks. This includes installing and configuring the required probe and gateway components. If the installation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the installation process.

Related tasks

“Viewing the installation log files (Windows)” on page 124

“Performing postinstallation tasks (Windows)” on page 155

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation directory structure (Windows)” on page 125

“Installation error messages” on page 613

Verifying the Netcool/OMNIbus installation (Windows)

The installation process results in a package installation of the Tivoli Netcool/OMNIbus components. After you have installed the Tivoli Netcool/OMNIbus server-side components, run the **nco_objserv.exe -version** utility to verify that the ObjectServer and its subcomponents have been successfully installed.

To verify that the ObjectServer and its subcomponents have been successfully installed:

1. Go to the %NCHOME%\omnibus\bin directory.
2. Enter the following command to run the utility:

```
./nco_objserv.exe -version
```

The following example shows the output of the command and confirms a successful installation. The revision and version numbers that are displayed are specific to your installation.

```
Netcool/OMNIbus Object Server - Version 7.3.1  
(C) Copyright IBM Corp. 1994, 2007
```

```
Library Revisions:  
libnetcool: 5.21.23  
libncmd: 5.21.23  
libnregion: 5.21.23  
libnmemstore: 5.21.23  
libnstore: 5.21.23  
libnproc: 5.21.23  
libnauto: 5.21.23  
libnipc: 5.21.23  
libnstk: 5.21.23  
libniduc_server: 5.21.23  
libnoam: 5.21.23  
libnsecurity: 5.21.23  
libnobjserv: 5.21.23  
network::ipv6: 5.21.23
```

```
Compilation Date:      Thu Dec 16 02:12:43 GMTST 2010  
Compilation Machine:  
Compilation System:    WIN2008-BUILD1
```

```
Code Generation: PRODUCTION
```

```
Registered debug facilities:  
nco_objserv_profiler_timings[ON], signal[ON], thread[ON], cmd[ON],  
region[ON], objserv[ON], nco_objserv_profiler[ON], ipc_s_rpc[ON],  
memstore[ON], nco_objserv[ON], clock[ON], timer[ON], sec_author[ON],  
prop_mgr[ON], cb_mgr[ON], arg_mgr[ON], ipc_s_res[ON], noam[ON],  
nco_objserv_vvtr[ON], ipc_s_not[ON], store[ON], auto[ON], proc[ON],  
region_mutation[ON], sec_audit[ON], module[ON], ipc_s_ini[ON],  
ipc_s_mut[ON], ipc_s_evt[ON]
```

If the command fails to generate output similar to this example, it might indicate one or more components have not installed successfully. Check the installation log files and review the installation messages to identify any problems with the installation.

Related tasks

“Viewing the installation log files (Windows)”

Viewing the installation log files (Windows)

The installation process generates a set of log files for the Tivoli Netcool/OMNIBus and Deployment Engine installations. You can use these files to verify that you installed Tivoli Netcool/OMNIBus successfully, or to troubleshoot your installation.

The installation log files are saved to different locations depending on the user who installs the product.

To view the installation log files:

- Open the InstallAnywhere log file for Tivoli Netcool/OMNIBus in the following location:

`C:\Documents and Settings\username\IA-Netcool-OMNIBus-hostname-
yyyymmddThhmmss-0.log`

Where *username* is the name of the logged-in user and *yyyymmddThhmmss* is the date and time the log file was first generated. Additional log files can be generated on subsequent modifications to the installation.

- Open the Tivoli Netcool/OMNIBus installation log file in the following location:

`%NCHOME%\OMNIBus_InstallLog.log`

- Open the Deployment Engine log files in the following location:

`C:\Program Files\IBM\Common\acsi\logs\username`

Where *username* is the name of the logged-in user.

Viewing installed packages (Windows)

The installer installs products as packages and you can view the versions of all installed packages at any time. You might be asked for package information by IBM Software Support.

To view installed packages:

1. From a command prompt, change to the following location: `C:\Program Files\IBM\Common\acsi\bin.`
2. Enter the following command: `listIU.cmd`

The list of installed packages and their versions is displayed.

Note: To generate more detailed package information, in the form of an XML file, from the same location enter the following command: `de_lsrootiu.cmd`. A file named `package.xml` is created. The file can either be examined or submitted to IBM Software Support.

Results

The list of installed packages, either presented on the command-line interface or written to the `package.xml` file, is formatted as follows: *VV.RR.MM.FF*, where *VV* represents the version number, *RR* represents the release number, *MM* represents the modification number and *FF* represents the fix pack number. Because a fix pack

might not update all the installed components, the fix pack levels can vary. The following example shows a sample output of the command:

```
IU UUID: 90A4E4492B83AEBAB0A24E2ADFCDA0E7 Name: SIU-ProbeSupport Version: 7.3.1.2
IU UUID: E852091AEE2609406F9E5EBC6C27BCB3 Name: SIU-Gateways Version: 7.3.1.1
IU UUID: 374A0E3526BE4C3E96F81987322C90F5 Name: SIU-Desktop Version: 7.3.1.0
```

Installation directory structure (Windows)

Packages are installed in various locations in the Netcool home directory (%NCHOME%) during the Tivoli Netcool/OMNIbus installation.

Packages common to products installed in the same %NCHOME% location

The following table describes the directories for common packages that are shared by products installed in the same Netcool home directory.

Table 35. Directories for common packages

Directory location	Description
%NCHOME%_uninst	Location of the files for uninstalling Tivoli Netcool/OMNIbus.
%NCHOME%\bin	Location of the iKeyman utilities.
%NCHOME%\ini	Location of the connections data file (sql.ini) and the localization configuration file (tds.dat).
%NCHOME%\ini\default	Location of the default reference versions of the connections data file (sql.ini) and the localization configuration file (tds.dat).
%NCHOME%\ini\security	Location of the FIPS 140-2 configuration file (fips.conf) that is required for FIPS 140-2 initialization on Tivoli Netcool/OMNIbus.
%NCHOME%\ini\security\keys	Location of the key database files that are created for managing digital certificates and Secure Sockets Layer (SSL) connections.
%NCHOME%\license	Location of IBM and non-IBM license files.
%NCHOME%\locales	Location of the language files for messages.
%NCHOME%\log	Location of the communication log file for the ObjectServer.
%NCHOME%\platform	Location of internal programs and libraries used by Tivoli Netcool/OMNIbus.
%NCHOME%\var	Location of the gateway log files.

Tivoli Netcool/OMNIbus packages

The following table describes the directories that are specific to Tivoli Netcool/OMNIbus.

Table 36. Tivoli Netcool/OMNIbus directories

Directory location	Description
%NCHOME%\omnibus\bin	Location of the Tivoli Netcool/OMNIbus executable files, and the IEHS executable files for starting and stopping an IEHS server that is running locally in standalone mode, or in information center mode.

Table 36. Tivoli Netcool/OMNIBus directories (continued)

Directory location	Description
%NCHOME%\omnibus\db	Location of the ObjectServer database files.
%NCHOME%\omnibus\desktop	Location of the Desktop executable files and libraries.
%NCHOME%\omnibus\etc	Location of the configuration files that the database initialization utility (nco_dbinit) requires to create an ObjectServer, and configuration files that can be used to upgrade the database schema. This location also holds properties files. You can modify these files.
%NCHOME%\omnibus\etc\default	Location of default reference versions of the properties files, and configuration files that are used by the nco_dbinit utility.
%NCHOME%\omnibus\etc\initial	Location of the writable copy of the ObjectServer source properties file (NCOMS.props), which is used by the nco_dbinit utility.
%NCHOME%\omnibus\extensions	Location of resources that you can use to extend the functionality of Tivoli Netcool/OMNIBus.
%NCHOME%\omnibus\ini	Location of the Desktop configuration files. This location also holds the configuration file for setting the values to run the online help system in information center mode. You can modify these files.
%NCHOME%\omnibus\ini\default	Location of reference versions of the Desktop and online help configuration files.
%NCHOME%\omnibus\install	Location of the installation resources for probes and gateways.
%NCHOME%\omnibus\java	Location of .jar files that support Java applications.
%NCHOME%\omnibus\locales	Location of language localization files.
%NCHOME%\omnibus\log	Location of the majority of the ObjectServer log files. (The ObjectServer communication log file is in %NCHOME%\log.)
%NCHOME%\omnibus\patches	Location of data required by the patching system.
%NCHOME%\omnibus\platform	Location of the platform-dependent Tivoli Netcool/OMNIBus libraries, modules, and IEHS files.
%NCHOME%\omnibus\savedconfig	This directory is visible only after running a V7.0 product upgrade, and stores the old configuration data that needs to be migrated into the new installation. You can delete this directory after confirming that your configuration and data have been safely migrated.
%NCHOME%\omnibus\scripts	Location of scripts that were migrated from a previous installation. The scripts directory is present only if you had a scripts directory, which was migrated during a V7.3.1 upgrade.
%NCHOME%\omnibus\tsm	Location for TSMs.
%NCHOME%\omnibus\upgrade	Location of the Tivoli Netcool/OMNIBus upgrade scripts, which migrate configuration data from a previous installation into a V7.3.1 installation.
%NCHOME%\omnibus\utils	Location of utilities that were migrated from a previous installation. The utils directory is only present if you had a utils directory, which was migrated during a V7.3.1 upgrade.

Table 36. Tivoli Netcool/OMNibus directories (continued)

Directory location	Description
%NCHOME%\omnibus\var	Location where internal runtime information is stored.

Probes

The following table describes the probe directory.

Table 37. Probes directory

Directory location	Description
%NCHOME%\omnibus\probes	Location where probes are installed.
%NCHOME%\omnibus\probes\win32	Location where the probe's configuration files are stored. For example, the properties and rules files.

Gateways

The following table describes the gateway directory.

Table 38. Gateways directory

Directory location	Description
%NCHOME%\omnibus\gates	Location where configuration data for the ObjectServer gateways is stored.

Deployment Engine

The Deployment Engine files are saved to different locations depending on the user who installs the product. The following table describes the Deployment Engine directories.

Table 39. Deployment Engine directories

Directory location	Description
C:\Program Files\IBM\Common\acsi	Location where the Deployment Engine files and scripts are stored.

Upgrading on Windows

On Windows systems, you can run the upgrade program for Tivoli Netcool/OMNibus by using the GUI or silent installation mode.

A Tivoli Netcool/OMNibus upgrade includes any of the following actions:

- Installing a new version of Tivoli Netcool/OMNibus V7.3.1 into an environment with a previous installation of V7.1, V7.2, or V7.2.1. The upgrade process retains the files in their current installation location and writes new or upgraded files into this location.
- Modifying your Tivoli Netcool/OMNibus V7.3.1 installation. This involves adding features to an existing V7.3.1 installation.

Note: If you want to upgrade from V3.6, first upgrade to V7.2.1, and then upgrade to V7.3.1. If you want to upgrade from V7.0, first upgrade to V7.3, and then

upgrade to V7.3.1. For information about upgrading from V3.6 to V7.2.1, and from V7.0 to V7.3, see the Tivoli Netcool/OMNIBus V7.2.1 documentation and the V7.3 documentation in the IBM Tivoli Network Management Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>.

Before you start the upgrade process, review the list of installable features for Tivoli Netcool/OMNIBus. If upgrading from V7.1, or earlier, two new features are available for selection: the Accelerated Event Notification Client, and the online help system, which runs on an IBM Eclipse Help System server. You must select the online help feature if you want to access the help files locally, in standalone mode. If your system is being set up so that the help files are accessed on a remote server, in information center mode, you do not have to install the online help feature, but will have to configure it after installation.

Tivoli Netcool/OMNIBus V7.2.1, or later, can operate in FIPS 140–2 mode. If you want to upgrade your current installation so that it runs in this mode, some initial configuration of your existing data might be required either before or after you upgrade.

Guidelines for upgrading to FIPS 140–2 mode (Windows)

If you want to upgrade to V7.3.1 in FIPS 140-2 mode, some initial configuration is required if the user passwords in your system are currently encrypted by using the DES algorithm, or if you are using property value encryption to encrypt string values in properties files.

If your existing installation uses DES encryption for passwords, you must change the encryption scheme to AES either before or after upgrading. You can then configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.

Upgrading from an installation with DES-encrypted user passwords (Windows)

When in FIPS 140–2 mode, the Advanced Encryption Standard (AES) algorithm must be used to encrypt user passwords that are stored in the ObjectServer.

If your existing installation uses DES encryption for passwords, you must change the encryption scheme to AES either before or after upgrading. You can then configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.

If you are running Tivoli Netcool/OMNIBus V7.1 or later, the encryption algorithm is either DES or AES. Check the value of the ObjectServer **PasswordEncryption** property to see whether it is set to DES or to AES.

To upgrade to V7.3.1 in FIPS 140–2 mode, perform the following set of actions *after* upgrading:

1. Upgrade to V7.3.1
2. In the V7.3.1 system, change the setting of the ObjectServer **PasswordEncryption** property to AES.
3. Ensure that all user passwords are changed or reset. The passwords are now AES encrypted. (See the information that follows this table for guidelines about how to change or reset passwords.)
4. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.
5. Restart Tivoli Netcool/OMNIBus.

Guidelines for changing or resetting passwords

You can use the SQL interactive interface (**isql**) for changing or resetting passwords.

If you ask users to change their passwords, you must to verify that the changes have been made, and will most likely have to send out reminders. To verify whether all passwords have been changed or to identify which ones still need to be changed, perform either of the following steps:

- Start the SQL interactive interface and then enter the following command:

```
select UserName,Passwd from security.users;
```

Check the length of the encrypted passwords returned. Passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

For information about starting the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

- From Netcool/OMNIBus Administrator:
 1. Connect to the relevant ObjectServer. Then click the **System** menu button and click **Databases** to open the Databases, Tables and Columns pane.
 2. Select the **security** database and the **users** table, and then click the **Data View** tab in the Databases, Tables and Columns pane to view user data.
In the **Passwd** column, passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

A system administrator can reset user passwords from the SQL interactive interface as follows:

```
alter user 'username' set password 'password';
```

Where *username* is the name of the user and *password* is their new password.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related reference

“Configuring the JRE for FIPS 140–2 mode (Windows)” on page 158

Upgrading from an installation with property value encryption (Windows)

When in FIPS 140–2 mode, property value encryption must be performed by using an algorithm and mode of operation defined as AES_FIPS. Property value encryption is used to encrypt string values in a properties file or configuration file so that the strings cannot be read without a key.

Tip: You can skip this task if you are upgrading from a V7.2.1 system that operates in FIPS 140-2 mode.

If your existing installation uses property value encryption with the AES algorithm, or uses the **nco_g_crypt** and **nco_pa_crypt** utilities to encrypt passwords, these encrypted values do not meet the requirements for FIPS 140–2 operation. (In V7.2, or earlier, the AES algorithm is used by default for property value encryption; in V7.2.1, AES can be explicitly specified.) To run your upgraded system in FIPS 140–2 mode, you must decrypt these values and then encrypt them

again by using the AES_FIPS algorithm. You must perform this task for each ObjectServer, proxy server, process agent, probe, and gateway that uses encrypted property values, including passwords.

To upgrade property value (and password) encryption for FIPS 140–2 mode, follow these guidelines:

1. In your existing installation, identify any keys that were generated by using the command-line key generator **nco_keygen**.

Tip: The **nco_keygen** utility stores keys within key files. You should be able to identify any key files used by checking the **ConfigKeyFile** property settings in your properties files.

2. Using the keys in your existing installation, decrypt all encrypted properties and passwords in your properties and configuration files by running the **nco_aes_crypt** utility with the **-d** command-line option.
3. Upgrade to V7.3.1.
4. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.
5. Encrypt the values again by using the **nco_keygen** utility to generate one or more new keys, and then running the **nco_aes_crypt** utility with the relevant key file setting.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related reference

“Property value encryption” on page 356

“nco_aes_crypt command-line options” on page 358

“Configuring the JRE for FIPS 140–2 mode (Windows)” on page 158

Upgrading using the installation wizard (Windows)

Use the wizard to present upgrade options within wizard pages in a graphical user interface. In this mode, you can choose to automatically migrate data from a previous installation during the upgrade process.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIBus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIBus or the Web GUI.

To upgrade Tivoli Netcool/OMNIBus:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running. If you are upgrading from V7.2 or later, also ensure that you shut down the IBM Eclipse Help System (IEHS) server from the command line as follows:
 - If you are running the IEHS server in standalone mode, enter the following command on the local computer:
`%NCHOME%\omnibus\bin\help_end.bat`
 - If you are running the IEHS server in information center mode on a remote server, enter the following command on the server computer:
`%NCHOME%\omnibus\bin\IC_end.bat`

2. If any services in your existing installation were manually installed, enter the following command to manually uninstall them.

```
%OMNIHOME%\bin\nco_name /remove [/instance ID]
```

Square brackets denote an optional command-line option. In this command %OMNIHOME% represents %NCHOME%\omnibus. Replace:

- *nco_name* with the executable name for a server component, or a probe or gateway.
- *ID* with a unique instance identifier that might have been specified when the service was installed.

For example, to uninstall an ObjectServer service with an OBJTWO ID, enter:

```
%OMNIHOME%\bin\nco_objserv /remove /instance OBJTWO
```

3. Back up the directory contents of your existing installation to a different location. When you run the installation program, the installer automatically detects and uninstalls an existing installation, or overwrites the files.
4. Navigate to the directory where you extracted the contents of the downloaded package.
5. Either start the installation by using the launchpad, or directly run the command to start the installation program:

- Double-click launchpad.exe to start the launchpad.

When the launchpad window opens, select a language, and then click through each of the following options in the left navigation pane in order to review the welcome information, prerequisite information, and installation scenarios: Welcome, Prerequisite Information, and Installation Scenarios.

When you are ready to install Tivoli Netcool/OMNIBus, click the Install Product option in the left navigation pane. Then click the **Start Tivoli Netcool/OMNIBus Installation** button that is associated with the Tivoli Netcool/OMNIBus installation. When the IBM Tivoli Netcool/OMNIBus splash screen is displayed, proceed to the next step.

- Double-click install.exe to run the installation program.

The JRE and installation resources are extracted from the installer archive, and the IBM Tivoli Netcool/OMNIBus splash screen is displayed.

Tip: When running the **install.exe** command, you can use the **-r** command-line option to save your installation settings to a response file named **installer.properties** that can later be used to run silent installations. No value is required for **-r**, and the **installer.properties** file is generated in the directory that contains the **install.exe** command. The **-r** command-line option must be the last option specified.

6. From the drop-down list at the bottom of the splash screen, select a language and click **OK**.
7. From the Introduction page of the installation wizard, click **Next** to proceed to the Software Licence Agreement page.

Tip: While configuring your installation options, you can click **Previous** to revisit previous pages of the wizard and make changes if required.

8. Read the license agreement and the non-IBM terms, and then accept both the IBM and non-IBM terms. You can also click **Print** to obtain a hardcopy of the license agreement.
9. Click **Next** and wait while the wizard installs or updates the Deployment Engine on the computer, as relevant.

The installation location defaults to C:\Program Files\IBM\Common\acsi.

The location of the Deployment Engine will be different if installed by a non-admin user. For more information, see "Directory structure for the Deployment Engine" on page 35.

10. From the Select Destination Folder page, specify an installation location for Tivoli Netcool/OMNIbus. This location becomes your NCHOME location. The installation location defaults to C:\IBM\tivoli\netcool.
11. Click **Next**. One of the following results occurs:
 - If you chose a different installation location, the "Previous Version of Tivoli Netcool/OMNIbus Detected" page is displayed. This page informs you that a previous version of Tivoli Netcool/OMNIbus has been detected, which must be removed before a new version can be installed. (This option uninstalls the previous version, leaving only the configuration data behind for migration). Click **Next** to progress to the Choose Install Set page.
 - If you chose the same installation location, the "Previous Version of Tivoli Netcool/OMNIbus Detected" page is displayed, informing you that a previous version has been detected. Click **Next** to progress to the Choose Install Set page. (If you choose the same location, the V7.3.1 files are installed within the existing location, and the configuration data from the previous installation will remain in place.)
12. From the Choose Install Set page, choose either of the installation options by clicking the associated icon:
 - **Typical**: Choose this option to install all the Tivoli Netcool/OMNIbus features. Then click **Next**.
 - **Custom**: Choose this option if you want to select the features that you want to install. Then:
 - a. Click **Next**.
 - b. From the list box, select each feature that you want to install by selecting its associated check box, or deselect a feature by clearing the associated check box.
 - c. Click **Next** after selecting all the features that you want to install.

Tip: You can switch from a custom to a typical installation by selecting Typical from the **Install Set** drop-down list. Typical selects all features, whereas Custom reverts to your subset of selected features.

After a short interval during which the system is configured, the Pre-Installation Summary page is displayed.

13. Review the installation settings and then click **Install** to start the installation. The Installing Netcool/OMNIbus page shows the progress of the installation. If you chose a different upgrade location, the configuration data is copied across.

On completion, the Installation Complete page is displayed. This page confirms that the installation was successful and informs you that the system needs to be restarted to complete the installation. Either choose to restart now or later.
14. Click **Done** to close the wizard.

If you started the installation program from the launchpad, and chose to restart later, you can return to the launchpad window and click **Post-Installation** in the navigation pane to review postinstallation information. Then click **Exit** and confirm that you want to exit the launchpad.

What to do next

You can open the installation log files to review the installation messages. Also review the migration log file to see if there were any problems.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIbus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related concepts

“Installable Tivoli Netcool/OMNIbus features (Windows)” on page 111

Related tasks

“Obtaining the installation package (Windows)” on page 113

“Viewing the migration log file (Windows)” on page 140

“Performing postinstallation tasks (Windows)” on page 155

Related reference

“Installation directory structure (Windows)” on page 125

“Additional upgrade and migration notes (Windows)” on page 141

Upgrading in console mode (Windows)

Use the console mode to present the upgrade options as a series of menus and prompts in a text-based user interface. In this mode, your data is automatically migrated from a previous installation during the upgrade process.

You can specify the same installation location, or a different location:

- If you specify the same location as your existing installation, the V7.3.1 files are installed within that location, and the configuration data from your previous installation will remain in place.
- If you specify a different location from your existing installation, the existing version is uninstalled, leaving only the configuration data behind for migration. As part of the upgrade process, these files are automatically migrated after V7.3.1 is installed.

Before you begin

Obtain the installation package for your operating system and extract the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIbus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIbus or the Web GUI.

Tip: During the installation, you can enter quit from most of the menu screens to exit the installer. You can also enter back from some of the menu screens to return to the previous screen.

To upgrade Tivoli Netcool/OMNIbus in console mode:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running. If you are upgrading from V7.2 or v7.2.1, also ensure that you shut down the IBM Eclipse Help System (IEHS) server from the command line as follows:
 - If you are running the IEHS server in standalone mode, enter the following command on the local computer:
`%NCHOME%\omnibus\bin\help_end.bat`
 - If you are running the IEHS server in information center mode on a remote server, enter the following command on the server computer:
`%NCHOME%\omnibus\bin\IC_end.bat`
2. If any services in your existing installation were manually installed, enter the following command to manually uninstall them.
`%OMNIHOME%\bin\ncn_name /remove [/instance ID]`
 Square brackets denote an optional command-line option. In this command `%OMNIHOME%` represents `%NCHOME%\omnibus`. Replace:
 - `ncn_name` with the executable name for a server component, or a probe or gateway.
 - `ID` with a unique instance identifier that might have been specified when the service was installed.

For example, to uninstall an ObjectServer service with an OBJTWO ID, enter:
`%OMNIHOME%\bin\ncn_objserv /remove /instance OBJTWO`
3. Back up the directory contents of your existing installation to a different location. When you run the installation program, the installer will automatically detect and uninstall an existing installation, or overwrite the files, as part of the process.
4. From a command prompt, change to the directory where you extracted the contents of the installation package.
5. Enter the following command to run the installation program:
`install.exe -i console`

Tip: When running the **install.exe** command, you can use the **-r** command-line option to save your installation settings to a response file named `installer.properties` that can later be used to run silent installations. No value is required for **-r**, and the `installer.properties` file is generated in the directory that contains the **install.exe** command. The **-r** command-line option must be the last option specified.

A new command window opens from which you can specify your installation settings.

6. Enter a number that corresponds to the language you want to use for the installation procedure.
7. Read the Introduction information and press Enter, as prompted.
8. To read the non-IBM terms, press 4, and then press Enter to scroll through the text. After reading the non-IBM terms, press q to return to the license agreement. Press Enter to scroll through the license agreement, and then enter 1 to accept the agreement.
9. Press Enter to install the Deployment Engine, or to update an existing version if present.

The installation location defaults to `C:\Program Files\IBM\Common\acsi`.

The location of the Deployment Engine will be different if installed by a non-admin user. For more information, see “Directory structure for the Deployment Engine” on page 35.

10. Specify an installation location for Tivoli Netcool/OMNIbus. This location becomes your NCHOME location.
The installation location defaults to C:\IBM\tivoli\netcool.
11. Press Enter. One of the following results occurs:
 - If you chose a different installation location, you are informed that a previous version of Tivoli Netcool/OMNIbus has been detected, which must be removed before a new version can be installed. (This option will uninstall the previous version, leaving only the configuration data behind for migration). Press Enter to progress to the Choose Install Set screen.
 - If you chose the same installation location, you are informed that a previous version has been detected. Press Enter to progress to the Choose Install Set screen. (If you choose the same location, the V7.3.1 files are installed within the existing location, and the configuration data from the previous installation will remain in place.)
12. Choose the type of installation to perform:
 - Enter 1 to install all the Tivoli Netcool/OMNIbus features.
 - Enter 2 to select the features that you want to install.
In the resulting menu, all features are initially selected, as denoted by [X], so enter a comma-separated list of numbers that correspond to the features you do *not* want to install. (Note that this screen has a toggle function that enables you to type a number and press Enter to deselect a selected feature denoted by [X], or to type a number and press Enter to select a deselected feature denoted by [].)

Your list of selected features is then shown. You can either enter back to return to the previous screen and revise your selection, or press Enter to confirm your selection and continue.
13. When the pre-installation summary is displayed, review the information and then press Enter to start the upgrade. As part of this process, the configuration files from your previous installation are migrated, as relevant. On completion, a confirmation message is displayed. You are also informed that your computer must be restarted for the upgrade process to complete.
14. Press Enter to exit the installer.
15. To complete the process, reboot the server on which you performed the upgrade.

Results

The upgrade adds a number of files to your system.

What to do next

You can open the installation log files to review the installation messages. Also review the migration log file to see if there were any problems.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIbus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (Windows)” on page 111

Related tasks

“Obtaining the installation package (Windows)” on page 113

“Viewing the migration log file (Windows)” on page 140

“Performing postinstallation tasks (Windows)” on page 155

Related reference

“Installation directory structure (Windows)” on page 125

“Additional upgrade and migration notes (Windows)” on page 141

Upgrading in silent mode (Windows)

Run the upgrade in silent mode for remote installations, or to propagate one configuration to multiple workstations. In silent mode, the installer suppresses the graphical or text interface and obtains the installation settings from a predefined response file.

Before you begin

You must have obtained the installation package for your operating system and extracted the contents.

Note: You must back up the DE database, the Tivoli Netcool/OMNIBus home directory, and the Web GUI configuration data before upgrading Tivoli Netcool/OMNIBus or the Web GUI.

The silent mode of installation is a two-step operation that requires you to define your installation settings in a response file and then run the upgrade program with the silent mode settings.

Related tasks

“Obtaining the installation package (Windows)” on page 113

Defining your Windows upgrade settings in a response file

Before you can run the upgrade program in silent mode, you must create a response file that defines the features you want to install.

The installation package includes a sample response file that is located in the directory where you extracted the package. The file is called OMNIBus-response.txt. Make a copy of the sample file and use the copy to specify your installation options.

Note: If you previously ran the wizard or console installer with the -r command-line option in order to save your installation settings to an auto-generated installer.properties file, you can use this file as your response file.

You can specify the same installation location, or a different location in your response file:

- If you specify the same location as your existing installation, the V7.3.1 files are installed within that location, and the configuration data from your previous installation will remain in place.

- If you specify a different location from your existing installation, the existing version is uninstalled, leaving only the configuration data behind for migration. As part of the upgrade process, these files are automatically migrated after V7.3.1 is installed.

Note: When specifying the installation location, use two backslashes (\\) as the path separator because a single backslash (\) is interpreted as an escape character. For example: C:\\IBM\\tivoli\\netcool.

To create a response file with your preferred installation options:

1. Copy the OMNibus-response.txt file and rename it appropriately. You can store this file in the same location as the extracted installation files or in another location.
2. Edit the configuration values in your copy of the response file as follows. Do not add spaces before or after the values that you specify.

INSTALLER_UI

Do not change this configuration value from the default SILENT setting.

LICENSE_ACCEPTED

Set this value to true to indicate your acceptance of the licence agreement. If you run the installer with this value set to false, the installation process terminates.

USER_INSTALL_DIR

Specify the location to which you want to install Tivoli Netcool/OMNIBus.

CHOSEN_INSTALL_SET

Specify the installable features as follows:

- To install all the features, leave the following lines commented out, as given by default:
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
- To install a subset of the features:
 - a. Uncomment the lines beginning:
#CHOSEN_INSTALL_SET...
#CHOSEN_INSTALL_FEATURE_LIST...
 - b. Leave the value of **CHOSEN_INSTALL_SET** as Custom.
 - c. Delete any features that you do not want to install from the list of comma-separated values given for **CHOSEN_INSTALL_FEATURE_LIST**. You must delete the nco_ value and the comma that follows. Spaces are not required in this list, and the last value does not require a comma.

SKIP_DE_PRECHECKS

Controls whether the installation is terminated if one of the Deployment Engine (DE) prechecks is failed. Possible values are as follows:

- true: If the installation fails the DE prechecks, the installation continues.
- false: If the installation fails any of the DE prechecks, the installation is terminated and a warning message is sent to the log file.

The DE prechecks might be failed depending on whether you are installing as root or a non-root user, and on whether a root instance of

the DE has already been installed. The following table describes the conditions under which a precheck might be failed, depending on which user is installing the product.

Table 40. Behavior of the installer in response to DE prechecks

User	Condition	Behavior if SKIP_DE_PRECHECKS is set to true	Behavior if SKIP_DE_PRECHECKS set to false
Root	A root instance of the DE is not installed on the computer.	A global DE is installed.	The installation is terminated. A warning message is generated stating that if you install a root instance of the DE, this DE will be used by any user who installs a DE-based product on that computer, unless that user already has a single-user DE.
Non-root	The non-root user does not have a single-user DE, and a root instance of the DE is already installed on the computer.	The installation is attempted using the root instances of the DE. Use this setting only if you have write-permission to the root instances of the DE and no one else is currently using it.	The installation is terminated. A warning message is generated stating that the root instance of the DE will be used, and that you must have write permissions to the database.

DE_SECURITY_MODE

When you install as root or as an Administrative user, a root instance of the Deployment Engine (DE) is installed on the server. You can select the user access policy to apply to this global instance of the DE by selecting an option for the **DE_SECURITY_MODE** parameter. Alternatively, you can skip this step and change the DE access policy at any time after installation by using the **de_security** script.

Valid options for **DE_SECURITY_MODE** are as follows:

- 0 - No change will be made (default).
- 1 - Single user (current user).
- 2 - Group (current user plus members of an existing user group).
- 3 - Global (all users).

If you use the 'Group' security mode (option 2), you must set the **DE_GROUP_NAME** parameter to a valid user group.

Note: The predefined Windows user groups can produce unexpected results when used by the Deployment Engine. Therefore you must define new user groups and avoid using the predefined user groups.

3. Save the response file.

Related concepts

“Installable Tivoli Netcool/OMNIBus features (Windows)” on page 111

Running the upgrade program with the silent mode settings (Windows)

After you create the response file that defines which features you want to install, run the installer in silent mode.

Note: No configuration options are displayed during installation.

To upgrade Tivoli Netcool/OMNIBus in silent mode:

1. Stop all Tivoli Netcool/OMNIBus processes that are currently running. If you are upgrading from V7.2 or v7.2.1, also ensure that you shut down the IBM Eclipse Help System (IEHS) server from the command line as follows:
 - If you are running the IEHS server in standalone mode, enter the following command on the local computer:
`%NCHOME%\omnibus\bin\help_end.bat`
 - If you are running the IEHS server in information center mode on a remote server, enter the following command on the server computer:
`%NCHOME%\omnibus\bin\IC_end.bat`
2. If any services in your existing installation were manually installed, enter the following command to manually uninstall them. An optional command parameter is shown in square brackets.

```
%OMNIHOME%\bin\ncn_name /remove [/instance ID]
```

In the above command:

- %OMNIHOME% represents %NCHOME%\omnibus.
- ncn_name is the executable name for a Tivoli Netcool/OMNIBus component, probe, or gateway.
- The ID variable represents a unique instance identifier that may have been specified when the service was installed.

For example, to uninstall an ObjectServer service with an OBJTWO ID, enter:

```
%OMNIHOME%\bin\ncn_objserv /remove /instance OBJTWO
```

3. Back up the directory contents of your existing installation to a different location. When you run the installation program, the installer will automatically detect and uninstall an existing installation, or overwrite the files, as part of the process.
4. From a command prompt, change to the directory where you extracted the contents of the downloaded package.
5. Enter the following command to run the upgrade:
`install.exe -i silent -f full_path_to_filename`
The *full_path_to_filename* value defines the full directory path and file name of the response file that contains your upgrade settings. If the path includes spaces, enclose it in quotation marks " ".
Wait for the upgrade to complete.
6. Reboot your computer to complete the process.

Results

The installation adds a number of files to your system.

What to do next

You can open the installation log files to review the installation messages. Also review the migration log file to see if there were any problems.

Additional upgrade and migration tasks are required to complete the upgrade of your system. One of these tasks involves updating the migrated database to the V7.3.1 database schema, which contains additional ObjectServer resources such as new or updated automations, tables, fields, tools, permissions, and conversions.

Before attempting to run Tivoli Netcool/OMNIbus, perform some postinstallation tasks. These tasks include upgrading and configuring the required probe and gateway components.

Related concepts

“Installable Tivoli Netcool/OMNIbus features (Windows)” on page 111

Related tasks

“Viewing the migration log file (Windows)”

“Performing postinstallation tasks (Windows)” on page 155

Related reference

“Installation directory structure (Windows)” on page 125

“Additional upgrade and migration notes (Windows)” on page 141

Viewing the migration log file (Windows)

After upgrading Tivoli Netcool/OMNIbus, and migrating your existing data into the new installation, you can review the migration log file to ensure the process was successful, or for troubleshooting purposes.

To view the migration log file:

Go to the location `%NCHOME%\omnibus\log` and examine the `migrate.log` file.

Modifying your V7.3.1 installation (Windows)

You can modify your Tivoli Netcool/OMNIbus V7.3.1 installation if you want to install additional components.

Note: To add features to your installation, run the installation again on your NCHOME location and choose which features you want to add. You cannot remove existing features.

To change the set of Tivoli Netcool/OMNIbus V7.3.1 features in an existing installation:

1. Stop all Tivoli Netcool/OMNIbus processes that are currently running.
2. Back up the current `%NCHOME%` directory in case you want to revert to that installation.
3. Run the installation program in GUI or silent mode. If running in silent mode, update the response file with the features to be added before running the installer.
4. Review the installation log file.

Results

Where relevant:

- Any existing packages that are of a lower version are replaced with an equivalent higher version.
- Packages for any new features are installed.

What to do next

Before attempting to run Tivoli Netcool/OMNIbus, you might be required to perform some post-installation tasks, depending on the features added.

Additional upgrade and migration notes (Windows)

Read these notes for additional information about the Tivoli Netcool/OMNIbus upgrade and migration process, and any actions you might be required to perform.

Notes on upgrading ObjectServer schemas to V7.3.1 schemas (Windows)

After upgrading to Tivoli Netcool/OMNIbus V7.3.1, upgrade your ObjectServer schemas to the V7.3.1 schema.

Important: Follow these instructions on each ObjectServer instance that is upgraded from V7.0 (through an upgrade to V7.1), V7.1, V7.2, V7.2.1, or V7.3. In each case, ensure that the ObjectServer is running.

Four .sql import files are provided in Tivoli Netcool/OMNIbus V7.3.1 that contain the required schema changes:

- `update70to71.sql`: This file upgrades a V7.0 ObjectServer schema to a V7.1 schema
- `update71to72.sql`: This file upgrades a V7.1 ObjectServer schema to a V7.2 schema. Note that the V7.2 and V7.2.1 schemas are identical.
- `update72xto73.sql`: This file upgrades a V7.2 or V7.2.1 ObjectServer schema to a V7.3 schema.
- `update73to731.sql`: This file upgrades a V7.3 ObjectServer schema to a V7.3.1 schema.

These files are located in the `%NCHOME%\omnibus\etc` directory.

Note that database initialization is *not* required after upgrading.

V7.0 to V7.1 schema upgrade

To upgrade a V7.0 ObjectServer schema to a V7.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.0 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'c:/tmp/70to72Upgrade/NCOMS';
```
3. Review the `update70to71.sql` file to ensure that it is not altering configuration that you have already added or customized for your business. Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer). In particular, the `connection_watch_disconnect` and `connection_watch_connect` automations were

changed in the V7.1 installation package. When you upgrade the schema, the new triggers are imported as `connection_watch_disconnect2` and `connection_watch_connect2`, and are disabled by default. If you want to use the new triggers, enable them, and then disable the original `connection_watch_disconnect` and `connection_watch_connect` triggers that were available in V7.0.

4. When you have resolved all the conflicts between the import file and the upgraded ObjectServer instance, import the file to the ObjectServer by using the **isql** command. For example:

```
%NCHOME%\omnibus\bin\isql" -U username -P password -S servername
-i update70to71.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.
5. When the import process is completed successfully, review the ObjectServer log file for any errors. If errors exist, you must identify the cause, and resolve the conflicts.
6. When all the conflicts are resolved, apply the V7.1 to V7.2 schema upgrade as described in the next section. Review the ObjectServer log file again for errors and determine whether the configuration of the system is acceptable. If not, revert to the backed-up image, make the necessary changes to the `update70to71.sql` file and reapply the file.

V7.1 to V7.2 or V7.2.1 schema upgrade

To upgrade a V7.1 ObjectServer schema to a V7.2 or V7.2.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.0 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'c:/tmp/70to72Upgrade/NCOMS';
```
3. Review the `update71to72.sql` file to ensure that it is not altering configuration that you have already added or customized for your business. Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer). If you have created your own tools and added them to menus, check the `tools.*` tables for conflicts. There are also several schema changes and new automations that support new functionality in IBM Tivoli Network Manager IP Edition V3.7 (formerly Netcool Precision IP). If you are already using Network Manager, the changes might already have been added to the ObjectServer. If this is the case, remove the duplicated configuration from the `update71to72.sql` file.
4. When you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the **isql** command. For example:

```
%NCHOME%\omnibus\bin\isql" -U username -P password -S servername
-i update71to72.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.
5. After the import process is complete, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine

whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the `update71to72.sql` file and reapply the file.

6. After all the conflicts are resolved, apply the V7.2 to V7.3 schema upgrade, or V7.2.1 to V7.3 schema upgrade, as described in the next section. Review the ObjectServer log file again for errors and determine whether the configuration of the system is acceptable. If not, revert to the backed-up image, make the necessary changes to the `update71to72.sql` file and reapply the file.

V7.2 or V7.2.1 to V7.3 schema upgrade

To upgrade a V7.2 or V7.2.1 ObjectServer schema to a V7.3 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.2 or V7.2.1 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'c:/tmp/70to72Upgrade/NCOMS';
```

3. Review the `update72xto73.sql` file:
 - Ensure that it is not altering configuration that you have already added or customized for your business.
 - Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer).

In particular, the deduplication and `new_row` automations were changed in the V7.3 installation package.

After you upgrade the schema, the new triggers are imported as `deduplication_73` and `new_row_73`, and are disabled by default. If you want to use the new triggers, enable them, and then disable the original deduplication and `new_row` triggers that were available in V7.2 or V7.2.1.

4. When you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the `isql` command. For example:

```
"%NCHOME%\omnibus\bin\isql" -U username -P password -S servername  
-i update72xto73.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.

5. After the import process is complete, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the `update72xto73.sql` file and reapply the file.

V7.3 to V7.3.1 schema upgrade

To upgrade a V7.3 ObjectServer schema to a V7.3.1 schema:

1. Start the SQL interactive interface if necessary.
2. Back up the V7.3 ObjectServer instance using the ALTER SYSTEM BACKUP command. The syntax is:

```
ALTER SYSTEM BACKUP 'directory_name';
```

For example:

```
alter system backup 'c:/tmp/70to72Upgrade/NCOMS';
```

3. Review the `update73to731.sql` file to ensure that it is not altering configuration that you have already added or customized for your business. Make any changes necessary to resolve conflicts and remove unrequired changes from the file (and consequently, the target ObjectServer). In particular, the `disconnect_iduc_missed` trigger has been updated to increase the maximum number of `iduc_missed` signals to 100 before the client is disconnected. This trigger replaces the 7.3 `disconnect_iduc_missed` trigger. The 7.3 trigger should be disabled and the updated trigger enabled before it can be used.
4. When you have resolved all the conflicts between the import file and the ObjectServer instance, import the file to the ObjectServer by using the `nco_sql` command. For example:


```
%NCHOME%\omnibus\bin\isql" -U username -P password -S servername -i update73to731.sql
```

In this command, *username* is a valid user name, *password* is the corresponding password, and *servername* is the name of the ObjectServer.
5. After the import process is completed, review the ObjectServer log file for any errors. If errors exist, identify the cause and resolve the conflicts. Determine whether the configuration of the system is acceptable. If not, revert to the backed up image, make the necessary changes to the `update73to731.sql` file and reapply the file.

Files migrated for an upgrade (Windows)

When you upgrade your Windows installation, a number of files are migrated from the old installation backup directory to the new V7.3.1 installation.

Important:

- Review all the migrated properties files. Where file paths are specified for a property, update the path (if necessary) so that it references the correct location in the new installation, rather than the old installation from which the file was migrated. If you used the `%OMNIHOME%` or `%NCHOME%` environment variable (rather than the expanded value of the variable), you do not need to make any changes because the environment variable automatically resolves to the new location.
- If your previous installation contained ObjectServer files, which were created as storage objects for log or report data, these logical files were stored in the ObjectServer database with a reference to the full directory path of the physical location. If your upgrade path is different from the path of the previous installation, check to see whether you have any file objects that reference the old location, and update the paths so that they reference the new location. You can check the paths from the `catalog.files` table. Alternatively, from the Netcool/OMNIBus Administrator window, select the **System** menu button, and then click **Log Files** to see the file details. For further information about the `catalog.files` table and Netcool/OMNIBus Administrator, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

The following table lists the migrated files and their locations in the new installation.

Table 41. Migrated file locations

File type	Migrated location
Connections data file	<code>%NCHOME%\ini\sql.ini</code>

Table 41. Migrated file locations (continued)

File type	Migrated location
Configuration files	<p>%NCHOME%\omnibus\ini*.props</p> <p>%NCHOME%\omnibus**.conf</p> <p>%NCHOME%\omnibus**.props</p> <p>The upgrade only copies configuration files that use default names; for example, nco_pa.props, *GATE.conf, and *GATE.props. Any other configuration files must be copied manually to the equivalent %NCHOME%\omnibus location.</p>
ObjectServer gateway configuration files	<p>%NCHOME%\omnibus\etc*GATE.props</p> <p>%NCHOME%\omnibus\etc*.tblrep.def</p> <p>%NCHOME%\omnibus\etc*.map</p> <p>%NCHOME%\omnibus\etc*.startup.cmd</p>
Database files	%NCHOME%\omnibus\db
Netcool/OMNIbus Administrator properties file	%NCHOME%\omnibus\etc\nco_config.props
Policy file	%NCHOME%\omnibus\etc\admin.policy
Exclusions file	<p>%NCHOME%\omnibus\etc\exclusions.old.xml</p> <p>If you had previously made changes to the exclusions file in your old installation, you must copy these changes from the migrated exclusions.old.xml file into the %NCHOME%\omnibus\etc\exclusions.xml file in your new installation.</p>
Confpack properties file	%NCHOME%\omnibus\etc\nco_confpack.props
Desktop files	<p>%NCHOME%\omnibus\ini\default.elc</p> <p>The V7.3 UNIX or Linux event list uses locale-specific default.elc and minimal.elc files in the location \$NCHOME/omnibus/desktop/locale/arch/locale.</p> <p>Note: Images and backdrops are no longer required and are not migrated.</p>
Key database files for SSL (V7.2.1)	%NCHOME%\ini\security\keys
FIPS 140-2 configuration file	%NCHOME%\ini\security\fips.conf
Utilities and scripts	<p>If the previous installation contained a utils directory, the upgrade process copies this directory and its contents to:</p> <p>%NCHOME%\omnibus\utils</p> <p>If the previous installation contained a scripts directory, the upgrade process copies this directory and its contents to:</p> <p>%NCHOME%\omnibus\scripts</p>

Table 41. Migrated file locations (continued)

File type	Migrated location
Probe properties and rules files (*rules and *.props)	%NCHOME%\omnibus\probes\migrated Note: All probes must be reinstalled, and the old data migrated into the directory above must be copied to the new probe location.

Guidelines for upgrading to UTF-8 encoding (Windows)

If you previously ran your ObjectServers, ObjectServer Gateways, and supported probes and gateways in the default system encoding on Windows, but want to switch to using UTF-8 encoding, you will need to convert some of your existing configuration files and the ObjectServer data to UTF-8 encoding.

You must convert the following files if they contain non-ASCII characters, to ensure that the files can be parsed properly:

- Convert your existing properties files for the ObjectServer, ObjectServer Gateway, probes, and **nco_dbinit**.
- Convert your existing probe rules files.
- Convert your existing gateway map files.
- Convert any existing customized .sql file for the **nco_dbinit** utility. For example, when creating the ObjectServer, you might have used the -desktopfile command-line option to specify a file other than the default \$NCHOME/omnibus/etc/desktop.sql file, which contains configuration data for the UNIX and Windows desktop.

You must also convert existing data in the ObjectServer from the default system encoding. This involves creating a new ObjectServer in UTF-8 encoding and then using a gateway to transfer the data from the old ObjectServer to the newly-created ObjectServer.

Before you begin

You must have completed the upgrade process and migrated your data. You must also have upgraded the ObjectServer schema.

To upgrade to UTF-8 encoding:

1. Convert your existing non-ASCII properties files, probe rules files, gateway map files, and .sql file, to UTF-8 encoding. You can use tools such as **iconv** and **uconv**, which can convert files from one encoding to another:
 - For further information about using **iconv** on Windows, go to <http://gnuwin32.sourceforge.net/summary.html>.
 - The **uconv** tool is available in the ICU4C binary distribution 4.0.1 which can be downloaded from <http://icu-project.org/download/>. The application can be found under the bin directory.
2. In your upgraded location, overwrite the non-ASCII files with the converted files.
3. Convert your existing ObjectServer data as follows:
 - a. Create a new ObjectServer in UTF-8 encoding by running the **nco_dbinit** utility with the -utf8enabled command-line option set to TRUE, and the -desktopfile command-line option set to the location of the converted .sql file (if available).

- b. If you used the `-desktopfile` command-line option to create the ObjectServer, as specified in step 3a on page 146, you must upgrade the ObjectServer schema because it was created using one of the `.sql` files from an earlier product version. Run the relevant `update*.sql` scripts in the `$NCHOME/omnibus/etc` directory.
- c. If necessary, install a unidirectional ObjectServer Gateway. Then configure the gateway to read data from the old non-UTF-8 ObjectServer, and write data to the new UTF-8 ObjectServer.
- d. Configure server communications across the components by using the **nco_xigen** command in `$NCHOME/omnibus/bin`, or the Server Editor.
- e. Start the old non-UTF-8 ObjectServer, which contains the event data to be converted.
- f. Start the new ObjectServer in UTF-8 encoding by running the **nco_objserv** command with the `-utf8enabled` command-line option set to `TRUE`.
- g. Start the ObjectServer Gateway:
 - If the gateway is running in the same default system encoding as the old non-UTF-8 ObjectServer, the gateway can run with the `-utf8enabled` command-line option set to either `TRUE` or `FALSE`.
 - If the gateway is running in a different encoding from the ObjectServer, you must run the gateway with the `-utf8enabled` command-line option set to `TRUE`.

For information about starting the ObjectServer Gateway, go to the IBM Tivoli Network Management Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>. Locate and expand the *IBM Tivoli Netcool/OMNIBus* node and then go to the *Tivoli Netcool/OMNIBus gateways* node. Look for the ObjectServer Gateway publication.

During the synchronization process, the data is transferred from the old ObjectServer to the new one.

The new ObjectServer will be ready for use after you have performed any other required postinstallation steps. Note that you might be required to update some of your configuration files with the name of the new ObjectServer, and possibly, the various file paths specified.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Related tasks

“Creating an ObjectServer” on page 231

“Starting an ObjectServer manually” on page 237

Related reference

“Notes on upgrading ObjectServer schemas to V7.3.1 schemas (Windows)” on page 141

Migrating your digital certificates and keys (Windows)

Tivoli Netcool/OMNIBus V7.2.1, or later, uses the IBM Key Management (iKeyman) utility as the certificate management tool for Secure Sockets Layer (SSL) communication. If you upgraded from an earlier version that used SSL for client and server communications, and want to continue to use your old certificates in V7.3, you must migrate your certificate files and private keys into the Certificate Management System (CMS) key database that is used for certificate management in V7.3.1

Note: The old certificates were encrypted with non-FIPS 140–2 certified algorithms, so certificate migration is supported only in non-FIPS 140–2 mode. If you intend to operate in FIPS 140–2 mode, you must use iKeyman to re-create all the old certificates that you want to reuse.

The key database is a file that stores digital certificates and keys. In V7.2.1, or later, a key database must be created on each server computer where an ObjectServer, process agent, or proxy server is configured for SSL, and on each client computer that uses SSL connections. The key database also requires a password for access control; this password must be stored in a stash file (`omni.sth`) in the same location as the key database. On Windows, the file name and location of the key database is `%NCHOME%\ini\security\keys\omni.kdb`.

After upgrading to V7.3.1 follow these guidelines to migrate your existing certificates and keys, assuming no key database currently exists:

1. Use the iKeyman GUI to create the key database and its stash file.
2. Run the certificate migration utility (`nco_ssl_migrate`).

V7.2.1 certificates are automatically migrated to V7.3.1

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

“Managing digital certificates for SSL communication” on page 380

Related reference

“Configuring the JRE for FIPS 140–2 mode (Windows)” on page 158

Running the certificate migration utility (Windows)

You can run the certificate migration utility (`nco_ssl_migrate`) on any server or client computer that has a trusted certificate database or server certificates to be migrated.

The certificate migration tool has two modes of operation, which can be used together, or separately:

- Automatic import: The utility locates the `sql.ini` file in the specified location from which you want to migrate certificates, and then locates the properties files for each of the SSL servers that are defined in the `sql.ini` file. The SSL properties are read to determine the location of the SSL certificates, which are then migrated, followed by the certificates in the trusted certificate database. For example, for a V7.2 to V7.3.1 upgrade, the following certificates are migrated into the key database:
 - Server certificates: `%NCHOME%\omnibus\etc\servername.crt`
 - Trusted certificate database: `%NCHOME%\ini\trusted.txt`
- Manual import: This mode of operation provides an additional method for importing certificates that the automatic import process cannot find; for example, process agent certificates, which are defined on the command line.

If you upgraded from V7.1 or V7.2, the upgrade process should have installed the new files into the same location as your previous installation. You can migrate your certificates by using the following methods:

- Automatic import: Run the **nco_ssl_migrate** utility with the **-auto** and **-fromnchome** options.
- Manual import: Migrate the server certificates by running the **nco_ssl_migrate** utility with the **-manual** option, and either or both of the **-servercerts** and **-trusted** options.

To import your SSL certificates and keys into the key database:

1. Create a key database (if one does not already exist). This must be a dedicated key database with the file name **omni.kdb**.
2. From a command prompt, run the following command to migrate the existing certificates into the key database:

```
%NCHOME%\omnibus\bin\nco_ssl_migrate options
```

In this command, *options* represents the command-line options, which are described in the following table.

Table 42. Command-line options for *nco_ssl_migrate*

Command-line option	Description
-auto	Specifies that the existing certificates and keys must be automatically imported from a specific location into the key database. Use the -auto option with the -fromnchome or -fromomnihome option. Either -auto , or -manual , or both, must be specified.
-dumpprops	Displays all the system and nco_ssl_migrate properties and exits. Use this option to verify that command-line settings are being parsed correctly.
-force	Migrates all certificates regardless of their validity. Expired certificates and certificates that are due to become valid at a future date are migrated. Also, certificates in the key database are overwritten if they have the same name as a certificate being migrated. If -force is not specified, only currently valid certificates are imported.
-fromnchome string	Specifies the NCHOME location from which certificates are to be migrated. Use the -fromnchome option with the -auto option to migrate certificates from V7.1 or V7.2. Set the value of -fromnchome to your previous NCHOME location, which should be the same as your new NCHOME location (if you chose to upgrade to the same location). You can set the -fromnchome option to the %NCHOME% environment variable or its expanded value. For example: -fromnchome "%NCHOME%" -fromnchome "C:\Program Files\Micromuse\netcool" Either -fromnchome or -fromomnihome must be specified. If both are specified, the -fromnchome option overrides the -fromomnihome option.
-fromomnihome string	This option is for migrating certificates from V7.0, but is currently not fully supported. Use the manual migration method instead.

Table 42. Command-line options for *nco_ssl_migrate* (continued)

Command-line option	Description
-help	Displays help information about the command-line options and exits.
-manual	Specifies that the existing certificates and keys must be manually imported into the key database. Either -auto, or -manual, or both, must be specified. If -manual is specified, the -servercerts option, or -trusted option, or both, must also be specified to identify the certificates to be imported.
-messagelevel <i>string</i>	Specifies the message logging level. Possible values are: debug, info, warn, error, and fatal. The default level is warn. Messages that are logged at each level are as follows: <ul style="list-style-type: none"> fatal: fatal only error: fatal and error warn: fatal, error, and warn info: fatal, error, warn, and info debug: fatal, error, warn, info, and debug These values can be uppercase, lowercase, or mixed case. Messages are logged to %NCHOME%\omnibus\log\nco_ssl_migrate.0.log, with a maximum limit of 1024 KB. When the file reaches this limit, it is closed and renamed nco_ssl_migrate.1.log, and a new nco_ssl_migrate.0.log file is started. When the new file reaches the maximum size, it is renamed nco_ssl_migrate.1.log, overwriting any existing file, and the process continues.
-nowarn	Indicates that you do not want to be prompted for confirmation of actions. Prompts for passwords will still be displayed if required.
-password <i>string</i>	Only use this command-line option if you want to use a different password to open the key database. The password that you specify overrides the stash file password. If you run nco_ssl_migrate without this command-line option, the stash file is used to open the key database so that the files can be migrated into it.
-servercerts <i>string1;string2;string3, ...</i>	Only use this option in conjunction with the -manual option. Specifies a comma-separated list of server certificates to import, where: <ul style="list-style-type: none"> <i>string1</i> is the name of the server. <i>string2</i> is the file path and name of the certificate. <i>string3</i> is the encrypted private key password, which was encrypted with the nco_g_crypt utility. If you do not specify the encrypted password here, you are prompted for the password later and will have to enter it in plain text. A semi-colon (;) is required to separate the server name, certificate, and password. In the following example, an encrypted password is provided for the first certificate entry, but no password is specified for the second entry. "NCOMS;%NCHOME%\ini\NCOMS.crt;EHEDAIBFAPFM,NCOMSB;%NCHOME%\ini\NCOMSB.crt"

Table 42. Command-line options for `nco_ssl_migrate` (continued)

Command-line option	Description
<code>-trusted string,...</code>	Only use this option in conjunction with the <code>-manual</code> option. Specifies a comma-separated list of trusted signer certificates to import. If all your trusted certificates were stored in the <code>trusted.txt</code> file, specify only this file here; for example: "%NCHOME%\ini\trusted.txt"
<code>-version</code>	Displays version information about the <code>nco_ssl_migrate</code> utility and exits.

Results

When you run `nco_ssl_migrate`, automatic import occurs first, followed by the manual import.

Related tasks

"Notes for SSL connections in FIPS 140–2 mode" on page 380

"Managing digital certificates for SSL communication" on page 380

IBM Tivoli Enterprise Console BAROC data migration (Windows)

Tivoli Netcool/OMNIBus provides integration with Tivoli Enterprise Console.

The Tivoli Enterprise Console product is a rules-based event management application that integrates system, network, database, and application management to help ensure the optimal availability of the IT services of an organization.

In Tivoli Enterprise Console, an event is an object that is created based on data that is obtained from a source that is monitored by an event adapter. Each event is identified by a class name, which the event adapter defines. Class names are used to label events, but each event contains additional information that helps define and locate a potential problem. Event classes can be subclassed to facilitate the further breakdown of information so that more detailed rules can be applied to the information. An adapter formats event information into attributes that contain a name and value, and sends this information to the event server for further processing.

An adapter uses various files for its operations. One of these files is the Basic recorder of objects in C (BAROC) file, which describes the classes of events that the adapter supports, to the event server. The event server must load this file before it can understand events received from the adapter. A BAROC file has a `.baroc` extension.

In Tivoli Netcool/OMNIBus, the ObjectServer stores and processes events in a 'flat' normalized representation, which is not compatible with the class hierarchy and extended attribute format that is adopted for Tivoli Enterprise Console events.

To support the migration of Tivoli Enterprise Console event data, Tivoli Netcool/OMNIBus provides a BAROC tool for converting the data. The ObjectServer schema also provides the following objects to support Tivoli Enterprise Console data migration:

- A master.class_membership table is used to store details of all Tivoli Enterprise Console classes with the class ID, name and parent ID. The BAROC tool populates this table.
- An ExtendedAttr column of data type varchar(4096) within the alerts.status table, stores multiple name-value pairs in one column, in a format compatible with Tivoli Enterprise Console event strings.
- SQL and probe rule functions
 - An instance_of sql function : Returns true if class is a subclass of parent_class or they are equal, using the hierarchy defined in the master.class_membership table.
 - An nvp_exists() sql function: Verifies whether a name-value pair exists.
 - An nvp_get() sql function: Retrieves the value of a specific name-value pair.
 - An nvp_set() sql function: Adds or replaces keys from a name-value pair string and returns the new name-value pair string.
 - An nvp_add() probe rule function: Adds or replaces variables and their values to a name-value pair list or creates a name-value pair list of all variables.
 - An nvp_remove() probe rule function: Removes keys from a name-value pair string and returns the new name-value pair string.

About the BAROC conversion tool (nco_baroc2sql) (Windows)

To support data migration from Tivoli Enterprise Console to Tivoli Netcool/OMNIbus, a tool is provided in Tivoli Netcool/OMNIbus for converting Tivoli Enterprise Console BAROC files to ObjectServer SQL files, which you can then import into the database.

The BAROC tool (**nco_baroc2sql**) is installed when you select the **Servers** feature during the Tivoli Netcool/OMNIbus installation. This tool is located in the \$NCHOME/omnibus/bin directory. You must first run the tool on your .baroc load file to create SQL INSERT statements that are compliant with ObjectServer SQL. These statements are saved to a file that you specify. After generating the SQL output, you must import the data that is defined in the INSERT statements into the ObjectServer database.

When you run the **nco_baroc2sql** tool, it writes an INSERT statement for the ObjectServer master.class_membership table for each class-to-parent relationship that exists in the BAROC file. Where the BAROC class has a multiple inheritance relationship to its parent classes, the **nco_baroc2sql** tool writes an INSERT statement for each class/parent relationship that exists in the BAROC file. The format of the INSERT statement that the **nco_baroc2sql** tool generates is:

```
insert into master.class_membership ( Class, ClassName, Parent ) values (int, 'string', int );
```

Where:

- The Class value contains a unique numeric identifier for the class. The generated class identifiers start from 76000, unless you specify a different start value when running the tool from the command line.
- The ClassName value contains the name of the class as it appears in the BAROC file.
- The Parent value contains the numeric class value of the parent class. If no parent class is defined in the BAROC file, an INSERT statement for the class is created, which has the Parent field set to -1. (These entries are known as root nodes.)

For example:

```
insert into master.class_membership (Class, ClassName, Parent ) values ( 76000, 'ABC_Base', 76001);
```

The master.class_membership table does not permit duplicate mappings of class names to class numbers. The table also does not permit multiple entries with either the same class name or class number.

The **nco_baroc2sql** tool also creates a class conversion entry for each class in the .baroc files. This enables class-specific tools to be written for the Tivoli Netcool/OMNIbus event list. The format of the INSERT statement that the tool generates is:

```
insert into alerts.conversions values ('Class+ClassID', 'Class', ClassID, 'ClassName' );
```

For example:

```
insert into alerts.conversions values ( 'Class76000', 'Class', 76000, 'ABC_Base');
```

Both types of INSERT statements are saved to the same output file.

Note: The **nco_baroc2sql** tool does not perform any validation to check whether the class identifiers that it allocates are available on the target system. The tool also does not put an upper limit on the class identifier values.

The Tivoli Enterprise Console classes are mapped to ObjectServer classes, and 10,000 class identifiers are reserved for this mapping, ranging from 76000 to 86000.

To map incoming Tivoli Enterprise Console events to ObjectServer class identifiers, the **nco_baroc2sql** tool can optionally generate a lookup table file that can be inserted into the rules file of a probe. The lookup table contains the Tivoli Enterprise Console ClassName values mapped to ObjectServer classes. The lookup table has the following format:

```
ClassName ClassID
```

Each class is defined on a separate line, with one definition for each row that is added to the alerts.conversions table by the SQL that the **nco_baroc2sql** tool generates.

Migrating BAROC data (Windows)

Before migrating Tivoli Enterprise Console data, you must prepare a load file that defines the BAROC files to be processed. The BAROC files specified in the load file must be located in the same directory as the load file.

When you run the migration, the **nco_baroc2sql** tool reads the specified load file and processes the BAROC files in the order in which they are presented in the load file.

To run the migration:

1. Enter the following command from the command line:

```
%NCHOME%\omnibus\bin\nco_baroc2sql -baroc baroc_load_file -sql  
output_file -lookup lookup_file
```

Where *baroc_load_file* represents the file path and name of the BAROC load file, *output_file* represents the file path and name of the output file, which is generated as an SQL file, and *lookup_file* optionally represents the file path and name of the lookup table file to which the mapping of ClassName values to ObjectServer classes is written.

The **nco_baroc2sql** command has the following command-line options. Either the **-sql** command-line option or the **-lookup** option must be specified, or both. If neither command-line option is specified, the **nco_baroc2sql** tool fails.

Table 43. Command-line options for the **nco_baroc2sql** command

Command-line option	Description
-baroc <i>file</i>	The path to the BAROC load file, which lists the BAROC files to be processed.
-sql <i>file</i>	The path to which the SQL output file will be written.
-help	Displays help text.
-version	Displays version information about the tool.
-classno <i>int</i>	The base class number to use for class conversions. The default is 76000. Only change this value if you have existing conversions in the range of 76000 to 86000.
-lookup <i>file</i>	Optional: The path to the lookup table file.

Wait for processing to complete.

- Log on to the SQL interactive interface and then import the SQL output file to the ObjectServer as follows:

```
%NCHOME%\omnibus\bin\isql -S servername -U root -password ""<
output_file
```

Where *servername* represents the name of the ObjectServer to which data will be imported, and *output_file* represents the file path and name of the SQL output file.

Results

Processing messages are output to the screen and are not generated to a log file. You can redirect the messages to a log file if required.

The Probe for Tivoli EIF provides event flow integration between Tivoli Enterprise Console and Tivoli Netcool/OMNIBus. Information about this probe is available at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.netcool_OMNIBus.doc/probes/tivoli_eif/tivoli_eif/wip/concept/tveif_intro.html

What to do next

If you need to change the master.class_membership table after you ran the **nco_baroc2sql** tool, proceed as follows:

- To add new entries to the master.class_membership table, determine the highest number for class conversions that the table currently contains. Then, rerun the **nco_baroc2sql** tool and use the **-classno** option to specify a base class number for class conversions that is greater than the currently-highest number.
- To change the mapping of class name to class number, you must delete the existing entries in the master.class_membership table. Then, rerun the **nco_baroc2sql** tool and use the **-classno** option to specify a different base class number to use for class conversions.

You can now reimport the SQL output file into the ObjectServer by repeating step 2 on page 154 of this task.

If you used the `-lookup` command-line option, you can now insert the generated lookup table file into the rules file of the required probe. The following example shows how to define the lookup table `tec_class` in the rules file:

```
table tec_class = "lookup_table"
default = "Unknown"
```

Where `lookup_table` is the path to the lookup table that is generated by the **nco_baroc2sql** tool. The following example shows how to use the `lookup` function to populate the Class element with the Tivoli Enterprise Console class name:

```
$Class = lookup($ClassName,tec_class)
```

Performing postinstallation tasks (Windows)

After installing or upgrading Tivoli Netcool/OMNIBus, you must perform a number of postinstallation tasks and then configure your system.

The Tivoli Netcool/OMNIBus installation or upgrade adds the following shortcuts to the Windows **Start** menu:

- **Start > All Programs > Netcool Conductor**
- **Start > All Programs > Netcool Suite**

The `NCHOME`, `OMNIHOME`, `SYBASE`, and `PATH` environment variables, which are required to run the installed features, are also automatically set or modified. The value of the `SYBASE` variable must not be changed.

Perform one or more of these tasks, depending on the features that you installed:

- Configure the IEHS server for online help access.
- If you want to operate in FIPS 140-2 mode, configure your JRE for FIPS 140-2. Also configure FIPS 140-2 support for the server components.
- Install probes and gateways.
- If you installed the Process Control feature, you can install all process agents to run as Windows services with an automatic startup.

You can optionally install ObjectServers, proxy servers, probes, and gateways as Windows services. Alternatively, you can configure these components to run as Tivoli Netcool/OMNIBus processes within a process control system. For further information about process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

- If you installed the Servers feature, create an ObjectServer.
- If you installed the Gateways, Servers, or Process Control feature, configure server communications.
- If you are creating a distributed installation, see the additional instructions in *Distributed Installations*.

What to do next

To obtain additional support with Tivoli Netcool/OMNIBus and to aid with problem determination, you can also install the IBM Support Assistant.

To use IBM Tivoli Monitoring to monitor and manage Tivoli Netcool/OMNIBus resources, install the IBM Tivoli Monitoring agent for Tivoli Netcool/OMNIBus.

For further information about this agent, see the *IBM Tivoli Monitoring for Tivoli Netcool/OMNIBus Agent User's Guide*.

Related concepts

"Configuring server communication details in the Server Editor" on page 240

Chapter 10, "Configuring FIPS 140-2 support for the server components," on page 295

Related tasks

"Creating an ObjectServer" on page 231

"Setting up distributed installations" on page 248

Related reference

"Configuring the JRE for FIPS 140-2 mode (Windows)" on page 158

Appendix A, "IBM Support Assistant," on page 605

Configuring settings for online help access (Windows)

After installing Tivoli Netcool/OMNIBus, you might need to configure your system for online help access. Online help is deployed using IBM Eclipse Help System (IEHS) and can be accessed in standalone mode or information center mode.

Configure the online help settings in the following table, as relevant for your system.

Table 44. Help configuration settings

Setting	Description
Standalone mode	Configuration is only necessary if the default IEHS port number of 8888 is being used by another local service.
Local Help System feature installed on a client workstation (Online help files installed on a local IEHS Web server)	<p>If this is the case, edit the IEHS configuration file %NCHOME%\omnibus\ini\nco_IEHS.cfg as follows:</p> <ul style="list-style-type: none">• IEHSMODE: 0• IEHSHost: <i>leave blank</i>• IEHSPort: <i>an unused port number</i> <p>Note: After changing the port number in the configuration file, you must shut down the local IEHS server for your changes to take effect. Run the %NCHOME%\omnibus\bin\help_end.bat command or double-click the file in Windows Explorer. (The server automatically restarts when you access online help.)</p>

Table 44. Help configuration settings (continued)

Setting	Description
Information center mode	On the computer designated as the IEHS server, edit the IEHS configuration file %NCHOME%\omnibus\ini\nco_IEHS.cfg as follows:
Local Help System feature installed on a remote server	<ul style="list-style-type: none"> • IEHSMode: 1 • IEHSHost: <i>leave blank, or specify the IP address or host name of the IEHS server</i> <p>Note: IEHS V3.1.1 does not support IPv6 addresses.</p> <ul style="list-style-type: none"> • IEHSPort: <i>an unused port number for the IEHS server</i> <p>The default port number is 8888. If necessary, update your firewall settings to open the port.</p>
(Online help files installed on a remote IEHS Web server, which is configured for client access; typically managed by a system administrator)	<p>The host name on which the IEHS server is running can be obtained from the %NCHOME%\omnibus\platform\win32\nco_IEHS\eclipse\workspace\.metadata\.connection file. Note that this file is available only when the IEHS server is running. The file is deleted when you shut down the IEHS server.</p> <p>On each client workstation, edit the IEHS configuration file %NCHOME%\omnibus\ini\nco_IEHS.cfg as follows:</p> <ul style="list-style-type: none"> • IEHSMode: 1 • IEHSHost: <i>IP address or host name of the IEHS server</i> • IEHSPort: <i>port number on which the IEHS server is running</i> <p>Important: You must instruct users to perform this task.</p>

Related concepts

“Online help requirements” on page 16

Running the IEHS server (Windows)

In information center mode, you must manually start the IEHS server by using the **IC_start.bat** command. In standalone mode, the local IEHS server starts automatically.

If you have configured your help system to use the information center mode, you must start the IEHS server to make it available to users who need to access online help. To start the IEHS server on the configured computer, enter the following command at the command line:

```
%NCHOME%\omnibus\bin\IC_start.bat
```

To stop the IEHS server, enter the following command at the command line:

```
%NCHOME%\omnibus\bin\IC_end.bat
```

In standalone mode, the local IEHS server automatically starts the first time that you make a help request. The local IEHS server continues to run until you stop it by using the following command:

```
%NCHOME%\omnibus\bin\help_end.bat
```

Configuring the JRE for FIPS 140–2 mode (Windows)

To configure the Tivoli Netcool/OMNIBus JRE for FIPS 140–2 operation, change the configuration of the security properties file. You can also download and add policy files to use enhanced encryption algorithms.

Configuration file changes

Make the following configuration changes to the security properties file:

1. Open the %NCHOME%\platform\win32\jre_1.6.7\jre\lib\security\java.security file for editing.
2. Edit the file as follows:
 - In the List of providers and their preference orders section, add the following lines:
security.provider.1=com.ibm.fips.jsse.IBMJSSEFIPSPProvider and
security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS. For all other providers, increment the number by two, as shown in the following table, for your operating system:

security.provider.1=com.ibm.fips.jsse.IBMJSSEFIPSPProvider
security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.3=com.ibm.jsse2.IBMJSSEProvider2
security.provider.4=com.ibm.crypto.provider.IBMJCE
security.provider.5=com.ibm.security.jgss.IBMJGSSProvider
security.provider.6=com.ibm.security.cert.IBMCertPath
security.provider.7=com.ibm.security.sasl.IBMSASL
security.provider.8=com.ibm.xml.crypto.IBMXMLCryptoProvider
security.provider.9=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.10=org.apache.harmony.security.provider.PolicyProvider
security.provider.11=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.12=com.ibm.security.cmskeystore.CMSProvider
 - Set the default key and trust manager factory algorithms for the javax.net.ssl package:

ssl.KeyManagerFactory.algorithm=IbmX509
ssl.TrustManagerFactory.algorithm=IbmX509
 - Set the default SSLSocketFactory and SSLServerSocketFactory provider implementations for the javax.net.ssl package:

ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
3. Save and close the file.

Enhanced encryption algorithms

To enable strong encryption, you need to download and install policy files that allow this feature, from IBM developerWorks. This involves acceptance of licensing terms.

The steps to enable strong encryption are as follows:

1. Go to the developerWorks Java Technology Security Web page at <http://www-106.ibm.com/developerworks/java/jdk/security/>.
2. Click the **Java SE 6** link. (The files are the same for JRE 1.5.n.)
3. Scroll down on the resulting page and click the **IBM SDK Policy files** link.
4. If you already have an IBM ID and password, click the **Sign in** link. Otherwise, click the **Register here** link to create an ID.
5. On the "Sign in" page, supply your IBM ID and password.
This takes you to the "Unrestricted JCE policy files for SDK 1.4" page.

6. Select **Unrestricted JCE Policy files for SDK for all newer versions** and click **Continue**.
7. Scroll down to the License section of the resulting page and click the **View license** link to see the licensing terms for the download.
8. If the licensing terms are acceptable, select **I agree** and click the **I confirm** link. If the terms are not acceptable, you will not be able to enable strong encryption and should click **I cancel**.
9. Click the **Download now** link to download the unrestricted.zip file.
10. Extract the local_policy.jar and US_export_policy.jar files from the unrestricted.zip archive.
11. Save these two files to the %NCHOME%\platform\win32\jre_1.6.7\jre\lib\security directory, replacing the existing files of the same names.
12. Update the policy files on each computer, and optionally run tests.

Related reference

“Switching your installation to FIPS 140-2 mode” on page 299

Installing probes and gateways into the Tivoli Netcool/OMNIbus environment (Windows)

Probes and gateways are part of the Tivoli Netcool/OMNIbus suite, and are available as download packages on the Passport Advantage Online Web site.

You can install probes and gateways into a new Tivoli Netcool/OMNIbus environment, or upgrade probes and gateways after upgrading Tivoli Netcool/OMNIbus.

Probes are generally installed on a separate workstation from the ObjectServer.

Note: As of Tivoli Netcool/OMNIbus V7.0, probes are no longer installed as Windows services. However, it is possible to install and run a probe as a service.

Tip: Probes can be deployed to remote computers by using the remote deployment mechanism provided by IBM Tivoli Monitoring.

Gateways are generally installed on either the primary server or other servers. You can install ObjectServer gateways as part of the Tivoli Netcool/OMNIbus installation. Other gateways are installed separately by using the download package for individual gateways.

Related concepts

“Deploying probes remotely” on page 451

Installing probes or gateways into a new Tivoli Netcool/OMNIbus environment (Windows)

For a new installation of Tivoli Netcool/OMNIbus V7.3.1, download and install each probe and gateway that you require. You can run the installer as a wizard, or in console or silent mode, in a similar manner used for installing Tivoli Netcool/OMNIbus.

Attention: *Download and use only repackaged or new probes and gateways in your V7.3.1 installation.* In V7.3.1, probes and gateways are installed using the **nco_install_integration** command. Earlier probe and gateway packages, which have not been repackaged, cannot be installed into your V7.3.1 installation by using the **nco_install_integration** command.

Before you begin

The computer on which you install the probe must have the Tivoli Netcool/OMNIbus **Probe Support** feature installed prior to probe installation. Any feature can be installed to obtain the infrastructure required for gateways.

Proceed as follows:

1. To download probes from the Passport Advantage Online Web site, follow the instructions that are available in the Tivoli Netcool/OMNIbus Information Center at:
http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.tivoli.nam.doc/welcome_ptsm.htm
2. To download gateways from the Passport Advantage Online Web site, follow the instructions that are available in the Tivoli Netcool/OMNIbus Information Center at:
http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.tivoli.nam.doc/welcome_og.htm
3. After downloading the Windows installation package for a probe or gateway, extract the contents of the package to a temporary location.
4. Install the probe or gateway by running the following command:
`%NCHOME%\omnibus\install\nci_install_integration option`
The value of *option* depends on your installation mode as follows:

Installation mode	Instruction
Installation wizard	No value is required for <i>option</i> . When the wizard runs, follow the prompts to: <ol style="list-style-type: none">1. Specify the location of the probe or gateway to be installed. This location is the directory containing the README.txt file in the extracted package.2. Accept the license conditions.
Console mode	Specify <i>option</i> as: <code>-i console</code> When the text-based installer runs, follow the prompts to: <ol style="list-style-type: none">1. Specify the location of the probe or gateway to be installed. This location is the directory containing the README.txt file in the extracted package.2. Accept the license conditions.
Silent mode	Specify <i>option</i> as: <code>-i silent -f full_path\response.txt</code> Where: <ul style="list-style-type: none">• <i>full_path</i> specifies the full path to a response file named response.txt that you are required to create.• response.txt is a text file that you create with the following contents: <code>LICENSE_ACCEPTED=true</code> <code>PROBE_OR_GATE_LOCATION=README_directorypath</code> <i>README_directorypath</i> is the path to the directory that contains the README.txt file.

Results

On completion:

- Probes are installed to the following directory:
%NCHOME%\omnibus\probes\win32
- Gateways are installed to the following directory:
%NCHOME%\omnibus\bin: Gateway binaries
%NCHOME%\omnibus\gates: Gateway configuration files

Installing probes or gateways into an upgraded Tivoli Netcool/OMNIbus environment (Windows)

If you have upgraded your of Tivoli Netcool/OMNIbus V7.3.1 from a version that is earlier than V7.3, reinstall all your probes and gateways and then import the old probe and gateway configuration data. You can run the installer as a wizard, or in console or silent mode, in a similar manner used for installing Tivoli Netcool/OMNIbus. If you have upgraded from Tivoli Netcool/OMNIbus V7.3, you do not have to perform this task.

To install probes or gateways and import existing configuration data:

1. Follow the instructions for installing probes and gateways into a new Tivoli Netcool/OMNIbus environment.
2. After installing, import the configuration data. The upgrade script, which migrated your old Tivoli Netcool/OMNIbus data into the new V7.3.1 installation, would have copied your old probe and gateway configuration files into the following locations:
 - Probe configuration files: %NCHOME%\omnibus\probes\migrated
 - Gateway configuration files: %NCHOME%\omnibus\etc
3. Copy the migrated probe configuration files in the %NCHOME%\omnibus\probes\migrated directory into the appropriate locations in %NCHOME%\omnibus\probes\win32.

Related tasks

“Installing probes or gateways into a new Tivoli Netcool/OMNIbus environment (Windows)” on page 159

Setting up Tivoli Netcool/OMNIbus components as Windows services

The Tivoli Netcool/OMNIbus server components and probes can be installed to run as services on a Windows host. The server components that you can install as services include the ObjectServer, process agent, proxy server, and gateways.

You can install, configure, and uninstall services by running a command-line utility. You can optionally configure installed services from the Services window in the Control Panel.

As is standard practice on Windows, you can configure your Tivoli Netcool/OMNIbus services with automatic startup.

You can alternatively configure Tivoli Netcool/OMNIbus components to run as Tivoli Netcool/OMNIbus processes within a process control system. If you plan to use process control, the preferred approach is to install your process agents as

Windows services, and then set up the other components to run as processes within the process control system. For further information about process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Installing, configuring, and uninstalling services for server components

To install, configure, or uninstall the Tivoli Netcool/OMNIBus server components as services, you must run the executable file for the component with one or more additional command-line options.

To install, configure, or uninstall a Tivoli Netcool/OMNIBus service:

1. From a command prompt, enter the following command:

```
%NCHOME%\omnibus\bin\nco_name [option [value]...]
```

In this command:

- **nco_name** represents any of the executable names that are shown in the following table.

Table 45. Executable names for server components

Component type	Executable name
ObjectServer	nco_objserv
Proxy server	nco_proxyserv
Process agent	nco_pad
Gateway	nco_g_gatewayname <i>gatewayname</i> is the abbreviated name of a gateway type.

- Square brackets represent optional entries for the command-line options that you can use with the **nco_name** command. The following table lists each *option* and its *value* (if required).

Table 46. Command-line options for installing, configuring, and uninstalling Tivoli Netcool/OMNIBus services

Command-line option	Function
/INSTALL	Installs a Tivoli Netcool/OMNIBus server component or probe as a service.
/REMOVE	Removes an installed service.
/NOAUTO	Installs the service with manual startup. Omit this option to run the service with automatic startup.

Table 46. Command-line options for installing, configuring, and uninstalling Tivoli Netcool/OMNibus services (continued)

Command-line option	Function
/DEPEND <i>srv</i> @ <i>grp</i> ...	<p>Specifies other services or groups on which the service being installed is dependent. If you use this option, this service does not start until the services (<i>srv</i>) and groups (@ <i>grp</i>) that you specify with this option have run.</p> <p>Note: The value of <i>srv</i> must be the Service name, not the Display name. To view the Service name, open the Windows Control Panel, and then double-click Administrative Tools and Services in succession. Double-click the relevant service entry to open the Properties window. The Service name is shown on the General tab of the Properties window.</p>
/GROUP <i>name</i>	<p>Installs the service as a member of a group, where <i>name</i> represents the group name. This option is used in conjunction with the /DEPEND command-line option.</p> <p>For example, you can group all probes together under the same group name. You can then force that group to be dependent on another service.</p>
/ACCOUNT [<i>domain</i> \] <i>user</i>	<p>Specifies that the service logs on to a user account. In this command syntax, <i>domain</i> represents the domain name and is optional (as depicted by the square brackets), and <i>user</i> represents the user name. If using the /ACCOUNT command-line option, you must also specify the /PASSWORD option.</p> <p>If you omit the /ACCOUNT command-line option, the service logs on to the local system account.</p> <p>For example, if you want to install a process agent service as a local workstation user (that is, Administrator), <i>domain</i> is not needed. Enter:</p> <pre>nco_pad /INSTALL /ACCOUNT Administrator</pre> <p>Tip: Be aware that the account requires "logon as service" rights, which is automatically granted when you specify a logon account for the service from the Services window in the Control Panel. This is not the case if you use the /ACCOUNT option with the /INSTALL option when installing the service from the command line.</p>
/PASSWORD <i>password</i>	Specifies a <i>password</i> string for the user account.
/INSTANCE <i>ID</i>	Specifies a unique instance identifier for a service, where <i>ID</i> represents the identifier. For example, if installing more than one process agent service, the second and subsequent services each require a unique ID.

Table 46. Command-line options for installing, configuring, and uninstalling Tivoli Netcool/OMNibus services (continued)

Command-line option	Function
/CMDLINE " <i>option</i> "	<p>Specifies one or more command-line options to be set whenever the service is restarted. Ensure that the command-line options are enclosed within double quotation marks.</p> <p>Use command-line options that are available for the type of component being configured. For example, for the ObjectServer, specify one or more command-line options that can be used with the nco_objserv command, or for the process agent, specify one or more command-line options that can be used with the nco_pad command.</p> <p>Examples:</p> <ul style="list-style-type: none"> To specify which process agent should run when a service for the NCO_PA process agent starts, set the value of <i>option</i> to "-name NCO_PA". To specify an alternative log file /tmp/my_pafile.log to which messages are written, set the value of <i>option</i> to "-logfile /tmp/my_pafile.log".
/BACKOFF <i>n</i>	<p>Defines the maximum number of startup or connection attempts of the service, where <i>n</i> is an integer representing this number.</p> <p>For example, to specify the maximum number of times a probe should attempt to connect to the NCOMS ObjectServer, include the following options when installing the probe service:</p> <p>/CMDLINE "-server NCOMS" /BACKOFF 3</p>

Tip: You can view the command-line options for installing, configuring, and uninstalling these Windows services by using the following command:
 %NCHOME%\omnibus\bin\nco_name.exe /?

2. After installing services, reboot the computer.

What to do next

After you install a server component as a service, you must use the Server Editor to set the host and port number for the service before starting it.

Also use the Services window in the Control Panel to assign either of the following logon accounts to the service:

- Local system account (LocalSystem). This is the default, and preferred, option. This account does not have a password.
- An account that belongs to the Administrators group on the local computer. With this option, it is advisable to use accounts with passwords that do not expire.

Related concepts

"Configuring server communication details in the Server Editor" on page 240

Installing, configuring, and uninstalling probe services

To install, configure, or uninstall probes as services, see the `description.txt` file that is available with the download package for each probe.

Viewing and reconfiguring installed services

You can view and reconfigure the Windows services that you have installed for Tivoli Netcool/OMNIbus server components and probes.

To view and reconfigure installed services:

1. Open the Windows Control Panel.
2. Double-click **Administrative Tools** and then double-click **Services**. The Services window opens with a list of all Windows services that are currently installed on your computer. The display and service names of all Tivoli Netcool/OMNIbus services start with either Netcool or NCO, as shown in the following table.

Table 47. Display and service names

Component type	Display name	Service name
ObjectServer	Netcool/OMNIbus Object Server Note: If more than one ObjectServer service is installed, the second and subsequent services are displayed as: Netcool/OMNIbus Object Server (ID) . In this case, <i>ID</i> is the identifier specified by the <code>/INSTANCE</code> command-line option when installing the service.	NCOObjectServer Additional service names have the format: NCOObjectServer\$ID Where <i>ID</i> is the identifier specified by the <code>/INSTANCE</code> command-line option when installing the service.
Process agent	NCO Process Agent	NCOProcessAgent
Proxy server	NCO Proxy Server	NCOProxyServer
ObjectServer Gateway (unidirectional)	Netcool/OMNIbus Uni-Directional ObjectServer Gateway	NCOObjectServerGatewayUni
ObjectServer Gateway (bidirectional)	Netcool/OMNIbus Bi-Directional ObjectServer Gateway	NCOObjectServerGatewayBi
Probe	NCO Name Probe Where <i>Name</i> is the unique abbreviated probe name. For example: NCO SimNet Probe	NCONameProbe

3. Use the Services window to start and stop the services as relevant.
4. Use the Properties window for each service to configure the following properties:
 - The startup type for the service. This can be Automatic, Manual, or Disabled.
 - The logon account for the service
 - The recovery action to take if the service fails

Results

Note: If an ObjectServer and a probe are started as services, the probe might start first, but cannot connect to the ObjectServer until the ObjectServer is running.

Examples: Setting up components as services

These examples show how to install, run, and uninstall Tivoli Netcool/OMNIbus components as Windows services.

Example: Installing and running the process agent as a service:

This example shows how to install, run, and uninstall the process agent as a Windows service.

To install the process agent as a Windows service:

1. Run the Server Editor to set connection details for the process agent.
2. Run the following commands in succession to open a command prompt window and install the process agent service:

```
cmd.exe
cd %NCHOME%\omnibus\bin
nco_pad /install
```

The service is installed with a Service name of **NCOProcessAgent**.

Another example command for installing the process agent service is as follows:

```
nco_pad /install /noauto /cmdline "-secure -debug 1 -cryptalgorithm
AES_FIPS -keyfile \"%OMNIHOME%\bin/keyfile1\""
```

In this example, the service is being installed with manual startup. The process agent is set to run in secure mode, and will log information about processes it is about to start, to its log file. The algorithm and key file, which can be used to decrypt passwords in the process agent configuration file, are also specified. Such password decryption is required if your configuration file contains encrypted passwords that were generated with the **nco_aes_crypt** command. Note that you must escape embedded quotation marks with backslashes, as shown with the **-keyfile** value.

To start the process agent service, perform either of the following actions:

- From the command line, run the following command:
`net start NCOProcessAgent`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus process agent service and then start this service. You can also set the startup type of the service to Automatic.

To stop the process agent service, perform either of the following actions:

- From the command line, run the following command:
`net stop NCOProcessAgent`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus process agent service and then stop this service.

To uninstall the service, run the following command:

```
nco_pad /remove
```

Example: Installing, running, and uninstalling an ObjectServer called MYSERV as a Windows service:

This example shows how to install, run, and uninstall an ObjectServer called MYSERV as a Windows service.

To install an ObjectServer called MYSERV as a service:

1. Run the Server Editor to set connection details for the new server.
2. Run the following commands in succession to open a command prompt window, initialize the database, and install the ObjectServer service:

```
cmd.exe
cd %NCHOME%\omnibus\bin
nco_dbinit -server MYSERV
nco_objserv /install /cmdline "-name MYSERV"
```

The service is installed with a Service name of **NCOObjectServer**.

To start the ObjectServer service, perform either of the following actions:

- From the command line, run the following command:
`net start NCOObjectServer`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus ObjectServer service and then start this service.

To stop the ObjectServer service, perform either of the following actions:

- From the command line, run the following command:
`net stop NCOObjectServer`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus ObjectServer service and then stop this service.

To uninstall the service, run the following command:

```
nco_objserv /remove
```

Related reference

"Example: Installing, running, and uninstalling a second ObjectServer called OSTWO on the same host"

Example: Installing, running, and uninstalling a second ObjectServer called OSTWO on the same host:

This example shows how to install, run, and uninstall a Windows service for a second ObjectServer called OSTWO, on the same host as the first ObjectServer.

To install a second ObjectServer called OSTWO as a service:

1. Run the Server Editor to set connection details for the new server.
2. Run the following commands in succession to open a command prompt window, initialize the database, and install the ObjectServer service:

```
cmd.exe
cd %NCHOME%\omnibus\bin
nco_dbinit -server OSTWO
nco_objserv /install /instance TWO /cmdline "-name OSTWO"
```

The service is installed with a Service name of **NCOObjectServer\$TWO**.

To start the ObjectServer service, perform either of the following actions:

- From the command line, run the following command:
`net start NCOObjectServer$TWO`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus ObjectServer service and then start this service.

To stop the ObjectServer service, perform either of the following actions:

- From the command line, run the following command:
`net stop NCOObjectServer$TWO`
- From the Windows Control Panel, double-click **Administrative Tools** and then **Services**. Locate the Tivoli Netcool/OMNIbus ObjectServer service and then stop this service.

To uninstall the service, run the following command:

```
nco_objserv /remove /instance TWO
```

Related reference

“Example: Installing, running, and uninstalling an ObjectServer called MYSERV as a Windows service” on page 167

Example: Installing and running the proxy server as a service:

This example shows how to install and run the proxy server as a Windows service.

To install and run the proxy server service, run the following commands in succession:

```
cmd.exe  
  
cd %NCHOME%\omnibus\bin  
  
nco_proxyserv /install  
  
net start NCOProxyServer
```

Example: Installing and running ObjectServer gateways as services:

This example shows how to install and run ObjectServer gateways as Windows services.

To install and run a bidirectional gateway called BI_GATE as a service, run the following commands in succession:

```
cmd.exe  
  
cd %NCHOME%\omnibus\bin  
  
nco_g_objserv_bi /install /cmdline "-name BI_GATE"  
  
net start NCOObjectServerGatewayBi
```

To install and run a unidirectional gateway called UNI_GATE as a service, run the following commands in succession:

```
cmd.exe

cd %NCHOME%\omnibus\bin

nco_g_objserv_uni /install /cmdline "-name UNI_GATE"

net start NCOObjectServerGatewayUni
```

Example: Installing services with dependencies:

This example shows how to install the core Tivoli Netcool/OMNIBus components as services in a group called OmniCore, and then install a bidirectional gateway that is dependent on this group.

Run the following commands in succession:

```
cmd.exe

cd %NCHOME%\omnibus\bin

nco_objserv /install /group OmniCore

nco_pad /install /group OmniCore

nco_proxyserv /install /group OmniCore

nco_g_objserv_bi /install /depend @OmniCore
```

Uninstalling Tivoli Netcool/OMNIBus (Windows)

You can uninstall Tivoli Netcool/OMNIBus by using the installation wizard, or the console or silent installation mode.

The installer records the mode that was used for the installation. When the **uninstall** command is invoked, the mode used for installing is, by default, used for uninstalling. The **uninstall** command provides command-line options that you can use to set the uninstallation mode irrespective of the mode used at installation time.

When you uninstall Tivoli Netcool/OMNIBus, the uninstallation process removes all files except for the following files:

- Files used by the installer program, such as the installer plan and log files, and the installer database files
- Common packages that are required by other products installed in the same NCHOME location
- Tivoli Netcool/OMNIBus configuration files that have been modified
- Probes and non-ObjectServer gateways - these have their own uninstaller

Note: Deployment Engine files are retained only if they are still required by another product on the same server.

To completely remove Tivoli Netcool/OMNIBus and any other related Tivoli products from the %NCHOME% location:

1. Uninstall Tivoli Netcool/OMNIBus (and the other products).
2. Manually delete the following directories and files:
 - %NCHOME% and its remaining subdirectories
 - C:\Program Files\IBM\Common\acsi (if present)
 - C:\Documents and Settings\username\coi
 - C:\Documents and Settings\username\IA-Netcool-OMNIBus-hostname-yyyymmddThhmmss-0.log and any other IA-*.log files in the same location

Where *username* is the name of the logged-in user who performed the installation, and *nn* is a two-digit number typically starting from 00.

Before you uninstall

Before uninstalling Tivoli Netcool/OMNIBus, you must remove any Tivoli Netcool/OMNIBus services that were manually installed.

Before you begin

Remember to stop the service before attempting to remove it.

You can remove a probe service by running the executable file for the probe with the -remove command-line option, as described in the description.txt file that accompanies each probe download on the IBM Support Site.

You can remove other Tivoli Netcool/OMNIBus services by running the relevant executable file with the /REMOVE command-line option. If you installed the service with an instance identifier, you must also include the /INSTANCE command-line option.

Related tasks

"Installing, configuring, and uninstalling services for server components" on page 162

Uninstalling using the wizard (Windows)

You can use the uninstall wizard to guide you through the uninstallation process for Tivoli Netcool/OMNIBus.

To uninstall Tivoli Netcool/OMNIBus by using the wizard:

1. Stop all processes that are currently running.
2. From a command prompt, change to the following directory:
%NCHOME%_uninst\OMNIBus
3. Enter the following command:
uninstall.exe -i gui

Note: You can also click **Start > All Programs > NETCOOL Suite > Uninstall OMNIBus** to uninstall. The mode in which the product was installed will, however, be used by default for uninstalling.

The Uninstall Wizard starts and displays the Uninstall OMNIBus page.

4. Click **Uninstall** to proceed with the uninstallation, and wait while the features are removed. On completion, the wizard confirms that Tivoli Netcool/OMNIBus was successfully uninstalled.

5. Click **Done** to close the wizard.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling in console mode (Windows)

Use the console mode to uninstall Tivoli Netcool/OMNIbus from the command-line interface.

To uninstall Tivoli Netcool/OMNIbus in console mode:

1. Stop all processes that are currently running.
2. From a command prompt, change to the following directory:
`%NCHOME%_uninst\OMNIbus`
3. Enter the following command:
`uninstall.exe -i console`
A new command window opens from which you can perform the uninstallation.
4. At the prompt, press Enter to proceed. A confirmation message is shown briefly, and the command window closes.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling in silent mode (Windows)

Use the silent mode to uninstall Tivoli Netcool/OMNIbus with no user interaction.

To uninstall Tivoli Netcool/OMNIbus in silent mode:

1. Stop all processes that are currently running.
2. From a command prompt, change to the following directory:
`%NCHOME%_uninst\OMNIbus`
3. Enter the following command:
`uninstall.exe -i silent`
You are returned to the command prompt. Wait for the uninstallation to complete.

What to do next

If the uninstallation failed, you might need to remove the Deployment Engine (DE) before you can reattempt the uninstallation process.

Related tasks

“Uninstalling the Deployment Engine” on page 621

Related reference

“Installation error messages” on page 613

Uninstalling probes and gateways (Windows)

Probes and non-ObjectServer gateways are separate packages and are not removed from your system when you remove Tivoli Netcool/OMNIbus. You must uninstall probes and gateways individually.

To uninstall a probe or gateway after uninstalling Tivoli Netcool/OMNIbus:

1. From a command prompt, change to the following directory:

`%NCHOME%_uninst\name`

Where *name* is a subdirectory named after the probe or gateway.

2. Enter the following command:

`uninstall.exe`

The uninstaller runs in the mode in which the probe or gateway was installed.

Chapter 6. Installing, upgrading, and uninstalling the Web GUI component

Read how to install, upgrade, and uninstall the Web GUI component. The installation, upgrade, and uninstallation process are identical for all operating systems.

Related concepts

Chapter 4, “Installing, upgrading, and uninstalling (UNIX and Linux),” on page 45

Chapter 5, “Installing, upgrading, and uninstalling (Windows),” on page 111

Preparing to install or upgrade the Web GUI

Before you install or upgrade the Web GUI, you might need to perform one or more preinstallation tasks, depending on the features that you want to install. You also need to obtain the installation package for your operating system.

Types of Installation

There are two types of installation available with the Web GUI component; default or advanced.

The default installation uses the ObjectServer as a user registry and the default context root for the Tivoli Integrated Portal.

The advanced installation allows the user to specify:

- The type of user registry: ObjectServer or file-based
- The default context root (/ibm/console)

In both types of installation you can choose to create a new directory to use or an existing Tivoli Integrated Portal installation. In addition, you define the ObjectServer to which the Web GUI connects and, optionally, a secondary ObjectServer for failover protection.

Note: The Web GUI can use other types of user registry, such as an LDAP repository. To use such a registry, you carry out an advanced installation of the Web GUI specifying a file-based registry. After the installation has completed, you perform the configuration for LDAP against the Tivoli Integrated Portal installation.

Related information



Managing the realm in a federated repository configuration

Web GUI installation or upgrade prerequisites

If you are installing or upgrading the Web GUI, you must take note of a number of prerequisites.

These prerequisites are as follows:

- Sufficient disk space must be available on the volume where you want to install the Web GUI. If you intend to install other Network Management products, the installation location must also have sufficient space to accommodate these installations.
- IBM Tivoli Netcool/OMNIbus is installed.
- An ObjectServer is installed and running.
- **Windows** Ensure that the account you intend to use for the installation has administrator privileges.

Related concepts

“Disk space requirements” on page 17

Obtaining the installation package

The Tivoli Netcool/OMNIbus Web GUI is distributed as a compressed file that is available on DVD, or that you can download from the IBM Passport Advantage Online Web site.

The boxed product package contains the DVD that you can use to install the Tivoli Netcool/OMNIbus Web GUI on your operating system.

If you are downloading the Web GUI installation package, proceed as follows:

1. Follow the instructions in the download document for your operating system:

Operating system	Download document location
AIX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026995
HP-UX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026996
HP-UX Integrity	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026999
Linux	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026997
Linux for System z	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026998
Solaris	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027000
Windows	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027001

2. Extract the contents of the installation package into a temporary location.

Important: When extracting the installation package be sure to use a tool that can handle very long file paths.

Tip: To run the Web GUI installer from the Tivoli Netcool/OMNIbus launchpad, you must create a directory called WebGUI, either as a parent

directory or subdirectory of the directory into which you extracted the Tivoli Netcool/OMNIBus installation package. Then, extract the Web GUI installation package into the WebGUI directory.

What to do next

Complete any of the other preinstallation tasks required for your installation or upgrade. After you have completed these tasks, you can run the installation program to perform a new installation of the Web GUI, or an installation that upgrades a previous version of IBM Tivoli Netcool/Webtop.

Related tasks

“Starting the installation launchpad” on page 178

“Upgrading from IBM Tivoli Netcool/Webtop version 2.2 or Tivoli Netcool/OMNIBus version 7.3.0 Web GUI to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI” on page 192

Gathering installation information

Before installing the Web GUI, gather the necessary information.

There are two sets of information that you may need to gather. First there is a collection of values and settings that apply to all installations of the Web GUI. This includes items such as the credentials of the administrative user and the characteristics of the ObjectServer that the Web GUI connects to. This set of information provides all you need to carry out a basic installation of the product (this is called a default installation when using the GUI installer).

There are an additional set of values that you can specify for an advanced installation. These enable you to specify whether to use features such as the context root of the Tivoli Integrated Portal. Gather these items only if your installation requires them

Tip: If you want to use LDAP as a user repository, carry out an advanced installation and specify a file-based user repository. You set up the LDAP repository when configuring the Web GUI after completing the installation.

Information for all installations

Gather the following information for all installations of the Web GUI:

Table 48. Information for a default installation

Item	Default value	Description
Deployment Engine access policy		

Table 48. Information for a default installation (continued)

Item	Default value	Description
Access policy	Do not change	<p>This panel is displayed only when you install as root.</p> <p>By default, the Deployment Engine (DE) that the installer uses allows write access to all users of the system. Decide on the access policy you require:</p> <ul style="list-style-type: none">• Leave the policy unchanged• Allow access to the current (root) user only• Allow access to a specified user group and the current (root) user only <p>Decide on the identity of the an existing user group on the operating system to use.</p> <ul style="list-style-type: none">• Allow access to all users
Installation directory of the Tivoli Integrated Portal		
Create a new instance of the Tivoli Integrated Portal or reuse an existing one?	Create	Specifies whether you want to create a new instance of the Tivoli Integrated Portal for the Web GUI or reuse an existing one.
Directory	/ibm/tivoli/tipv2	<p>When creating a new instance of the Tivoli Integrated Portal, decide on the installation directory to use for the Web GUI and the Tivoli Integrated Portal.</p> <div><div>UNIX</div><div>Linux</div><div>The path name cannot contain any spaces.</div></div>
Tivoli Integrated Portal instance		When reusing a Tivoli Integrated Portal instance, the directory containing the instance to use.
Installation directory for the Web GUI		
Directory	/ibm/tivoli/netcool/omnibus_webgui	<p>Decide on a directory to hold the Web GUI. This is also called the product home directory.</p> <div><div>UNIX</div><div>Linux</div><div>The path name cannot contain any spaces.</div></div>
Administrative user		
User ID	tipadmin	The login credentials of the Web GUI administrative user.
Password		Restriction: The password for the administrative user cannot begin with a hyphen (-).
Communication ports		

Table 48. Information for a default installation (continued)

Item	Default value	Description
Web GUI port	16310	<p>The nonsecured port that the Web GUI uses to listen for connection requests from users. The installation procedure also reserves the port one greater than this (by default 16311) for secured connections.</p> <p>Make sure that both ports are currently not used on the server that the Web GUI is to use.</p>
Primary ObjectServer characteristics		
User ID	root	The identity and credentials of the primary ObjectServer in a dual ObjectServer configuration or the only ObjectServer that provides Web GUI with data.
Password	*****	
Name	NCOMS	
Host name	myobjectserver.ibm.com	
Port	4100	
Secondary ObjectServer characteristics		
Enable secondary server for failover?	No	Determines whether your site uses a secondary ObjectServer for failover protection. If your site has a secondary ObjectServer gather the identity and credentials of that server.
Name		
Host name		
Port		

Information for advanced installations

To carry out an advanced installation you must create a new instance of the Tivoli Integrated Portal for this installation. In addition, gather one or more of the following sets of values as your site needs. When installing into an existing instance of the Tivoli Integrated Portal these items are already defined and in use so you cannot change them.

Table 49. Information for an advanced installation

Item	Default value	Description
Type of user repository	ObjectServer	Decide whether to use the ObjectServer as a user repository or a local, file-based repository.
Context root of the Tivoli Integrated Portal	/ibm/console	The context root determines the URL that users supply to access the Tivoli Integrated Portal and hence the Web GUI.

Starting the installation launchpad

After you have downloaded the installation package, you can optionally start the installation launchpad, from which you select the Web GUI component for installation. You can use the launchpad only to install Web GUI using the GUI installer.

Before you begin

You must have downloaded the Web GUI installation package into a directory called WebGUI that is either a subdirectory or parent directory of the Tivoli Netcool/OMNIBus installation package.

To start the launchpad:

1. In the location where you extracted the installation media, change to the `cdimage` directory and run the executable for your operating system:
 - **UNIX** **Linux** `launchpad.sh`
 - **Windows** `launchpad.bat`
2. Click **Install Product**.
3. Click **Start Web GUI Installation**.

Results

The GUI installer for the Web GUI starts.

Related tasks

“Using the GUI installer”

Installing the Web GUI

Use any of three ways to install the Web GUI.

Using the GUI installer

The GUI installer provides a structured sequence of windows to guide you through the installation process. The installer provides two ways of installing the Tivoli Netcool/OMNIBus Web GUI: default and advanced.

Before you begin

Before you begin the installation, carry out the following steps:

1. **AIX** If a non-root user is to perform the installation, carry out one of the following actions as a root user:
 - Make sure that the user has access to the **slibclean** command and that they run the command before they begin the installation. The executable file for this command is typically in `/usr/sbin/slibclean`.
 - Run the **slibclean** command before the user begins the installation.
2. If you intend to install the Web GUI into an existing instance of the Tivoli Integrated Portal, stop that instance.

For instructions on how to stop a Tivoli Integrated Portal server, refer to the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.
3. Log in as the user for the installation. That user must have write permission to the directory where you are to install Tivoli Netcool/OMNIBus Web GUI.

UNIX

Linux

Always log in as the preferred user for the installation. Do not log in as root and then switch to the preferred user.

4. Have to hand the installation information you gathered during the preparation.

Use this procedure to install the Tivoli Netcool/OMNIBus Web GUI using the GUI installer on each computer where you want to install the product.

1. Change to the `cdimage` directory of the DVD or downloaded installation image and run the installer for your operating system:

- **Windows** `install.exe`
- **UNIX** **Linux** `./install.sh`

2. Select the language for the installation and click **OK**.
3. On the Introduction screen, click **Next**.
4. Read the license agreement, select **I accept the terms in the license agreement** and click **Next**.

The installer adds the Deployment Engine if it is not installed already.

5. Choose the access policy you wish to apply to the Deployment Engine and click **Next**.
6. Choose whether you want to install the Web GUI in a new instance of the Tivoli Integrated Portal or reuse an existing instance:
 - a. To install in a new instance, click **Create an installation directory**. Either accept the default value or supply another location, if you wish.
 - b. To install in an existing instance, click **Reuse an existing installation directory**.

From the list of existing instances, select the one you want to use.

After making your selection, click **Next**.

7. Set the installation directory (product home) for the Web GUI or accept the default. Then click **Next**.
8. Choose the type of installation and click **Next**.
9. **Advanced installation in a new directory only:** Specify whether you want to use the ObjectServer as a user repository or a file-based user repository and click **Next**.
10. In the WebSphere information window, provide the credentials of the administrative user and the port the Web GUI uses. Then click **Next**.
11. **Advanced installation in a new directory only:** Specify the context root of the Tivoli Integrated Portal and click **Next**.
12. If you selected the ObjectServer in step 9, select the default user registry and click **Next**:
 - **File based repository:** Newly-created users and user groups are created in the file-based repository, and users and groups are read from both the file-based repository and the ObjectServer repository.
 - **ObjectServer:** Newly-created users and user groups are created in the ObjectServer repository, and users and groups are read from both the file-based repository and the ObjectServer repository.
13. Provide the characteristics of the primary ObjectServer and click **Next**.
If you have a secondary ObjectServer, set **Enable Secondary Server for Failover** before clicking **Next**.
14. Optional: If you set **Enable Secondary Server for Failover** enter the details of the secondary ObjectServer and click **Next**.
15. Check that the details on the summary window are correct, then click **Install**.

The installation can take some time depending on your processor configuration. The installation window shows the progress of the process along with the name of the current installation task.

16. When the installation complete window appears, click **Done**.

The login window for the Web GUI opens in a browser window.

Related tasks

“Performing post-installation tasks” on page 208

Using the console installer

The console installer enables you to install the Tivoli Netcool/OMNIbus Web GUI from the command line.

Before you begin

Before you begin the installation, carry out the following steps:

1. **AIX** If a non-root user is to perform the installation, carry out one of the following actions as a root user:
 - Make sure that the user has access to the **slibclean** command and that they run the command before they begin the installation. The executable file for this command is typically in `/usr/sbin/slibclean`.
 - Run the **slibclean** command before the user begins the installation.
2. If you intend to install the Web GUI into an existing instance of the Tivoli Integrated Portal, stop that instance.

For instructions on how to stop a Tivoli Integrated Portal server, refer to the *IBM Tivoli Netcool/OMNIbus Web GUI Administration and User's Guide*.
3. Log in as the user for the installation. That user must have write permission to the directory where you are to install Tivoli Netcool/OMNIbus Web GUI.

UNIX **Linux** Always log in as the preferred user for the installation. Do not log in as root and then switch to the preferred user.
4. Have to hand the installation information you gathered during the preparation.

Use this procedure to install the Tivoli Netcool/OMNIbus Web GUI using the GUI installer on each computer where you want to install the product.

1. **UNIX** If you are not running XServer, unset the `DISPLAY` variable. Use one of the following sets of commands, depending on the shell you use:
 - `unset DISPLAY`
 - `set DISPLAY=`
`export DISPLAY`
2. Change to the `cdimage` directory of the installation DVD or the downloaded installation image.
3. Enter the following command:
 - **Windows** `install.exe -i console`
 - **Linux** `./install.sh -i console`
 - **UNIX** `./install.sh -i console`
4. At each prompt supply the corresponding item of information you gathered during the preparation.

When supplying information be sure to use escape characters in the way that Java properties expects them. Non-text characters must be UTF-8 escaped.

Related tasks

“Performing post-installation tasks” on page 208

Using the silent installer

Use the silent installer to deploy the Tivoli Netcool/OMNIBus Web GUI with identical settings on multiple computers. The installer obtains the installation settings from a response file and does not prompt you for any information.

Before you begin

Before you begin the installation, carry out the following steps:

1. **AIX** If a non-root user is to perform the installation, carry out one of the following actions as a root user:
 - Make sure that the user has access to the **slibclean** command and that they run the command before they begin the installation. The executable file for this command is typically in `/usr/sbin/slibclean`.
 - Run the **slibclean** command before the user begins the installation.

2. If you intend to install the Web GUI into an existing instance of the Tivoli Integrated Portal, stop that instance.

For instructions on how to stop a Tivoli Integrated Portal server, refer to the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

3. Log in as the user for the installation. That user must have write permission to the directory where you are to install Tivoli Netcool/OMNIBus Web GUI.

UNIX **Linux** Always log in as the preferred user for the installation. Do not log in as root and then switch to the preferred user.

4. Have to hand the installation information you gathered during the preparation.

1. **UNIX** If you are not running XServer, unset the `DISPLAY` variable. Use one of the following sets of commands, depending on the shell you use:

- `unset DISPLAY`
- `set DISPLAY=`
`export DISPLAY`

2. In the `cdimage` directory of the DVD or downloaded installation image edit `sample_response.txt` and modify the settings to match the information you gathered during the preparation.

Alternatively, you can copy a response file you set up previously on another computer to this one.

When modifying the settings in the file:

- Remove the comment marker from the line for the **LICENSE_ACCEPTED** parameter and set the value of that parameter to true. Note that doing this signifies your acceptance of the product license agreement.
 - Remove the comment marker from one of the entries for **USER_INSTALL_DIR** depending on the platform you are using.
 - Remove the comment marker from one of the entries for **IAGLOBAL_OMNIBUS_WEBGUI_HOME** depending on the platform you are using.
3. In the `cdimage` directory of the DVD or downloaded installation image enter the following command:
 - **Windows** `install.exe -f full_path_to_sample_response\sample_response.txt`

- `Linux` `UNIX` `./install.sh -f full_path_to_sample_response/sample_response.txt`

Replace *full_path_to_sample_response* with the full path of the directory that contains *sample_response.txt*.

Ensure that you enter escape characters the way the Java properties expect them. Non-text characters must be UTF-8 escaped (such as `\u0022` for the " double quote). In addition, be sure to specify the full (absolute) path of the response file.

Results

The installer adds the Web GUI to the system using the information in the response file.

What to do next

The passwords entered in the response file can be seen by anyone who has read access to the file. When you have completed the installation, remove the file or move it to a secure place.

Related tasks

"Performing post-installation tasks" on page 208

Installation parameters in the response file

Every item of information in the GUI installer has its equivalent in a console parameter that you can see in the sample response file (for a silent installation) and enter at the command line or in a script.

Installation parameters

The following lists the parameters that appear in the response file together with the initial setting, if any, of each.

The passwords entered in the response file can be seen by anyone who has read access to the file. When you have completed the installation, remove the file or move it to a secure place.

LICENSE_ACCEPTED=false

Used to present the license prompt. Set to true, it signifies your acceptance of the product license agreement. A setting of false stops the installation process.

DE_INSTALL_MODE=0

Sets the user access policy of the Deployment Engine when you are installing as the root user (on the UNIX and Linux operating systems) or an Administrator user (on the Windows operating system). Possible values are as follows:

0: All users

1: Single user access (the root or Administrator user)

2: Users in a specific operating system group and the root or Administrator user.

If you use the value 2 for this parameter, supply a value for the `DE_GROUP_NAME` parameter.

The default value is 0.

DE_GROUP_NAME=

The name of the operating system user group whose members have access to the Deployment Engine. Define this parameter when the **DE_INSTALL_MODE** parameter has the value 2.

USER_INSTALL_DIR=C:\\IBM\\tivoli\\tipv2

The installation directory to use for the Tivoli Integrated Portal.

Windows The \ backslash is seen as an escape character. Use \\ two backslashes when defining the path.

UNIX **Linux** The default installation directory is /opt/IBM/tivoli/tipv2.

If Tivoli Netcool/OMNIBus Web GUI (the Web GUI) has been installed before, you can specify the existing location to reuse the instance.

IAGLOBAL_OMNIBUS_WEBGUI_HOME=C:\\IBM\\TIVOLI\\tivoli\\netcool\\omnibus_webgui

The installation directory to use for the Tivoli Netcool/OMNIBus Web GUI.

Windows The \ backslash is seen as an escape character. Use \\ two backslashes when defining the path.

UNIX **Linux** The default installation directory is /opt/IBM/tivoli/netcool/omnibus_webgui.

IAGLOBAL_INSTALL_LOCATION_SELECTION=create

Specifies whether to use an existing location or to create a new one. Set the parameter to reuse to use an existing Tivoli Integrated Portal location. Otherwise use a value of create.

CHOSEN_INSTALL_SET=default

Specifies whether to use the default install set or to allow the user to create a customized install set. The parameter can take the values default (the chosen install set is used) or advanced (the chosen install set can be configured by the user).

IAGLOBAL_USER_REGISTRY_OBJECTSERVER_SELECTED=true

Specifies whether to use the ObjectServer as a user registry. The parameter can take the values true (ObjectServer is used as a user registry) or false (ObjectServer is not used as user registry).

IAGLOBAL_WASUserID=tipadmin**IAGLOBAL_WASPassword=mypassword**

The user ID and password of the administrator of the application server. The tipadmin ID is the default user ID, which you can change to another name. The password entered here is required when you log in to the Web GUI. The password cannot begin with a hyphen (-). Note that the password is also used for the default accounts that the installer creates: ncouser and ncoadmin.

IAGLOBAL_WC_defaulthost=16310

The port that the application server uses. You can change the port number so long as it is not already in use. At installation time, if the port you specified is in use, the installer attempts to use this port number plus 30. If that is in use it tries this port number plus 50.

IAGLOBAL_CONSOLE_CONTEXT_ROOT=/ibm/console

The context root for the TIP console and hence the Web GUI. This determines the URL that users supply to access TIP. Setting this parameter has no effect when the value of **IAGLOBAL_INSTALL_LOCATION_SELECTION** is reuse.

IAGLOBAL_LOCALE

The locale that the installer users. To use a non-English locale, set the value of this parameter accordingly, for example zh_cn. When the parameter has no value, the installer uses the en locale.

IAGLOBAL_DEFAULT_USER_REGISTRY_SELECTION= OBJECT_SERVER

The place where new users and groups are stored. The values of this parameter are FILE_BASED (users and groups are held in the local file system) or OBJECT_SERVER (users and groups are held in the ObjectServer. If you set this parameter to OBJECT_SERVER, set

IAGLOBAL_USER_REGISTRY_OBJECTSERVER_SELECTED to true. If you use FILE_BASED, set **IAGLOBAL_USER_REGISTRY_OBJECTSERVER_SELECTED** to false.

Setting this parameter has no effect when the value of **IAGLOBAL_INSTALL_LOCATION_SELECTION** is reuse.

IAGLOBAL_OBJECTSERVER_USER=root**IAGLOBAL_OBJECTSERVER_PASSWORD=*********IAGLOBAL_OBJECTSERVER_PRIMARY_NAME=NCOMS**

The username, password, and name of the ObjectServer that supplies the Web GUI with data. Set each of these values according to the configuration of your ObjectServer.

IAGLOBAL_OBJECTSERVER_PRIMARY_HOST=myobjectserver.ibm.com**IAGLOBAL_OBJECTSERVER_PRIMARY_PORT=4100**

The name of the host and the port that the ObjectServer uses. Set each of these with values according to the configuration of your ObjectServer.

IAGLOBAL_OBJECTSERVER_ENABLE_SECONDARY_SERVER=false**IAGLOBAL_OBJECTSERVER_SECONDARY_HOST=****IAGLOBAL_OBJECTSERVER_SECONDARY_PORT=**

The characteristics of an optional, secondary ObjectServer. If your site uses a secondary server, set **IAGLOBAL_OBJECTSERVER_ENABLE_SECONDARY_SERVER** to true, and set **IAGLOBAL_OBJECTSERVER_SECONDARY_HOST** and **IAGLOBAL_OBJECTSERVER_SECONDARY_PORT** to the host name and port that the ObjectServer uses.

Guidelines for default and advanced installations

Use the following guidelines when editing a response file in preparation for a default or advanced installation of Tivoli Netcool/OMNIBus Web GUI using the silent installer.

For a default installation, set the values of the following parameters appropriate to your site, or use the default values:

LICENSE_ACCEPTED
USER_INSTALL_DIR
IAGLOBAL_OMNIBUS_WEBGUI_HOME
IAGLOBAL_INSTALL_LOCATION_SELECTION
IAGLOBAL_WASUserID
IAGLOBAL_WASPassword
IAGLOBAL_WC_defaulthost
IAGLOBAL_OBJECTSERVER_USER
IAGLOBAL_OBJECTSERVER_PASSWORD
IAGLOBAL_OBJECTSERVER_PRIMARY_NAME
IAGLOBAL_OBJECTSERVER_PRIMARY_HOST

IAGLOBAL_OBJECTSERVER_PRIMARY_PORT

In addition set values for the following parameters if your site uses a secondary ObjectServer:

IAGLOBAL_OBJECTSERVER_ENABLE_SECONDARY_SERVER

IAGLOBAL_OBJECTSERVER_SECONDARY_HOST

IAGLOBAL_OBJECTSERVER_SECONDARY_PORT

For an advanced installation define the parameters required for a default installation. In addition you can define any of the following features, as required at your site:

- Context root: **IAGLOBAL_CONSOLE_CONTEXT_ROOT** (only if **IAGLOBAL_INSTALL_LOCATION_SELECTION** has the value create)
- Type of user registry: **IAGLOBAL_DEFAULT_USER_REGISTRY_SELECTION** and **IAGLOBAL_USER_REGISTRY_OBJECTSERVER_SELECTED** (only if **IAGLOBAL_INSTALL_LOCATION_SELECTION** has the value create)
- Other parameters to suit the needs of your site.

Running the installer in an existing environment

The Tivoli Integrated Portal platform is laid down during product installation. You can install additional products and they will all share the same platform.

Before you begin

Back up the current *tip_home_dir* directory branch in case you want to revert to that installation.

When a product is installed into an existing Tivoli Integrated Portal environment, some options might be disabled, depending on what was installed before. When you rerun the installer, the product installation runs in maintenance mode.

1. Back up the deployment engine database in case you want to revert to that installation. You might also want to back up the *tip_home_dir* directory for any data files that you need to retrieve.
2. If you will be running in silent mode, update the *sample_response.txt* file with the features to be installed.
3. Run the installation program in silent mode.

Upgrading and migrating to the Web GUI component

You can upgrade an existing installation of IBM Tivoli Netcool/Webtop or the Web GUI component of Tivoli Netcool/OMNIBus V7.3.0 to the Web GUI component of Tivoli Netcool/OMNIBus V7.3.1, and migrate data from Netcool/Webtop to the Web GUI.

The actions that you take to upgrade and migrate depend on which version of Netcool/Webtop or the Web GUI you have installed:

Netcool/Webtop V2.2 or Web GUI V7.3.0

Install the Tivoli Netcool/OMNIBus V7.3.1 Web GUI in a different location to the existing Netcool/Webtop installation and use the upgrade tool to export the data, files, and configuration options from Netcool/Webtop or Web GUI V7.3.0 and import them to the Web GUI.

Netcool/Webtop V2.1, V2.0, or V1.3.1

Install the Tivoli Netcool/OMNIBus V7.3.1 Web GUI in a different location to the existing Netcool/Webtop installation and use the migration tool to export the data, files, and configuration options from Netcool/Webtop and import them to the Web GUI.

Upgrade and migration notes

Use this information to understand how IBM Tivoli Netcool/Webtop features are migrated to the Tivoli Netcool/OMNIBus V7.3.1 Web GUI and, if required, the Migration Tool works.

Versions of Netcool/Webtop earlier than V2.2 do not use the Tivoli Integrated Portal framework. Netcool/Webtop V2.1 and V2.0 use the IBM Tivoli Netcool GUI Foundation framework.

How IBM Tivoli Netcool/Webtop features are migrated to the Tivoli Netcool/OMNIBus Web GUI

Use this information to understand how data is migrated from IBM Tivoli Netcool/Webtop to Tivoli Netcool/OMNIBus V7.3.1.

- “Data sources”
- “Entities”
- “Initialization file” on page 187
- “Maps” on page 187
- “SmartPages” on page 187
- “WAAPI” on page 187

Data sources

During an upgrade, the existing `ncwDataSourceDefinitions.xml` data source configuration file is backed up and migrated to a new format. The changes to the format are:

- The `<results-cache>` element contains all cache-tuning attributes. This element has a child element `<config>`.
- The `<ncwResultsCacheParameters>` element is removed.
- The cache-tuning attributes are moved from the `<ncwResultsCacheParameters>` element to the `<config>` element.

Entities

Tivoli Netcool/OMNIBus V7.3.1 does not use the entity feature. During the upgrade or migration process, the entity configuration artifacts are migrated to XML format. The original configuration artifacts are backed up.

After you migrated from an earlier version of Netcool/Webtop the entity configuration artifacts are migrated as follows:

Entities

Migrated to filters. A filter category, called a *system filter*, is added to Tivoli Netcool/OMNIBus V7.3.1; entities from Netcool/Webtop are migrated to system filters.

Entity views

Migrated to views. A view category, called a *system view*, is added to Tivoli Netcool/OMNIBus V7.3.1; entity views from Netcool/Webtop are migrated to system views.

Entity groups

Migrated to a feature called *filter collections*. Administrators can manage filter collections from the Filter Builder.

Initialization file

During the upgrade or migration process, the properties of the Netcool/Webtop `server.init` file are merged with the V7.3.1 Web GUI `server.init` file. On an upgraded or a migrated V7.3.1 Web GUI installation, in the `webgui_home_dir/etc/server.init` file, migrated properties are denoted by the following comment: Migrated from old Webtop.

Maps

During an upgrade, all existing map configuration artifacts are backed up and upgraded as follows:

- The entity attributes of all map objects are replaced with filter attributes.
- An attribute, called `filtertype` is added to all map objects. For each upgraded map, the `filtertype` attribute has the value `system`.

SmartPages

The deprecation of entities from Tivoli Netcool/OMNIBus V7.3 and later affects the following SmartPage commands:

- **insert:AEL**
- **insert:TableView**

The parameters of these commands change as follows:

- The entity and filter parameters are deprecated.
- The `filtername` parameter is introduced. This parameter specifies the name of the filter that is invoked by the SmartPage command.
- The `filtertype` parameter is introduced. This parameter describes the type of filter that is specified by the `filtername` parameter.

If your existing installation of IBM Tivoli Netcool/Webtop uses custom HTML pages with the preceding commands, the commands are not migrated. The pages function in Tivoli Netcool/OMNIBus V7.3.1. However, an entry denoting the use of deprecated features is added to the log file in `tip_home_dir/profiles/TIPProfile/logs/`. Additionally, if you create a new SmartPage-based HTML page that uses a deprecated parameter, the use of the parameter is logged.

WAAPI

In Tivoli Netcool/OMNIBus V7.3.1, if any of the following WAAPI elements are used, an entry is added to the log file in `tip_home_dir/profiles/TIPProfile/logs/` to denote the use of a deprecated feature. These elements continue to function as in your existing version of Netcool/Webtop.

- `entity`
- `entitygroup`

- entitylist

If any of the following attributes are used, an entry is added to the log file in *tip_home_dir/profiles/TIPProfile/logs/* to denote the use of a deprecated feature. These attributes continue to function as in your existing version of Netcool/Webtop.

- entity
- entity_status_indicator

If any of the following values of the methodName attribute are used, an entry is added to the log file in *tip_home_dir/profiles/TIPProfile/logs/* to denote the use of a deprecated feature. These values continue to function as in your existing version of Netcool/Webtop.

- entity.addEntity
- entity.createOrReplaceEntity
- entity.deleteEntity
- entity.deleteEntityForced
- entity.modifyEntity
- entity.setDefaultGroup
- entity.setDefaultView

If elements, attributes or methodName values pertaining to entities are used in Tivoli Netcool/OMNIBus V7.3.1, the entity is interpreted as a system filter. If elements, attributes or methodName values pertaining to entity views are used in Tivoli Netcool/OMNIBus V7.3.1, the entity is interpreted as a system view.

Related reference

“Element reference” on page 523

Upgrade tool overview

The upgrade tool transfers configuration data from the Netcool/Webtop version 2.2 or Web GUI version 7.3.0 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI. That configuration data includes Tivoli Integrated Portal version 1.0 or 1.1.1 information, user information, filters, views, and pages.

The upgrade tool can be used if the old and new systems are installed on different servers that are not physically connected. So, you can use the tool to transfer data from a production server running a previous version of Netcool/Webtop or Web GUI to a test server.

The upgrade tool consists of an export module and an import module.

Export module

The export module collects data from Netcool/Webtop or Web GUI and packages it in an archive file. The files that are exported include:

- Netcool/Webtop or Web GUI configuration files including `server.init` but excluding `ncwDataSourceDefinitions.xml`
- CGI scripts
- Chart XML files
- Menus and tools
- Maps and map resources
- WAAPI configuration files

- Views
- Filters
- Security information

Import module

The import module loads the exported data into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

The module can run in the following ways:

Migration

Loads the data in the archive file into the appropriate directories in the Web GUI and Tivoli Integrated Portal directory structures.

Rollback

Reverses the changes made during the upgrade.

Upgrade tool prerequisites

Before you run the upgrade take note of the information you need.

For upgrade, you require some of the characteristics that you defined when installing the existing Netcool/Webtop and Web GUI server, and the new Web GUI server. For both servers you need:

- The user name of the Tivoli Integrated Portal Administrator, for example, tipadmin.
- The password for the Tivoli Integrated Portal Administrator. for example, tippass.
- The installation directory of the Tivoli Integrated Portal.

Migration tool overview

The migration tool migrates IBM Tivoli Netcool GUI Foundation pages and IBM Tivoli Netcool/Webtop configuration data from Netcool/Webtop versions 1.3.1, 2.0, and 2.1 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

The migration tool can also be used if the old and new systems are installed on different servers that are not physically connected. So, you can use the migration tool to migrate data from a production server running a previous version of Netcool/Webtop to a test server.

Restriction: The migration tool migrates most, but not all IBM Tivoli Netcool GUI Foundation pages and layouts to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

The migration tool consists of an export and an import module.

Export module

The export module collects data from Netcool/Webtop. The following table lists the files that are exported, depending on the version of the existing installation.

Table 50. Files that the export module exports from Netcool/Webtop

Versions of the existing installation	Files exported from Netcool/Webtop
1.3.1, 2.0, and 2.1	Netcool/Webtop configuration files

Table 50. Files that the export module exports from Netcool/Webtop (continued)

Versions of the existing installation	Files exported from Netcool/Webtop
2.0 and 2.1	<p>User-created IBM Tivoli Netcool GUI Foundation pages.</p> <p>Netcool Security Manager data pertaining to users (user name, password, first name, last name, the “user active” flag), groups (group name, group display name, the users belonging to the group) and roles.</p> <p>If the external repository storing security data is the Object Server, the script exports the users (user name, first name, last name, the “user enabled” flag) and the groups (group name, group display name, the users belonging to the group).</p> <p>Note: If the external repository is the Object Server, the user password is not exported and it will be set to a blank password after the import.</p>
1.3.1	Netcool/Webtop local users

Import module

The import module transforms the exported data and loads it into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

The import module can be run in one of the following ways:

Migration

Transforms exported data and loads it into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

Rollback

Reverses changes made in a migration.

The following table lists the files that are imported, depending on the version of the existing installation.

Table 51. Files that the import module imports into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

Versions of the existing installation	Files imported to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI
1.3.1, 2.0, and 2.1	Netcool/Webtop configuration files.
2.0 and 2.1	<p>IBM Tivoli Netcool GUI Foundation pages.</p> <p>Netcool Security Manager data into the ObjectServer or LDAP, depending on the settings.</p>
1.3.1	Netcool/Webtop local users into the ObjectServer or LDAP, depending on the settings.

Related reference

“Netcool GUI Foundation files not migrated” on page 208

Migration tool prerequisites

Before you run the migration tool, take note of the security and WAAPI requirements, and invalid characters.

Security settings

For migration, you require some of the characteristics that you defined during the installation and configuration of Web GUI. Gather the following information:

- The user name of the Tivoli Integrated Portal Administrator, for example, tipadmin.
- The password for the Tivoli Integrated Portal Administrator, for example, tippass.
- The user registry, which can be one of the following types:
 - Lightweight Directory Access Protocol (LDAP) server
 - ObjectServer
 - Tivoli Integrated Portal file-based
- If your site uses an LDAP registry, obtain base Distinguished Name (DN) of the LDAP server.
- If your site uses an ObjectServer registry, obtain the following information:
 - The name of the server that runs the ObjectServer
 - The port that the ObjectServer uses
 - The name of the ObjectServer root user.
 - The password for the ObjectServer root user.

In addition, decide on a default password for all users once they are imported in to the Web GUI.

- If your site uses a file-based registry, decide on a default password for all users after they are imported in to the Web GUI.

WAAPI client

Determine whether the WAAPI client is installed on the existing Netcool/Webtop servers.

Invalid characters

Review the use of special characters in the existing Netcool/Webtop servers, and compare them against the characters listed in *webgui_home_dir/etc/illegalChar.prop* on the Web GUI. Make a note of any characters used in the existing servers and appears in the file.

Related tasks

“Migrating from IBM Tivoli Netcool/Webtop versions 2.0 or 2.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI” on page 194

“Migrating from IBM Tivoli Netcool/Webtop version 1.3.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI” on page 199

Upgrading from IBM Tivoli Netcool/Webtop version 2.2 or Tivoli Netcool/OMNIBus version 7.3.0 Web GUI to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI

To upgrade your existing Netcool/Webtop V2.2 or Tivoli Netcool/OMNIBus V7.3.0 data to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI, run the upgrade tool export module scripts on the existing server. Then import the data into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

Before you begin

Obtain the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI installation package and, if necessary, extract its contents.

The migration process migrates files generated by Netcool/Webtop or the Web GUI and some user-modified files. However some manual migration steps might be required.

The procedure to upgrade to the Web GUI has the following parts:

1. Install the upgrade tool.
2. Export the data from the existing Netcool/Webtop or Web GUI server.
3. Install the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.
4. Copy the data to the new Web GUI server.
5. Import the data.
6. Remove the upgrade tool.

Related concepts

“Web GUI installation or upgrade prerequisites” on page 174

Related tasks

“Obtaining the installation package” on page 174

Installing the upgrade tool

1. Navigate to the `cdimage` directory of the installation DVD for Tivoli Netcool/OMNIBus version 7.3.1 Web GUI, or the downloaded installation image and copy the following file to the existing server:
`C0I/PackageSteps/Webtop/FILES/Preupgrade.zip`
2. On the existing server, extract the `Preupgrade.zip` file to the following directory:
`tip_dir/profiles/TIPProfile`
Replace `tip_dir` with the name of the installation directory of the Tivoli Integrated Portal on the existing server.
The directory where you extract the file is called `UPGRADE_TOOL_HOME`.

Exporting the data

1. Log in to the existing server as an administrative user.
Make sure that the existing Netcool/Webtop or Web GUI server is running.
2. Navigate to the `UPGRADE_TOOL_HOME` directory.
3. The ESS Server is an optional component for Tivoli Integrated Portal that the existing version of Netcool/Webtop or Web GUI uses. If your installation of the Tivoli Integrated Portal does not include the ESS Server carry out the following:
 - a. Navigate to the plugins directory in `UPGRADE_TOOL_HOME`.
 - b. Edit the file `OMNIBusWebGUI.properties`.
 - c. Locate the following line and insert a comment marker (#) at the beginning of that line.
`components=ESSServer`
 - d. Save the file and exit from the text editor.
4. Navigate to the `bin` directory in `UPGRADE_TOOL_HOME`.
5. Enter one of the following commands:

```
UNIX      Linux      preupgrade.sh tipdir --username tipadmin --password tippass --productId OMNIBusWebGUI --ignoreDEListGeneration true
```

```
Windows   preupgrade.bat tipdir --username tipadmin --password tippass --productId OMNIBusWebGUI --ignoreDEListGeneration true
```

Replace *tipdir* with the installation directory of the Tivoli Integrated Portal, *tipadmin* with the user ID of the Tivoli Integrated Portal administrative user, and *tippass* with the password for that user.

The data is exported to a file named `upgradeData.zip` in `UPGRADE_TOOL_HOME/upgrade/data`. In addition the utility creates a log file (named `tipExport.log`) in `install_dir/profiles/TIPProfile/logs`, where *install_dir* is the installation directory for the Tivoli Integrated Portal.

Installing the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI

Install the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI on the new Web GUI server.

Copying the data

Copy `upgradeData.zip` to the new Web GUI server. Put the file in `webgui_home_dir/integration/plugins`.

Importing the data

1. Log in to the new server as an administrative user and make sure that the Web GUI server is running.
2. Edit the `ncwDataSourceDefinitions.xml` file and apply any settings you want to copy over from the existing Netcool/Webtop or Web GUI server.
3. If the Tivoli Integrated Portal on the existing Netcool/Webtop or Web GUI server did not include the ESS Server, carry out the following:
 - a. Navigate to `webgui_home_dir/integration/plugins` and edit the file `OMNIBusWebGUI.properties`.
 - b. Locate the following line and insert a comment marker (#) at the beginning of that line.
`components=ESSServer`
 - c. Save the file and exit from the text editor.

4. Navigate to *tip_home_dir*/profiles/TIPProfile/upgrade/bin and enter one of the following commands:

```
UNIX      Linux      upgrade.sh tip_home_dir --username tipadmin  
--password tippass --productId OMNIBusWebGUI --upgradeDataFile  
webgui_home_dir/integration/plugins/upgradeData.zip
```

```
Windows   upgrade.bat tip_home_dir --username tipadmin --password tippass  
--productId OMNIBusWebGUI --upgradeDataFile webgui_home_dir/integration/  
plugins/upgradeData.zip
```

Replace *tip_home_dir* with the full path of the installation directory for the Tivoli Integrated Portal, *tipadmin* with the user ID of the Tivoli Integrated Portal administrative user, and *tippass* with the password for that user.

The tool creates two log files (named *tipcli.log* and *upgrade.log*) in *tip_home_dir*/profiles/TIPProfile/logs.

5. Restart the Tivoli Integrated Portal server.

Related tasks

“Restarting the server” on page 568

Removing the upgrade tool

Remove the upgrade tool from the existing Web GUI server.

Migrating from IBM Tivoli Netcool/Webtop versions 2.0 or 2.1 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI

To migrate your existing Netcool/Webtop version 2.0 or version 2.1 data to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI, run the migration tool export module scripts on the servers on which IBM Tivoli Netcool Security Manager and Netcool GUI Foundation are installed. Then import the migration data into the Tivoli Netcool/OMNIBus version 7.3 Web GUI.

Before you begin

Install the Tivoli Netcool/OMNIBus V7.3.1 Web GUI. After installation of the Web GUI, the migration tool is in *webgui_home_dir*/integration/migration_tool/migration_tool_export.zip.

The migration process migrates all files generated by Netcool/Webtop and some user-modified files. However, some manual migration steps might be required.

The procedure to migrate from Netcool/Webtop to the Web GUI has the following parts:

1. Install the migration tool.
2. Export data from the Netcool/Webtop server.
3. Copy the data to the Web GUI server.
4. Configure the import module.
5. Import the data.
6. Remove migration tool.

Related concepts

“Migration tool overview” on page 189

“Migration tool prerequisites” on page 191

Related tasks

“Viewing the installation log file” on page 218

Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173

“Rolling back migration” on page 202

“Removing the migratedRoles.war file” on page 203

Related reference

“Netcool GUI Foundation to Tivoli Integrated Portal migration notes” on page 204

“Netcool GUI Foundation files not migrated” on page 208

Installing the migration tool

1. On the Web GUI host, copy the `webgui_home_dir/integration/migration_tool/migration_tool_export.zip` file to the Netcool/Webtop host.

If the Netcool Security Manager component and the Netcool GUI Foundation component are installed on separate servers, copy the `migration_tool_export.zip` file to each server.

2. On each server, extract the `migration_tool_export.zip` file to any suitable directory.

This directory is called `MIGRATION_TOOL_HOME`.

Exporting the data

Before you begin

Before exporting the data, ensure that the names of all user roles contain only letters, numbers, and the underscore character. In particular, make sure that roles names do not include the minus sign (-).

1. Log in to the server where Netcool Security Manager is installed as an administrative user.
2. Set `NCHOME` to the installation directory of Netcool Security Manager.
3. Enter:

```
MIGRATION_TOOL_HOME/bin/sm_migration_export The user data is exported to a file named SecurityMigration.xml in MIGRATION_TOOL_HOME/output/SecurityManager.
```

4. Optional: the `sm_migration_export` script may fail if there is a problem with role names. If this occurs, go to `MIGRATION_TOOL_HOME/etc` and edit the file `rolesRenaming.properties` to define the role mappings. Then repeat step 3
5. If the Netcool GUI Foundation is on a separate server, log in to that machine and set `NCHOME` to the Netcool GUI Foundation installation directory.
6. Edit `settings.properties` in `MIGRATION_TOOL_HOME/etc` and set the following values:

Table 52. Migration tool settings for exporting Netcool GUI Foundation data

Property	Value
<code>NGF.Server.URL</code>	The URL of the Netcool GUI Foundation.
<code>NGF.Admin.user</code>	The user ID of the Netcool GUI Foundation administrative user.

Table 52. Migration tool settings for exporting Netcool GUI Foundation data (continued)

Property	Value
NGF.Admin.password	The password of the Netcool GUI Foundation administrative user.

- If WAAPI is not installed, edit `export.lst` in the same directory, and comment out the following line:
`com.ibm.tivoli.nc.migration.plugin.webtop.WAAPIInit2xExportPlugin`
- Navigate to `MIGRATION_TOOL_HOME/bin` and enter:
`migration_export`
The data is exported to a file named `data.zip` in `MIGRATION_TOOL_HOME/output`.

Copying data to the Web GUI server

Copy `SecurityMigration.xml` and `data.zip` to the Web GUI server.

- Put `SecurityMigration.xml` in `webgui_home_dir/integration/migration_tool/output/SecurityManager`.
- Put `data.zip` in `webgui_home_dir/integration/migration_tool/output`.

Configuring the import module

In this task, `MIGRATION_TOOL_HOME` refers to `webgui_home_dir/integration/migration_tool`

- Set the following variables:

Table 53. Environment variables for importing data

Variable	Value
<code>TIPHOME</code>	The installation directory of the Tivoli Integrated Portal.
<code>PROD_HOME</code>	The installation directory of the Tivoli Netcool/OMNIBus V7.3.1 Web GUI.

- Edit `settings.properties` in `MIGRATION_TOOL_HOME/etc` and set the following values:

Table 54. Migration tool settings for importing data

Property	Value
TIP.WSAdmin.user	The Tivoli Integrated Portal administrative user. For example: <code>tipadmin</code> .
TIP.WSAdmin.password	The password for the Tivoli Integrated Portal administrative user. For example: <code>tippass</code> .

Table 54. Migration tool settings for importing data (continued)

Property	Value
Importer.Destination.Choice	<p>The registry into which the Netcool Security Manager is imported. Use one of the following values:</p> <p>FBAUTH Imports the data into the default Tivoli Integrated Portal file-based repository.</p> <p>NCOS Imports the data into the ObjectServer. The users are created automatically in the ObjectServer.</p> <p>NONE Does not import the data.</p> <p>LDAP Imports the data into an LDAP registry. The user data is imported into a .ldiff file, which you must then import into LDAP.</p>
Set the following properties if the value of Importer.Destination.Choice is NCOS:	
Importer.NCOS.Server	The name of the server that runs the ObjectServer.
Importer.NCOS.Port	The port number that the ObjectServer uses.
Importer.NCOS.Admin	The ObjectServer root user.
Importer.NCOS.Password	The password for the ObjectServer root user.
Importer.NCOS.defaultPassword	A default Web GUI password to generate for all imported users.
Set the following properties if the value of Importer.Destination.Choice is FBAuth:	
Importer.FBAUTH.defaultPassword	A default Web GUI password to generate for all imported users.
Importer.FBAUTH.DefaultWIMRealm	Do not change this property.
Set the following property if the value of Importer.Destination.Choice is LDAP:	
Importer.LDAP.BaseDn	The Distinguished Name (DN) of the LDAP server.

- If WAAPI was not installed on the Netcool/Webtop server, edit `import.lst` in `MIGRATION_TOOL_HOME/etc` and comment out the following line:

```
com.ibm.tivoli.nc.migration.plugin.webtop.WAAPIInitImportPlugin
```

Importing the data

In this task, `MIGRATION_TOOL_HOME` refers to `webgui_home_dir/integration/migration_tool`

- Make sure that the Web GUI and Tivoli Integrated Portal server is running.
- Review the file `webgui_home_dir/etc/illegalChar.prop` and ensure it is appropriate for your installation.
 For most installations, it is sufficient to remove the space character from the list of invalid characters.
- Navigate to the `MIGRATION_TOOL_HOME/bin` directory and enter:

```
migration_import -migration
```

4. If you set the property **Importer.Destination.Choice** in `settings.properties` to LDAP, import the file `generatedUsersAndGroups.ldif` into the LDAP server. Refer to the documentation for your LDAP server for instructions on how to import a `.ldif` file.
5. To import the users and groups from Netcool/Webtop go to the `MIGRATION_TOOL_HOME/import/roles` directory and enter:
 - UNIX Linux `addAllRolesAndRelationships tip_home_dir tipadmin tippass`
 - Windows `addAllRolesAndRelationships "tip_home_dir" tipadmin tippass`Replace `tip_home_dir` with the installation directory of the Tivoli Integrated Portal, `tipadmin` with the user ID of the Tivoli Integrated Portal administrative user and `tippass` with the password for that user.
6. Restart the Tivoli Integrated Portal server.

Results

After you have restarted the server, the users and groups migrated from Netcool Security Manager and the pages migrated from Netcool GUI Foundation are visible. The users are assigned to the same roles as in Netcool Security Manager.

What to do next

To obtain the complete default V7.3.1 Web GUI configuration artifacts, merge the files contained in `webgui_home_dir/etc/default` folder with the configuration artifacts that were migrated from Netcool/Webtop. For example, to obtain the default global filters, merge the `webgui_home_dir/etc/default/data/global/filter.xml` file with the `webgui_home_dir/etc/data/global/filter.xml` file.

Removing the migration tool

After the migration has run successfully, delete the `settings.properties` file, which contains login and password information. You can also remove the migration tool from all hosts.

Rerunning the migration tool

If needed you can rerun the migration tool. Before you can rerun the tool, you must remove the following files and directories:

1. Rollback the migration on the Web GUI server.
2. Remove the following files and directories:
 - a. On the Netcool Security Manager server, you must remove the `MIGRATION_TOOL_HOME/output/SecurityManager` directory before you can rerun the **sm_migration_export** command.
 - b. On the Netcool GUI Foundation server, you must remove the `MIGRATION_TOOL_HOME/output` directory before you can rerun the **migration_export** command.
 - c. On the Web GUI server, you must remove the `MIGRATION_TOOL_HOME/output` directory before you can rerun the **migration_import** command. You must also remove the `migratedRoles.war` file before you can rerun the `addAllRolesAndRelationships` script.

Migrating from IBM Tivoli Netcool/Webtop version 1.3.1 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI

To migrate Netcool/Webtop version 1.3.1 data to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI, run the migration tool export module on the host on which Netcool/Webtop version 1.3.1 is installed. After that, you can import the migration data into the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI.

The migration process migrates all files generated by Netcool/Webtop and some user-modified files. However, some manual migration steps might be required.

To migrate from Netcool/Webtop to the Web GUI:

The procedure to migrate from Netcool/Webtop to the Web GUI has the following parts:

1. Install the migration tool.
2. Export the data from the Netcool/Webtop server.
3. Copy the data to the Web GUI server.
4. Configure the import module.
5. Import the data.
6. Remove migration tool.

Related concepts

“Migration tool overview” on page 189

“Migration tool prerequisites” on page 191

Related tasks

“Viewing the installation log file” on page 218

Chapter 6, “Installing, upgrading, and uninstalling the Web GUI component,” on page 173

“Rolling back migration” on page 202

“Removing the migratedRoles.war file” on page 203

Related reference

“Netcool GUI Foundation files not migrated” on page 208

Installing the migration tool

1. On the Web GUI host, copy the *webgui_home_dir/integration/migration_tool/migration_tool_export.zip* file to the Netcool/Webtop host.
2. Extract the *migration_tool_export.zip* file to any suitable directory.
This directory is known as *MIGRATION_TOOL_HOME*.

Exporting the data

1. Log in to the server where Netcool/Webtop is installed as an administrative user.
2. Set *WEBTOP_HOME* to the installation directory of Netcool/Webtop.
3. If WAAPI is not installed, edit *export.lst* in *MIGRATION_TOOL_HOME/etc*, and comment out the following line:
`com.ibm.tivoli.nc.migration.plugin.webtop.WAAPIInit13ExportPlugin`
4. Navigate to *MIGRATION_TOOL_HOME/bin* and enter:

UNIX

Linux

`webtop13_migration_export`

Windows

`webtop13_migration_export.cmd`

The data is exported to a file named `data.zip` in `MIGRATION_TOOL_HOME/output`.

Copying data to the Web GUI server

Copy `data.zip` to the Web GUI server.

Put the file in `webgui_home_dir/integration/migration_tool/output`.

Configuring the import module

In this task, `MIGRATION_TOOL_HOME` refers to `webgui_home_dir/integration/migration_tool`

1. Set the following variables:

Table 55. Environment variables for importing data

Variable	Value
<code>TIPHOME</code>	The installation directory of the Tivoli Integrated Portal.
<code>WEBTOP_HOME</code>	The installation directory of the Tivoli Netcool/OMNIBus V7.3.1 Web GUI.

2. Edit `settings.properties` in `MIGRATION_TOOL_HOME/etc` and set the following values:

Table 56. Migration tool settings for importing data

Property	Value
<code>TIP.WSAdmin.user</code>	The Tivoli Integrated Portal administrative user. For example: <code>tipadmin</code> .
<code>TIP.WSAdmin.password</code>	The password for the Tivoli Integrated Portal administrative user. For example: <code>tippass</code> .
<code>Importer.Destination.Choice</code>	<p>The registry into which the Netcool Security Manager is imported. Use one of the following values:</p> <p>FBAUTH Imports the data into the default Tivoli Integrated Portal file-based repository.</p> <p>NCOS Imports the data into the ObjectServer. The users are created automatically in the ObjectServer.</p> <p>NONE Does not import the data.</p> <p>LDAP Imports the data into an LDAP registry. The user data is imported into a <code>.ldiff</code> file, which you must then import into LDAP.</p>
Set the following properties if the value of <code>Importer.Destination.Choice</code> is <code>NCOS</code> :	
<code>Importer.NCOS.Server</code>	The name of the server that runs the ObjectServer.
<code>Importer.NCOS.Port</code>	The port number that the ObjectServer uses.
<code>Importer.NCOS.Admin</code>	The ObjectServer root user.

Table 56. Migration tool settings for importing data (continued)

Property	Value
Importer.NCOS.Password	The password for the ObjectServer root user.
Importer.NCOS.defaultPassword	A default Web GUI password to generate for all imported users.
Set the following properties if the value of Importer.Destination.Choice is FBAuth:	
Importer.FBAUTH.defaultPassword	A default Web GUI password to generate for all imported users.
Importer.FBAUTH.DefaultWIMRealm	Do not change this property.
Set the following property if the value of Importer.Destination.Choice is LDAP:	
Importer.LDAP.BaseDn	The Distinguished Name (DN) of the LDAP server.

3. If WAAPI was not installed on the Netcool/Webtop server, edit `import.lst` in `MIGRATION_TOOL_HOME/etc` and comment out the following line:
`com.ibm.tivoli.nc.migration.plugin.webtop.WAAPIInitImportPlugin`

Importing the data

In this task, `MIGRATION_TOOL_HOME` refers to `webgui_home_dir/integration/migration_tool`

1. Make sure that the Web GUI and Tivoli Integrated Portal server is running.
2. Review the file `webgui_home_dir/etc/illegalChar.prop` and ensure it is appropriate for your installation.
 For most installations, it is sufficient to remove the space character from the list of invalid characters.
3. Go to the `MIGRATION_TOOL_HOME/bin` directory and enter:

UNIX **Linux** `webtop13_migration_import -migration`

Windows `webtop13_migration_import.cmd -migration`
4. If you set the property **Importer.Destination.Choice** in `settings.properties` to LDAP, import the file `generatedUsersAndGroups.ldif` into the LDAP server.
 Refer to the documentation for your LDAP server for instructions on how to import a `.ldif` file.
5. To import the users and groups from Netcool/Webtop go to the `MIGRATION_TOOL_HOME/import/roles` directory and enter:
 - UNIX

Linux

`addAllRolesAndRelationships.sh tip_home_dir tipadmin tippass`
 - Windows

`addAllRolesAndRelationships.bat "tip_home_dir" tipadmin tippass`

Replace `tip_home_dir` with the installation directory of the Tivoli Integrated Portal, `tipadmin` with the user ID of the Tivoli Integrated Portal administrative user and `tippass` with the password for that user.
6. Restart the Web GUI and Tivoli Integrated Portal server

Results

After you have restarted the server, the configuration data migrated from Netcool/Webtop is in place along with the migrated users.

What to do next

To obtain the complete default V7.3.1 Web GUI configuration artifacts, merge the files contained in *webgui_home_dir/etc/default* folder with the configuration artifacts that were migrated from Netcool/Webtop. For example, to obtain the default global filters, merge the *webgui_home_dir/etc/default/data/global/filter.xml* file with the *webgui_home_dir/etc/data/global/filter.xml* file.

Removing the migration tool

After the migration has run successfully, delete the *settings.properties* file, which contains login and password information. You can also remove the migration tool from all hosts.

Rerunning the migration tool

If needed you can rerun the migration tool. Before you can rerun the tool:

1. Rollback the migration on the Web GUI server.
2. Remove the following files and directories:
 - a. On the Netcool/Webtop server, you must remove the *MIGRATION_TOOL_HOME/output* directory before you can rerun the **webtop13_migration_export** command.
 - b. On the Web GUI server, you must remove the *MIGRATION_TOOL_HOME/output* directory before you can rerun the **migration_import** command. You must also remove the *migratedRoles.war* file before you can rerun the **addAllRolesAndRelationships** script.

Rolling back migration

If you want to undo the changes made by the process of migrating to the Tivoli Netcool/OMNIbus V7.3.1 Web GUI, you can roll back the migration. When the migration is rolled back, deleted and changed files, and configurations from the previous version of IBM Tivoli Netcool/Webtop are restored.

MIGRATION_TOOL_HOME refers to the *webgui_home_dir/etc/integration/migration_tool* directory on the Web GUI server.

Restriction: The rollback process does not remove roles, users, or user groups that were imported to the Tivoli Netcool/OMNIbus V7.3.1 Web GUI by using the **addAllRolesAndRelationships** script. The roles, users, and groups are stored in the *migratedRoles.war* file.

To roll back migration:

1. Ensure that the Tivoli Integrated Portal server is running.
2. On the Tivoli Netcool/OMNIbus V7.3.1 Web GUI host, set *TIPHOME* to point to the directory where you have installed Tivoli Integrated Portal and *NCHOME* to point to the directory where you have installed the Web GUI.
3. Navigate to the *MIGRATION_TOOL_HOME/bin* directory.
4. Enter the following command:

```
migration_import -rollback
```

What to do next

Check the *migration_import.log* file in the following directory:
MIGRATION_TOOL_HOME/log.

If you need to remigrate files and configurations from a IBM Tivoli Netcool/Webtop installation to the Tivoli Netcool/OMNIBus V7.3.1 Web GUI, you must manually remove the `migratedRoles.war` file before you can use the **`addAllRolesAndRelationships`** script to migrate users, roles, and groups.

Removing the `migratedRoles.war` file

If you need to remigrate files and configurations from a IBM Tivoli Netcool/Webtop installation to the Tivoli Netcool/OMNIBus V7.3.1 Web GUI, you must manually remove the `migratedRoles.war` file before you can use the **`addAllRolesAndRelationships`** script to migrate users, roles, and groups.

Before you begin

Make sure you have rolled back your migrated Tivoli Netcool/OMNIBus V7.3.1 Web GUI installation to Netcool/Webtop. Also make sure that you have set *TIPHOME* to point the Tivoli Integrated Portal installation directory and *NCHOME* to point to the Web GUI installation directory.

To remove the `migratedRoles.war` file:

1. Navigate to the *TIPHOME/bin* directory.
2. Enter the following command:

```
wsadmin -user tipadmin -password tippass -c "$AdminApp update isclite  
modulefile {-operation delete -contenturi migratedRoles.war}"
```

Replace:

`tipadmin`

with the user name of the administrator user specified during the installation of Netcool/Webtop.

`tippass`

with the password of the administrator user.

3. To save the configuration, enter:

```
$AdminConfig save
```

Results

The `migratedRoles.war` file is removed. Note that the removal of the `migratedRoles.war` file does not remove the migrated roles, users, and user groups.

What to do next

You can now rerun the `addAllRolesAndRelationships` script. You can ignore the following messages should the script generate them when you rerun it:

- WARNING: Group: *groupname* already exists
- WARNING: Group: *groupname* is already assigned to user: *username*

Netcool GUI Foundation to Tivoli Integrated Portal migration notes

When migrating an older installation of Netcool/Webtop that uses Netcool GUI Foundation, to the Tivoli Netcool/OMNIBus V7.3.1 Web GUI, some manual migration steps are required to ensure equivalence.

Tivoli Netcool/OMNIBus V7.3.1 Web GUI components are displayed in Tivoli Integrated Portal by using Tivoli Integrated Portal GUI technology. Consequently, the look and feel is different to a Netcool/Webtop deployment in Netcool GUI Foundation.

“Security IDs”

“Components”

“Layout” on page 206

“Localized pages” on page 206

“Authorization” on page 207

Security IDs

Only the default user and the user_view security IDs are migrated. If any Netcool GUI Foundation GUI items, such as pages, tabs, menu options or views use security IDs other than user or user_view, the conversion process treats them as if the user security ID was specified. After migration, administrator users can open all migrated pages in Tivoli Integrated Portal.

Note: Warning messages are logged, specifying which Netcool GUI Foundation GUI items with unsupported security ID were converted to the default user security ID.

Important: During migration, the migration tool assumes that the standard user and user_view security IDs are applied. This means that if the standard Netcool GUI Foundation user or user_view security IDs have been customized in the previous Netcool/Webtop installation, this is ignored during migration and standard settings are applied.

Components

The following table describes the how the components of Netcool/Webtop in Netcool GUI Foundation map to the components of the Tivoli Netcool/OMNIBus Web GUI in Tivoli Integrated Portal.

Table 57. Tivoli Netcool/OMNIBus Web GUI components: Netcool GUI Foundation to Tivoli Integrated Portal

Netcool GUI Foundation	Corresponding Tivoli Integrated Portal	Function
AELAction	AELPortlet	View Active Event List rendered into portal page Edit Administrators can configure the AEL using the AELPortlet Preferences Editor

Table 57. Tivoli Netcool/OMNibus Web GUI components: Netcool GUI Foundation to Tivoli Integrated Portal (continued)

Netcool GUI Foundation	Corresponding Tivoli Integrated Portal	Function
MapAction	MapPortlet	View Map rendered into portal page Edit Administrators can configure which map is rendered into the page using the MapPortlet Preferences Editor
LELAction	LELPortlet	View Lightweight Event List rendered into portal page Edit Administrators can configure the LEL using the AELPortlet Preferences Editor
Custom viewpoints	IFramePortlet	View Custom viewpoints rendered as IFrame portlets Edit Administrators can configure the portlets using the Entity Configuration window
TableviewAction	TableviewPortlet	View Events rendered into portal page as HTML table Edit Administrators can configure the table using the TableviewPortlet Preferences Editor
ChartAction	ChartPortlet	View Chart image rendered into portal page based on preferences Edit Administrators can configure how a chart image is generated using the ChartPortlet Preferences Editor
N/A	AboutPortlet	Provides Web GUI version information

Layout

The migrated Netcool/Webtop version 2.2 layout is different from the original Netcool GUI Foundation layout.

NGF pages

Each migrated page becomes a Tivoli Integrated Portal view, which can be selected from the **View** list above the navigation pane in the user interface.

When a view is selected:

- The navigation pane in the user interface is filtered to show only the nodes associated with the migrated page, now a view.
- All tabs, including menu options open as tabs in the work area.

NGF tabs and menu options

Each tab becomes another view nested underneath the original page, now displayed as a view. When a view is selected from the Tivoli Integrated Portal **View** list, the original tabs are displayed underneath the **View as tree node** elements and can be selected. Access to each view will be the same as for the parent folder of the original page.

When menu panes and tab panes, including state-maintained tab panes, are migrated to Tivoli Integrated Portal, menu and tab panes are transformed into tree leaves and are displayed in the navigation area as children of the tree leaf for a migrated page.

NGF columns

The following Netcool GUI Foundation layouts are supported:

- One column
- Two columns 25/75, 34/66, 50/50, 75/25
- Three columns 25/50/25, 33/33/33

Empty columns, that is, columns that do not contain viewpoints, are displayed in Netcool GUI Foundation, but are not migrated to Tivoli Integrated Portal.

NGF views

Only views referenced from the Netcool GUI Foundation Menu Pane or Tab Pane layouts are migrated. The migration of Views that have been referred from pages using column layouts is not supported.

Custom NGF viewpoints

Customized, user-created viewpoints embedded on Netcool GUI Foundation pages are transformed into Tivoli Integrated Portal IFrame portlets that are displayed underneath, and can be selected from, the **View** drop-down list.

The following custom viewpoints parameters, however, are ignored during migration because Tivoli Integrated Portal portlets do not have corresponding parameters:

- Hidden
- Application
- Cached On URL

Localized pages

Netcool GUI Foundation supports localized pages by maintaining a separate version of the page for each language into which the page was translated. As a

result, during migration each localized Netcool GUI Foundation page is transformed into a separate Tivoli Integrated Portal page.

All Tivoli Integrated Portal page corresponding to a localized Netcool GUI Foundation page are grouped together in folders that represent localized Netcool GUI Foundation pages on the navigation tree. The different language versions are indicated by language-specific prefixes.

Authorization

Tivoli Integrated Portal navigation folders

Tivoli Integrated Portal navigation folders that contain other navigation elements and therefore do not directly open a page, but merely contain other navigation elements, have the Tivoli Integrated Portal role “All authenticated portal users” assigned. This provides access for all authenticated Tivoli Integrated Portal users.

Tivoli Integrated Portal navigation links

Tivoli Integrated Portal navigation links that open a page have the Tivoli Integrated Portal role or name `NGF_USER_{username}` or `{rolename}` or `NGF_GROUP_{groupname}` assigned, depending on page assignment to **User/Role/Group** in Netcool GUI Foundation. This role provides user or editor access to Tivoli Integrated Portal pages, depending on the Security ID assigned to **Page/Tab/Menu Option/View** in Netcool GUI Foundation. Security IDs other than user and user_views are not migrated.

- The user security ID gives editor access for users with the user or admin roles.
- The user_views security ID gives user access for users with the user role, and editor access for users with the admin role.

Note: If a security ID other than user or user_views applies to the old Netcool GUI Foundation **Page/Tab/Menu Option/View** settings, it is not migrated and the security ID user is applied. This requires an administrator to manually apply the old Netcool GUI Foundation security ID settings in Tivoli Integrated Portal. To do so, an administrator creates a new role in Tivoli Integrated Portal, then assigns it to the required Tivoli Integrated Portal pages, and then assigns it to the users who need to access these pages.

Tivoli Integrated Portal portlet roles and security IDs

The roles assigned to Tivoli Integrated Portal portlets correspond to the security ID assigned to the corresponding viewpoints in Netcool GUI Foundation. All security IDs are migrated.

Changes to the Restricted user group

In older versions of Netcool/Webtop, access to the Delete tool requires membership of the Restricted group. When you install the Web GUI, the Restricted user group is no longer created and all users will have access to the Delete tool. However, when you run the migration tool, the “Restricted” group is created and all previous settings, such as restricted access to the Delete tool, are maintained.

Netcool GUI Foundation files not migrated

When you run the migration tool, some of the Netcool GUI Foundation PSML files are not migrated to the Tivoli Netcool/OMNIBus Web GUI installation.

The following table describes the PSML files that are not migrated by default. To suppress the migration of further files, modify the *MIGRATION_TOOL_HOME/etc/not_migratable_psmls.lst* file (*MIGRATION_TOOL_HOME* represents the directory where you extracted the migration tool on the Netcool/Webtop server).

Table 58. PSML files not migrated

File (association)	Page title	Description
default.psml (user-associated)	New Page Template	Not a real page, but a template for user-created Netcool GUI Foundation pages
default.psml (role-associated)	Desktop	Default page, has no counterpart in Tivoli Integrated Portal
managepages.psml (role-associated)	My Pages	A page used to manage Netcool GUI Foundation pages Tivoli Integrated Portal provides its own mechanisms for managing pages
admin.psml (role-associated)	Administration (for Netcool GUI Foundation)	Netcool GUI Foundation administration page Tivoli Integrated Portal has its own administrative mechanism
ncw_admin.psml (role-associated)	Netcool/Webtop Admin	Version 7.3.1 has a new set of administration pages
webtop.psml (role-associated)	Netcool/Webtop Desktop	A set of demo pages for Netcool/Webtop 2.0, 2.1 that do not apply to the Tivoli Netcool/OMNIBus Web GUI

Performing post-installation tasks

After installation, there are a number of setup tasks, some required and others that are optional, for completing the initial setup of your product environment.

Related tasks

“Using the GUI installer” on page 178

Logging in

Log in to the console whenever you want to start a work session.

Before you begin

The Tivoli Integrated Portal Server must be running before you can connect to it from your browser.

Complete these steps to log in:

1. In a Web browser, enter the URL of the Tivoli Integrated Portal Server:
`http://host.domain:16310/ibm/console`; or `https://host.domain:16311/ibm/console` if it is configured for secure access.

- *host.domain* is the fully qualified host name or IP address of the Tivoli Integrated Portal Server (such as *MyServer.MySubdomain.MyDomain.com* or *9.51.111.121*, or *localhost* if you are running the Tivoli Integrated Portal Server locally).
 - 16310 is the default nonsecure port number for the administrative console and 16311 is the default secure port number. If your environment was configured with a port number other than the default, enter that number instead. If you are not sure of the port number, read the application server profile to get the correct number.
 - *ibm/console* is the default path to the Tivoli Integrated Portal Server, however this path is configurable and might differ from the default in your environment.
2. In the login page, enter your user ID and password and click **Log in**. This is the user ID and password that are stored with the Tivoli Integrated Portal Server.

Attention: After authentication, the web container used by the Tivoli Integrated Portal Server redirects to the last URL requested. This is usually `https://host:port/ibm/console`. But if you manually change the page URL, after being initially directed to the login page, or if you make a separate request to the server in a discrete browser window before logging in, you may be redirected unexpectedly.

Note: If you have more than one instance of the Tivoli Integrated Portal Server installed on your computer, do not run more than one instance in a browser session, that is, do not log in to different instances on separate browser tabs.

Results

After your user credentials have been verified, the console Welcome page is displayed. If you entered the localhost or port number incorrectly, the URL does not resolve. View the application server profile to check the settings for localhost, port, and user ID.

What to do next

Select any of the items in the navigation tree to begin working with the console.

While you are logged into the Tivoli Integrated Portal Server, avoid clicking the browser **Back** button to go to the previous Web page because you will be logged out automatically. Click **Forward** and you are logged out and must resubmit your credentials to log in again.

Note: If you want to use single sign-on (SSO) then you must use the fully-qualified domain name of the Tivoli Integrated Portal host.

Accepting the security certificate

When logging in, you might see a security alert with a message that says there is a problem with the security certificate. This indicates that the browser application is verifying the security certificate of the application server.

Self-signed or CA-signed certificate

The application server uses a self-signed security certificate. You might see a Security Alert when you first connect to the portal that alerts you to a problem with the security certificate. You might be warned of a possible invalid certificate and be recommended to not log in.

Although this warning appears, the certificate is valid and you can accept it. Or, if you prefer, you can install your own CA-signed certificate. For information on creating your own CA-signed certificate, go to: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/tsec_sslcreateCArequest.html

For more information about certificates, go to the IBM WebSphere Application Server Community Edition Documentation Project at <http://publib.boulder.ibm.com/wasce/V2.1.1/en/overview.html>, and search for *Managing trust* and *Managing SSL certificates*.

Protecting the vault key file

The encryption key for the administrator password is held in the vault key file. Establish strict read-only access to this file.

To restrict access to the file:

1. Change to the `webgui_home_dir/etc/encrypt` directory.
2. Use the method provided by your operating system to ensure that the `vault.key` file has read-only access.

Example

UNIX **Linux** Use the following commands:

```
cd opt/IBM/tivoli/netcool/omnibus_webgui/etc/encrypt
chmod 444 vault.key
```

Windows Use Windows Explorer to navigate to `C:\IBM\tivoli\netcool\omnibus_webgui\etc\encrypt`. Right click on the `vault.key` file and choose **Properties**. Then select the **Read-only** check box and click **OK**.

Assigning Web GUI roles to the administrative user

To enable the administrative user that was created during installation to access the Web GUI pages and portlets, you must assign additional roles to that user.

You specify the user name and password of the administrative user during installation. The default user name is `tipadmin`.

To assign Web GUI roles to the administrative user:

1. Log in to the Web GUI with the administrative user credentials that you provided during installation.

2. Assign the administration roles to that user or add the user to the Netcool_OMNIBus_Admin group.

Table 59. Adding roles or groups to the administrative user

Activity	Procedure
Assign administrative roles to the user.	<ol style="list-style-type: none">1. Click Users and Groups > User Roles.2. Click Search.3. Locate the user (for example tipadmin) in the grid at the foot of the page and click its unique name.4. In the User Roles page, select the following check boxes:<ul style="list-style-type: none">• ncw_admin• ncw_user<p>Attention: Do not clear any check boxes that are already set.</p>5. Click Save.
Add the user to the Netcool_OMNIBus_Admin group.	<ol style="list-style-type: none">1. Click Users and Groups > Manage Users.2. Click Search.3. Locate the user (for example tipadmin) in the grid at the foot of the page and click its User ID.4. On the User Properties page, click the Groups tab and click Add.5. On the Add User to groups page click Search.6. Click the Netcool_OMNIBus_Admin group name, click Add, and then click Close.7. Click the General tab and click OK.

3. Log out and log back in to the Web GUI.

Results

After you have logged back in, the Web GUI portlets and pages are displayed in the navigation pane.

What to do next

Click the following links in the navigation pane to access the Web GUI:

- To access the administrative functions, for example the Filter Builder, View Builder, Tools Editor, and the Map Creation resources, click **Administration > Event Management Tools**.
- To access the event display functions, click **Availability > Events**.

Changing the passwords of the supplied users

Initially the supplied users (ncouser and ncoadmin) have the same password as the administrative user. For security reasons you may wish to change the passwords for these users.

To change the passwords of the ncouser and ncoadmin accounts:

1. Make sure you are logged in as the administrative user (for example, tipadmin).
2. Click **Users and Groups > Manage Users**.
3. Click **Search**.
4. For each of the passwords you want to change:
 - a. Click on the User ID in the list at the foot of the page.
 - b. Enter the new password in the **Password** and **Confirm password** fields.
 - c. Click **OK**.

Setting up the WAAPI client

To configure the usage of predictive eventing and IBM Tivoli Application Dependency Discovery Manager (TADDMM) event monitoring, you must configure a minimal setup for the WAAPI client by specifying a user and password.

Before you begin

You must have assigned the ncw_admin role to the administrative user, or to the required WAAPI client user.

Important: You must perform step 1 at a minimum before you can configure the Web GUI for predictive eventing or for monitoring TADDMM events. This configuration requires the use of the **runwaapi** command, for which a user and password must be specified.

To set up the WAAPI client:

1. Edit the *webgui_home_dir/waapi/etc/waapi.init* file and set the values of the following properties:

waapi.user

Type your user name. The user must have the ncw_admin role assigned.

waapi.password

Type your password.

waapi.port

Optional: If, during installation, you changed the port for the Web GUI server from the default of 16310, type the port.

2. Optional: Set the values of the remaining properties in the file.

Note: You do not have to specify the rest of the properties at this time; the **waapi.user** property and the **waapi.password** property are sufficient for configuring predictive eventing or for TADDMM event monitoring.

3. Save and close the file.

Results

The user and password for the WAAPI client are now set, and you can now run the **runwaapi** command.

For more information about using the WAAPI client to administer the Web GUI, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Related tasks

“Assigning Web GUI roles to the administrative user” on page 210

“Enabling predictive eventing in the Web GUI” on page 532

“Enabling support for TADDM events in the Web GUI” on page 534

Enabling multicultural support for the Web GUI

You might have to perform additional configuration steps to display the Web GUI in your national language.

Configuring the Web GUI for GB18030 characters

To make your Chinese Web GUI installation compliant with the GB18030 standard for Chinese characters, you must install the GB18030 character set on your system, and configure the client systems to display GB18030 characters.

Before you begin

Make sure that you have met the prerequisites for your operating system:

- **UNIX** **Linux** The Web GUI client and server operating systems must have the zh_CN.utf8 locale installed.
- All operating systems: The fonts that support GB18030 must be installed, as follows:
 - **UNIX** **Linux** You might have to download the fonts separately.
 - **Windows** The support package that contains font support for GB18030 must be installed.

For more information about the requirements of your operating system, refer to the documentation provided by your operating system vendor.

These configuration steps pertain to the Web GUI component only. You must also configure the other Tivoli Netcool/OMNIBus components for use with GB18030.

To configure the Web GUI for GB18030 compliance:

1. **UNIX** **Linux** On the client and server operating systems, set the LANG environment variable and the LC_ALL environment variable to LC_ALL=zh_CN.utf8.
2. **Windows** On the client operating systems, set up font linking with the SimSun-18030 font.
3. Enable your Web browser to automatically determine language encoding.

Related concepts

“Web browsers, JREs, and mobile devices for the Web GUI” on page 14

“Setting your locale” on page 403

Setting the default timezone for new users

To ensure that new users have the correct timezone settings, you must specify which timezone is applied when the users are created.

When a new user is created, the initial values or default values are populated from the following file: *webgui_home_dir/etc/system/userdefaults.props*. The default timezone setting for new users is GMT+00:00. Permissible timezone values are specified in the tz database. For more information about the tz database, see the following Web site:

<http://www.twinsun.com/tz/tz-link.htm>

For possible timezone values, see the following Web site:

<http://twiki.org/cgi-bin/xtra/tzdatepick.html>

After you have edited the *userdefaults.props* file, you must restart the server.

Tip: The timezone settings for existing users are stored in the following file: *webgui_home_dir/etc/configstore/ncwUserPreferences/username.nova*, where *username* is the respective user.

To change the default timezone for new users:

1. Open the *webgui_home_dir/etc/system/userdefaults.props* file.
2. Set the value of the **timezone** property to the required time zone.

Tip: Choose the name of a locale-based timezone (for example America/Chicago) rather than one relative to GMT (for example, etc/GMT-6).

3. Set the value of the **acl_user_properties_timezone_updated** parameter to true.
4. Restart the server.

Results

When new users are created, the timezone in the *webgui_home_dir/etc/configstore/ncwUserPreferences/username.nova* files is set to the value of the **timezone** property.

Related tasks

“Restarting the server” on page 568

Uninstalling the Web GUI

Uninstall the Web GUI when you no longer need it on a computer using one of three methods.

Before you begin

If the server is part of a load-balancing cluster, remove it from the cluster.

Related tasks

“Troubleshooting a failed uninstallation on Windows” on page 221

Using the GUI uninstaller

How to uninstall the Web GUI using the GUI uninstaller.

To use the GUI uninstaller:

1. From the command-line interface, change to the OMNIBusWebGUI uninstall directory:

```
cd webgui_home_dir/_uninst/OMNIBusWebGUI
```


For example: `/opt/IBM/tivoli/netcool/omnibus_webgui/_uninst/OMNIBusWebGUI` or `C:\IBM\tivoli\netcool\omnibus_webgui_uninst\OMNIBusWebGUI`.
2. Enter the following command:

```
uninstall -i swing
```


The GUI uninstaller starts.
3. At the welcome screen, click **Next**.
4. Supply the administrator user ID and password. Then click **Uninstall**.
5. Click **Done** to exit from the uninstaller.
6. Delete the `webgui_home_dir` directory if it remains.
7. Depending on the usage of the Tivoli Integrated Portal on this computer, determine the steps to take next:

Table 60. Actions for completing the uninstallation

Tivoli Integrated Portal Usage	What to do
No other applications use this instance of Tivoli Integrated Portal and there are no other Tivoli Integrated Portal instances on this computer.	<ol style="list-style-type: none">1. Navigate to <code>tip_home_dir/WebSphereUpdateInstallerV7/uninstall</code>, if it exists, and run the command: <pre>./uninstall</pre>2. Delete the <code>tip_home_dir</code> if it remains.
No other applications use this instance of Tivoli Integrated Portal and there are instances of Tivoli Integrated Portal on this computer.	Delete all directories in <code>tip_home_dir</code> , if it remains, except <code>WebSphereUpdateInstallerV7</code> and its subdirectories, if they exist.

Results

The Web GUI is removed from the system. In addition, the uninstaller removes the Deployment Engine (DE) if there are no products registered with it after the Web GUI is removed.

Using the console uninstaller

How to uninstall the Web GUI using the console uninstaller.

To use the console uninstaller:

1. **UNIX** If you are not running XServer, unset the `DISPLAY` variable. Use one of the following sets of commands, depending on the shell you use:
 - `unset DISPLAY`
 - `set DISPLAY=`
`export DISPLAY`
2. From the command-line interface, change to the OMNIBus Web GUI uninstall directory:

```
cd webgui_home_dir/_uninst/OMNIBusWebGUI
```

For example: /opt/IBM/tivoli/netcool/omnibus_webgui/_unist/OMNIBusWebGUI
or C:\IBM\tivoli\netcool\omnibus_webgui_unist\OMNIBusWebGUI.

3. Enter the following command:
`uninstall -i console`
The console uninstaller starts.
4. Enter the administrator user ID and password when requested.
5. Enter the number 1 to start the uninstallation process.
6. Delete the *webgui_home_dir* directory if it remains.
7. Depending on the usage of the Tivoli Integrated Portal on this computer, determine the steps to take next:

Table 61. Actions for completing the uninstallation

Tivoli Integrated Portal Usage	What to do
No other applications use this instance of Tivoli Integrated Portal and there are no other Tivoli Integrated Portal instances on this computer.	<ol style="list-style-type: none">1. Navigate to <i>tip_home_dir</i>/WebSphereUpdateInstallerV7/uninstall, if it exists, and run the command: <code>./uninstall</code>2. Delete the <i>tip_home_dir</i> if it remains.
No other applications use this instance of Tivoli Integrated Portal and there are instances of Tivoli Integrated Portal on this computer.	Delete all directories in <i>tip_home_dir</i> , if it remains. except WebSphereUpdateInstallerV7 and its subdirectories, if they exist.

Results

The Web GUI is removed from the system. In addition, the uninstaller removes the Deployment Engine (DE) if there are no products registered with it after the Web GUI is removed.

What to do next

Linux If the uninstallation fails with an error similar to the following example, reset the DISPLAY environment variable.

```
/space/leecyp/installer/omni731/webgui/linux/cdimage >./install.sh -i console
Preparing to install...
Extracting the JRE from the installer archive...
Unpacking the JRE...
Extracting the installation resources from the installer archive...
Configuring the installer for this system's environment...

Launching installer...

X connection to localhost:11.0 broken (explicit kill or server shutdown).
```

Using the silent uninstaller

How to uninstall the Web GUI using the silent uninstaller.

To use the silent uninstaller:

1. **UNIX** If you are not running XServer, unset the DISPLAY variable. Use one of the following sets of commands, depending on the shell you use:
 - unset DISPLAY
 - set DISPLAY=
 - export DISPLAY
2. Copy `uninstall_response.txt` from the `cdimage` directory of the installation DVD or the downloaded installation image to a suitable, secure directory.
3. Edit the copy of `uninstall_response.txt` and modify the settings as follows:

IAGLOBAL_WASUserID

The user ID of the Tivoli Integrated Portal administrative user. For example: `tipadmin`.

IAGLOBAL_WASPassword

The password for the Tivoli Integrated Portal administrative user. For example: `tippass`.

4. From the command-line interface, change to the uninstall directory:
`cd webgui_home_dir/_uninst/OMNIBusWebGUI`
For example: `/opt/IBM/tivoli/netcool/omnibus_webgui/_uninst/OMNIBusWebGUI` or `C:\IBM\tivoli\netcool\omnibus_webgui_uninst\OMNIBusWebGUI`.

5. Enter this command:

- **Windows** `uninstall -f uninstall_response_location\uninstall_response.txt`
- **UNIX** **Linux** `uninstall -f uninstall_response_location/uninstall_response.txt`

Replace `uninstall_response_location` with the full path of the directory that contains `uninstall_response.txt`.

The console uninstaller starts.

6. Delete the `webgui_home_dir` directory if it remains.
7. Depending on the usage of the Tivoli Integrated Portal on this computer, determine the steps to take next:

Table 62. Actions for completing the uninstallation

Tivoli Integrated Portal Usage	What to do
No other applications use this instance of Tivoli Integrated Portal and there are no other Tivoli Integrated Portal instances on this computer.	<ol style="list-style-type: none">1. Navigate to <code>tip_home_dir/WebSphereUpdateInstallerV7/uninstall</code>, if it exists, and run the command: <code>./uninstall</code>2. Delete the <code>tip_home_dir</code> if it remains.
No other applications use this instance of Tivoli Integrated Portal and there are instances of Tivoli Integrated Portal on this computer.	Delete all directories in <code>tip_home_dir</code> , if it remains, except <code>WebSphereUpdateInstallerV7</code> and its subdirectories, if they exist.

Results

The Web GUI is removed from the system. In addition, the uninstaller removes the Deployment Engine (DE) if there are no products registered with it after the Web GUI is removed.

What to do next

The password entered in the response file can be seen by anyone who has read access to the file. When you have completed the uninstallation, remove the file or move it to a secure place.

Troubleshooting installation

Review the following information for help and support in resolving installation issues you might encounter.

Viewing the installation log file

If the installation fails, the process generates an installation log file of actions performed during the installation. You can use this file to troubleshoot the failure.

To check the log files for your installation:

1. Navigate to the *webgui_home_dir/logs* directory and look for the compressed *omnibus_webgui_install_logs.zip* file.
2. Extract the contents of this file to a temporary location.
3. Analyze the content of the logs to help you determine the cause of the failure.

Tip: If the compressed log file does not exist, you can manually locate the logs.

Results

The compressed log file contains information from the following locations:

- Log details located in *webgui_home_dir/_uninst/OMNIBusWebGUI/plan*
- Log details located in *tip_home_dir/logs*
- Installer log details in one of the following directories:
 - **Linux** **UNIX** */home/username/IBM_Tivoli_Netcool_OMNIBus_Web_GUI_Install-xx.log*
 - **Windows** *C:\Documents and Settings\username\IBM_Tivoli_Netcool_OMNIBus_Web_GUI_Install-xx.log*
- Deployment Engine (DE) log details in one of the following directories:

Table 63. Deployment Engine (DE) directories

Operating system and user	Location
UNIX non-root installation	<i>/home/username/.acsi_machinename</i>
UNIX root installation	<i>/usr/ibm/common/acsi</i>
Windows administrator	<i>C:\Program Files\IBM\Common\acsi</i>

TIPProfile_create log

Review the TIPProfile_create log when your installation ends in error.

Purpose

The TIPProfile_create log records the messages that result from the successful or failed completion of a task in the process of creating the Tivoli Netcool/OMNIBus Web GUI profile during installation.

Sample

This is a sample of the final records of a TIPProfile_create.log where errors were encountered.

```
<record>
  <date>2008-05-19T01:20:43</date>
  <millis>1211185243859</millis>
  <sequence>1007</sequence>
  <logger>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</logger>
  <level>INFO</level>
  <class>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</class>
  <method>areCommandLineArgumentsValid</method>
  <thread>10</thread>
  <message>Validation Error for profilePath: The profile path is not valid.
</message>
</record>
<record>
  <date>2008-05-19T01:20:43</date>
  <millis>1211185243859</millis>
  <sequence>1008</sequence>
  <logger>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</logger>
  <level>SEVERE</level>
  <class>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</class>
  <method>invokeWSProfile</method>
  <thread>10</thread>
  <message>Argument Validation Failed.</message>
</record>
<record>
  <date>2008-05-19T01:20:43</date>
  <millis>1211185243859</millis>
  <sequence>1009</sequence>
  <logger>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</logger>
  <level>INFO</level>
  <class>com.ibm.ws.profile.cli.WSProfileCLIModeInvoker</class>
  <method>invokeWSProfile</method>
  <thread>10</thread>
  <message>Returning with return code: INSTCONFFAILED</message>
</record>
<record>
  <date>2008-05-19T01:20:43</date>
  <millis>1211185243859</millis>
  <sequence>1010</sequence>
  <logger>com.ibm.wsspi.profile.WSProfileCLI</logger>
  <level>INFO</level>
  <class>com.ibm.wsspi.profile.WSProfileCLI</class>
  <method>invokeWSProfile</method>
  <thread>10</thread>
  <message>Returning with return code: INSTCONFFAILED</message>
</record>
```

Harmless installation messages

A review of the installation log might show error messages that are actually harmless.

After installing Tivoli Netcool/OMNIBus Web GUI, you might encounter a reflection error when reviewing the installation logs. The installation is successful, but the log shows variations of this error:

```
+++ Warning +++: IWAV0003E Could not reflect methods for com.ibm.sec.iauthz.  
InstanceAuthzServiceLocalHome because one of the methods references a type that  
could not be loaded.  
Exception: java.lang.NoClassDefFoundError: com.ibm.sec.iauthz.InstanceAuthorization  
+++ Warning +++: IWAV0002E Failed reflecting values  
+++ Warning +++: java.lang.NoClassDefFoundError: com.ibm.sec.  
iauthz.InstanceAuthorization
```

This error can be safely ignored.

Viewing installed packages

You can view the versions of all installed packages at any time by using scripts. You might be asked for package information by IBM Software Support.

To view a list of packages installed:

1. Open a command prompt.
2. Change to one of the following locations depending on the operating system and the user who installed the product:

Table 64. Deployment Engine (DE) directories

Operating system and user	Location
UNIX non-root installation	/home/username/.acsi_machinename
UNIX root installation	/usr/ibm/common/acsi
Windows administrator	C:\Program Files\IBM\Common\acsi

3. To set the environment variables for the DE, run the appropriate command for your operating system:

- **UNIX** **Linux** setenv.sh
- **Windows** setenv.cmd

4. To list the installed packages, run the appropriate command for your operating system:

- **UNIX** **Linux** /bin/listIU.sh
- **Windows** /bin/listIU.cmd

Results

The list of packages is displayed.

Troubleshooting a failed installation on Solaris operating systems

If an installation using the root user fails on a Solaris operating system that uses Solaris zones partitioning with the sparse-root zone model, you can override the default location of the Deployment Engine (DE) to troubleshoot the installation.

When the Web GUI is installed on a UNIX operating system using the root user account, the DE is installed in the `/usr/ibm/common/acsi` directory. However, on Solaris operating systems, when Solaris zones partitioning technology with the sparse-root zone model is used, the root user does not have write access to the `/usr` directory.

To solve this problem, start the Web GUI installer from the command-line interface and override the default installation location for the DE.

To override the default DE location:

- In GUI mode: Enter the following command:
`/install.sh -DIAGLOBAL_DE_INSTALL_LOCATION=/location`
Where *location* is a writable location for the root user.
- In console mode: Enter the following command:
`/install.sh -DIAGLOBAL_DE_INSTALL_LOCATION=/location -i console`
Where *location* is a writable location for the root user.
- In silent mode:
 1. After you have edited the `sample_response.txt` file to specify your installation parameters, append the following line:
`IAGLOBAL_DE_INSTALL_LOCATION=location`
Where *location* is a writable location for the root user.
 2. Enter the following command:
`./install.sh -f full-path-to-response-file/sample_response.txt`

Troubleshooting a failed uninstallation on Windows

On Windows operating systems, if the uninstallation of the Web GUI fails, Windows services might be left behind on the server. These services must be removed before you can reattempt the installation of the Web GUI, or before you can install any other Tivoli product that is based on Tivoli Integrated Portal.

You can verify the success of the uninstallation process by checking the Windows services that are installed on the server. To access the services, open the Control Panel and click **Administrative Tools > Services**. If the uninstallation process failed, the Tivoli Integrated Portal Windows service is still displayed in the Services window as **Tivoli Integrated Portal - TIPProfile_Port_port**, where *port* is the port number on which the Web GUI was installed. Unless you troubleshoot the uninstallation, any subsequent installation on that port of a product based on Tivoli Integrated Portal will fail.

To troubleshoot the failed uninstallation:

- Reinstall the Web GUI, or install the required Tivoli Integrated Portal product on a different port than the port specified for the uninstalled Web GUI.
- To remove the Windows service left behind by the uninstallation process:
 1. Change to the `install_dir/tip/bin` directory.
 2. To stop the Windows service, enter the following command:

```
WASService -stop TIPProfile_Port_port
```

Where *port* is the port number on which the Web GUI was installed.

3. To remove the Windows service, enter the following command:

```
WASService -remove TIPProfile_Port_port
```

Where *port* is the port number on which the Web GUI was installed.

Related tasks

“Uninstalling the Web GUI” on page 214

Troubleshooting user registries

If, after installation, you cannot log in through the user registry that you specified, disable the login feature and then modify the Web GUI configuration settings for the registry.

To disable the login:

1. Back up the following file: *install_dir*/profiles/TIPProfile/config/cells/TIPCell/security.xml.

2. On the server, edit the security.xml file (not the backup copy) by setting the first enabled attribute to false, as shown in the following example:

```
<security:Security xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:orb.securityprotocol=
"http://www.ibm.com/websphere/appserver/schemas/5.0/
orb.securityprotocol.xmi"
xmlns:security="http://www.ibm.com/websphere/appserver/schemas/5.0/
security.xmi"
xmi:id="Security_1" useLocalSecurityServer="true"
useDomainQualifiedUserNames="false"
enabled="false" cacheTimeout="600" issuePermissionWarning="true"
activeProtocol="BOTH"
enforceJava2Security="false" enforceFineGrainedJCASecurity="false"
appEnabled="true"
dynamicallyUpdateSSLConfig="true" activeAuthMechanism="LTPA_1"
activeUserRegistry="WIMUserRegistry_1"
defaultSSLSettings="SSLConfig_TIPNode_1">
```

3. Restart the server.

What to do next

Check, and if necessary change the settings for your user registry. If required, change the user registry.

Related tasks

“Adding an external LDAP repository” on page 481

“Changing user registries after installation” on page 477

“Restarting the server” on page 568

Installation fails at step “Integrated Solutions Console”

If you attempt to install the Web GUI on a host whose name contains an underscore (_), the installation fails at the step “Integrated Solutions Console.” This failure occurs on all operating systems.

In this instance, the following example shows the messages that are recorded in the TIPProfile_create.log file:

```
<record>
<date>2009-01-20T00:32:16</date>
<millis>1232382736455</millis>
<sequence>978</sequence>
<logger>com.ibm.ws.profile.validators.GenericValidator</logger>
<level>SEVERE</level>
<class>com.ibm.ws.profile.validators.GenericValidator</class>
<method>getErrorOutput</method>
<thread>10</thread>
<message>Returning error message:{0} is not a properly formed host name.</message>
</record>
<record>
<date>2009-01-20T00:32:16</date>
<millis>1232382736455</millis>
<sequence>979</sequence>
<logger>com.ibm.ws.install.configmanager.actionengine.IJCAction</logger>
<level>INFO</level>
<class>com.ibm.ws.install.configmanager.actionengine.IJCAction</class>
<method>executeAction</method>
<thread>10</thread>
<message>Result of executing /opt/netcool/tip/profileTemplates/default/validators/
hostnameValidator.ijc was: false</message>
</record>
```

To solve this problem, remove the underscore from the host name and reattempt the installation, or install the product on a host whose name does not contain an underscore.

Installation failure scenario

Review the IA-TIPInstall-xx.log for any errors that might have occurred during installation.

IA-TIPInstall-xx.log

Typically, the installation process stops when a failure occurs. But it can also appear to complete successfully and then later, such as when attempting to log in, you find that there is a problem. Review the IA-TIPInstall-xx.log in your home directory to confirm that the installation was successful. For example, if you are logged in as Administrator on a Windows system, then you would look in C:\Documents and Settings\Administrator.

Log review scenario

In this example on a Windows system, the ESSServerConfig.xml step failed and IA-TIPInstall-xx.log as shown here appears to have a COI (Composite Offering Installer) failure at line 134.

```
C:\IBM\tivoli\tip\_uninst\ITNM\plan\install\MachinePlan_localhost\
0011_IAGLOBAL_COI_STEP_ESSServerConfig\IAGLOBAL_COI_STEP_ESSServerConfig.xml:134:
xec returned: 105
Wed May 28 15:25:54.078 EDT 2008 : STDERR :
at org.apache.tools.ant.ProjectHelper.
addLocationToBuildException(ProjectHelper.java:539)
```

```

Wed May 28 15:25:54.078 EDT 2008 : STDERR :
at org.apache.tools.ant.taskdefs.Ant.
execute(Ant.java:384)
Wed May 28 15:25:54.078 EDT 2008 : STDERR :
at org.apache.tools.ant.Task.perform
(Task.java:364)
Wed May 28 15:25:54.078 EDT 2008 : STDERR :
at com.ibm.ac.coi.impl.utils.
AntHelper.ant(AntHelper.java:88)
Wed May 28 15:25:54.078 EDT 2008 : STDERR : ... 3 more

```

The log provides you with the full path to the location of the failing file. Navigate to that location, open the file indicated, and check the line that failed. In this example you would navigate to:

```

C:\IBM\tivoli\tip\_uninst\ITNM\plan\install\MachinePlan_localhost\
00011_IAGLOBAL_COI_STEP_ESSServerConfig\IAGLOBAL_COI_STEP_ESSServerConfig.xml

```

and study line 134. At line 134 of target configureESS, the following command did not execute successfully

```

<target name="configureESS" depends="setProperties">
    <echo message="Start to configure Authentication Service..."/>
    <iaecho message="$ESSSERVER_CONFIGURING$"/>
    .....
line134: <exec
dir="{IAGLOBAL_installLocation}/bin"
executable="{IAGLOBAL_installLocation}/bin/wsadmin${platform.script.ext}"
failonerror="true">
    <redirector output="{IAGLOBAL_installLocation}/logs/
ESSConfiguration.out" error="{IAGLOBAL_installLocation}/logs
/ESSConfiguration.err"/>
...

```

As you can see, the wsadmin call from Ant sends stdout to *tip_home_dir/logs/ESSConfiguration.out* and stderr to *tip_home_dir/logs/ESSConfiguration.err*. A review of the ESSConfiguration.out file shows that the Tivoli Integrated Portal Server (WAS) might have a problem:

```

WASX7209I: Connected to process "server1" on node TIPNode using SOAP connector;
The type of process is: UnManagedProcess
WASX7303I: The following options are passed to the scripting environment and
are available as arguments that are stored in the argv variable:
"[C:/IBM/tivoli/tip/logs/ltpaOutput.txt, Integrate]"
WASX7017E: Exception received while running file "C:\IBM\tivoli\tip\bin
\configureESS.jacl";
exception information: com.ibm.bsf.BSFException: error while eval'ing
Jacl expression:
no accessible method "isESSConfigured" in class
com.ibm.ws.scripting.adminCommand.AdminTask
while executing
"$AdminTask isESSConfigured"
invoked from within
"set essCheck [$AdminTask isESSConfigured]"

```

Check the *tip_home_dir/profiles/TIPProfile/logs/server1/SystemOut.log* for any exceptions that might be related to the Authentication Service. If you are not able to assess this, ask the resident Tivoli Integrated Portal Server expert or gather the Tivoli Netcool/OMNIBus Web GUI logs, including SystemOut.log, and contact IBM Support.

Install fails after deployment engine upgrade

Running the installer on a computer that has an existing Tivoli Integrated Portal environment can fail if the deployment engine (DE) was upgraded from a very early version.

If you have an old version of the DE installed, the Tivoli Integrated Portal installer will upgrade it and continue with the installation. On rare occasions certain older versions of the DE might not be upgraded successfully. When this happens, the installation can fail. If you are aware that your product uses a very old version of the DE (such as Version 1.2), you can install on the same machine, but sign on to the portal with a different user name. If your old version of the DE was initially installed as root user on the Linux or UNIX operating system, consider uninstalling it if your new installation is failing after the DE upgrade.

Migration fails with "Out of Memory" errors

How to recover from an Out of memory error during migration.

The import phase of a migration may fail if the Java Virtual Machine (JVM) has insufficient memory, and Out of memory errors appear in the log files for the migration. If this occurs, increase the amount of memory for the JVM, roll back the migration, and then repeat the import phase.

Related tasks

"Migrating from IBM Tivoli Netcool/Webtop versions 2.0 or 2.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI" on page 194

"Migrating from IBM Tivoli Netcool/Webtop version 1.3.1 to the Tivoli Netcool/OMNIbus version 7.3.1 Web GUI" on page 199

Migration from Netcool/Webtop V2.1 or earlier with a large number of users fails

Migration of a Netcoll/Webtop installation that has a large number of users fails with the following error appearing in the dci-security.log file: CWWIM1018E 'nnnn' search results exceeds the '4500 maximum search limit.

1. Ensure that the **users.reload.mode** property in the **server.init** file has a value of 1.
If the property has any other value:
 - a. Set the property's value to 1.
 - b. Restart the server.
 - c. Repeat the migration.
2. If the same error occurs:
 - a. Navigate to *tip_home_dir/profiles/TIPProfile/config/cels/TIPCell/wim/config*
 - b. Make a backup copy of the *wimconfig.xml* file.
 - c. Edit the *wimconfig.xml* file and locate the attribute *maxSearchResults*.
 - d. Change the value of the attribute to a large number, for example 10000 and save the file.
 - e. Restart the server and repeat the migration.

Related tasks

“Migrating from IBM Tivoli Netcool/Webtop versions 2.0 or 2.1 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI” on page 194

“Migrating from IBM Tivoli Netcool/Webtop version 1.3.1 to the Tivoli Netcool/OMNIBus version 7.3.1 Web GUI” on page 199

“Restarting the server” on page 568

Directory structure

During installation of the Web GUI, packages are installed in various locations. Additionally, the Deployment Engine (DE) creates files that are used during the installation; the location of these files depends on the operating system and the user that installed the product.

The following table describes the directory structure of the Web GUI.

Windows Replace the forward slash (/) with backward slash (\).

Table 65. Web GUI directories

Directory	Description
<code>tip_home_dir/bin</code>	Contains the scripts for starting and stopping the Tivoli Integrated Portal server.
<code>webgui_home_dir/etc</code>	Contains Web GUI configuration files, including the user-configurable file <code>server.init</code> .
<code>webgui_home_dir/etc/cgi</code>	Contains configuration options for the CGI Registry.
<code>webgui_home_dir/etc/cgi-bin</code>	Contains CGI scripts.
<code>webgui_home_dir/etc/charts</code>	Contains chart XML files.
<code>webgui_home_dir/etc/configstore</code>	Contains configuration files for menus, tools, prompts, metrics, and filter collections.
<code>webgui_home_dir/etc/data</code>	Contains global and user-specific filter and view definitions.
<code>webgui_home_dir/etc/datasources</code>	Contains information about Web GUI data sources.
<code>webgui_home_dir/etc/default</code>	Contains a copy of Web GUI default configuration files.
<code>webgui_home_dir/etc/deprecated</code>	For upgraded installations of IBM Tivoli Netcool/Webtop V2.2 only: Contains migrated Netcool/Webtop configuration artifacts, for example entities.
<code>webgui_home_dir/etc/maps</code>	Contains map files.
<code>webgui_home_dir/etc/resources</code>	Contains resources used in maps.
<code>webgui_home_dir/etc/system</code>	Contains system HTML files and resources.
<code>webgui_home_dir/etc/templates</code>	Contains system HTML templates.
<code>tip_home_dir/profiles/TIPProfile/installed Apps//TIPCell/isc.ear/OMNIBusWebGUI.war</code>	Contains the Web GUI Web application files.
<code>tip_home_dir/profiles/TIPProfile/logs</code>	Contains log files for applications.

Table 65. Web GUI directories (continued)

Directory	Description
<i>tip_home_dir</i> /java/jre	Contains the Java Runtime Environment (JRE).
<i>webgui_home_dir</i>	Contains files specific to Web GUI.
<i>webgui_home_dir</i> /bin	Contains Web GUI scripts.
<i>webgui_home_dir</i> /integration/migration_tool	Contains files for the Web GUI migration utility.
<i>webgui_home_dir</i> /waapi	Contains WAAPI files.
<i>tip_home_dir</i> Components/ESSServer	Contains the Embedded Security Services (ESS) component used to manage single sign-on between the Tivoli Integrated Portal and non-WebSphere applications such as TADDM. The name of the directory is built from the name of the Tivoli Integrated Portal installation directory and the word "Components".

The following table describes the location of the directories created by the DE during installation.

Table 66. Deployment Engine (DE) directories

Operating system and user	Location
UNIX non-root installation	/home/username/.acsi_machinename
UNIX root installation	/usr/ibm/common/acsi
Windows administrator	C:\Program Files\IBM\Common\acsi

The DE can be updated if another IBM product is installed on the same host. This does not affect the operation of any existing installations.

Linux **UNIX** If you are installing as a non-root user, the DE creates files in the home directory of the user performing the installation. If the home directory is on a shared network drive and the home directory is shared by different operating systems (for example, AIX and Linux), the user cannot install on a host that shares the network drive with any host using a different operating system that already has a product using the Tivoli Integrated Portal framework installed.

Chapter 7. Setting up the Tivoli Netcool/OMNIbus system

After installing Tivoli Netcool/OMNIbus, you can create and set up one or more ObjectServers and configure communications information for your Tivoli Netcool/OMNIbus server components.

Creating and running the ObjectServer

Each Tivoli Netcool/OMNIbus installation must have at least one ObjectServer to store and manage alert information. You can also set up multiple ObjectServers on one or more host computers.

ObjectServer overview

The ObjectServer is the database server at the core of Tivoli Netcool/OMNIbus. Event information is forwarded to the ObjectServer from external programs such as probes and gateways. The ObjectServer stores and manages this information in database tables, and displays the information in the event list.

In a standard configuration, events are forwarded directly to the ObjectServer. You can use the proxy server to reduce the number of probe connections to an ObjectServer.

You can run the ObjectServer and proxy server in secure mode. In this mode, the ObjectServer and proxy server authenticate connection requests from probes, gateways, and proxy servers by requiring a user name and password.

The ObjectServer supports persistence of data by using disk-based checkpoints and logs. Checkpoints write all data to disk at system-defined intervals to enable data recovery if the server stops unexpectedly. Between checkpoints, additional modifications to the database are logged to disk.

ObjectServer database initialization

To create a new ObjectServer, you must run the database initialization utility (**nco_dbinit**).

The **nco_dbinit** utility performs the following functions:

- Creates a properties file for the new ObjectServer
The properties file contains the settings for configuring the ObjectServer. The properties file is called `$NCHOME/omnibus/etc/servername.props`, where *servername* is the name that you specify when creating the ObjectServer.
- Creates the default database tables and data for the new ObjectServer
- Creates default users, groups, and roles for the new ObjectServer
The default ObjectServer root and nobody users are created, along with default groups and roles to help you manage permissions.

The **nco_dbinit** utility uses a number of database initialization files to create the default data.

Database initialization files

The database initialization utility (nco_dbinit) uses SQL files to create the default database tables and data for a new ObjectServer.

These SQL files are as follows:

- application.sql: This file creates the initial tables for the alerts and tools databases.
- automation.sql: This file creates the initial trigger groups, triggers, and procedures.
- desktop.sql: This file specifies initial values for the desktop tables, including default colors, conversions, tools, and menus.
- system.sql: This file specifies the security database and tables, and system users, groups, roles, and permissions. You must not edit this file.
- security.sql: This file specifies additional operator and administrator roles.

These files are held in the \$NCHOME/omnibus/etc directory.

ObjectServer database directory

The ObjectServer database tables are stored in a default directory.

The ObjectServer uses \$NCHOME/omnibus/db on UNIX systems and %NCHOME%\omnibus\db on Windows systems.

Naming conventions for ObjectServers

When an installation includes more than one ObjectServer, each ObjectServer must have a unique name. The name is used throughout the configuration to identify the ObjectServer.

Probes, gateways, and desktop clients use the name of the ObjectServer to connect to it.

The name of an ObjectServer must consist of 29 or fewer uppercase letters and cannot begin with an integer.

Do not use a name that ends with _PA, _GATE, or _PROXY. These strings are reserved to identify a process agent (_PA), a gateway (_GATE), or a proxy server (_PROXY).

Configuring automated failover and failback

Tivoli Netcool/OMNIBus supports automated failover and failback of ObjectServers. In this mode, a backup ObjectServer holds a replica of the event data and automatically takes over all operations of the primary ObjectServer, such as automation control, when the primary ObjectServer goes down.

Failover occurs when applications connect to the backup ObjectServer when they lose connection to the primary ObjectServer. The failback function enables applications to reconnect to the primary ObjectServer when it becomes active again.

A bidirectional ObjectServer Gateway is used to replicate the event data between the primary and backup ObjectServers. The gateway has active connections to both ObjectServers and is aware of the status of each ObjectServer. This gateway uses signal notifications to inform each ObjectServer in the failover pair setup, when its

counterpart fails or restarts. Messages are also written to the ObjectServer log file when the signals are raised. The message logging level on failover is ERROR, whereas for failback, the level is INFO.

Automated failover and failback is supported with the following signals, procedures, and triggers:

- Signals: `gw_counterpart_down` and `gw_counterpart_up`
- Procedures: `automation_disable` and `automation_enable`
- Triggers: `backup_startup`, `backup_counterpart_down`, and `backup_counterpart_up`. These triggers are supplied as disabled in the `$NCHOME/omnibus/etc/automation.sql` file, which is one of the database initialization files.

To enable automated failover and failback, the `backup_startup`, `backup_counterpart_down`, and `backup_counterpart_up` triggers in the `$NCHOME/omnibus/etc/automation.sql` file must be enabled either before or after running `nco_dbinit` to create the ObjectServers. You can enable the triggers before running `nco_dbinit` by editing the `automation.sql` file. You can enable the triggers in an existing ObjectServer by using the `ALTER TRIGGER` command, or by using Netcool/OMNIBus Administrator. For further information about enabling triggers, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Note: If you configure your primary and backup ObjectServers to run as a virtual pair in the aggregation layer of the standard multitiered architecture, the `backup_startup`, `backup_counterpart_down`, and `backup_counterpart_up` triggers are automatically enabled as part of the configuration.

Related concepts

“Database initialization files” on page 230

Chapter 8, “Configuring and deploying a multitiered architecture,” on page 251

Creating an ObjectServer

You create one or more ObjectServers on a host workstation by running the database initialization utility (`nco_dbinit`).

Important notes for multicultural support:

- You must run `nco_dbinit` in the locale in which you are going to start and run the ObjectServer.
- If you intend to run the ObjectServer in UTF-8 encoding on Windows (which also ensures GB18030 compliance), run the `nco_dbinit` utility with the `-utf8enabled` command-line option set to `TRUE`.
- When creating an ObjectServer, you can set it up so that configuration data for the UNIX and Windows desktop is displayed in your locale. This configuration data is typically displayed in the event list and Conductor, and includes default colors, column visuals, conversions, tools, and menus. When you run `nco_dbinit`, the configuration data is read from the default desktop SQL definition file, but you can specify a different file for your locale by using the `-desktopfile` command-line option with the `nco_dbinit` command.

To create a new ObjectServer:

From the command line, enter the appropriate command for your operating system:

Option	Description
UNIX	<code>\$NCHOME/omnibus/bin/nco_dbinit -server <i>servername</i></code>
Windows	<code>%NCHOME%\omnibus\bin\nco_dbinit -server <i>servername</i></code>

In this command, *servername* is the name of the new ObjectServer that you want to create. Additional command-line options and properties are available for the **nco_dbinit** utility.

Results

The properties file `$NCHOME/omnibus/etc/servername.props` is created for the new ObjectServer. Additionally, the default database tables, data, users, groups, and roles are created for the ObjectServer. You can create additional ObjectServer objects by using Netcool/OMNIBus Administrator or ObjectServer SQL.

Related concepts

“Setting your locale” on page 403

Related tasks

“Setting up the ObjectServer to use translated user interface text in the desktop” on page 410

Properties and command-line options for nco_dbinit

When the database initialization utility **nco_dbinit** starts, it reads a properties file. If a property is not specified in this file, the default value is used, unless you override it with a command-line option.

The default location of the properties file is `$NCHOME/omnibus/etc/nco_dbinit.props`.

The properties and command-line options that **nco_dbinit** supports are described in the following table.

Table 67. Properties and command-line options for nco_dbinit

Property	Command-line option	Description
AlertsData TRUE FALSE	<code>-alertsdata</code>	Controls whether the file specified by the AlertsDataFile property is read. The default is FALSE; that is, the file is not read.

Table 67. Properties and command-line options for `nco_dbinit` (continued)

Property	Command-line option	Description
AlertsDataFile <i>string</i>	<code>-alertsdatafile</code> <i>string</i>	<p>An additional application SQL definition file that is read after the file specified by the ApplicationFile property. If you want this file to be read, you must set the AlertsData property to TRUE. The default file is <code>\$NCHOME/etc/alertsdata.sql</code>.</p> <p>INSERT statements that contain fields of type INCR are processed differently in this file than in the file specified by the ApplicationFile property. In this file, such fields are <i>not</i> incremented, but are assigned the values given in the INSERT statements.</p> <p>For example, the alerts.journal table contains a foreign key that is based on the Serial field of the alerts.status table. To maintain this reference, values for the Serial field must be inserted without changes.</p> <p>Use this file only for data that is created by automated tools, for example, the nco_osreport utility.</p>
ApplicationFile <i>string</i>	<code>-applicationfile</code> <i>string</i>	The application SQL definition file, which creates the default tables for the alerts and tools databases. The default file is <code>\$NCHOME/omnibus/etc/application.sql</code> .
AutomationFile <i>string</i>	<code>-automationfile</code> <i>string</i>	The automation SQL definition file, which creates the default triggers and trigger groups. The default file is <code>\$NCHOME/omnibus/etc/automation.sql</code> .
CopyPropsFile TRUE FALSE	<code>-nopropsfile</code>	<p>Determines whether a new properties file is created for the target ObjectServer. If the <code>-nopropsfile</code> command-line option is specified or the CopyPropsFile property is set to FALSE, the default properties file is not copied for the target ObjectServer.</p> <p>The default is TRUE; that is, the default properties file is copied and renamed for the new ObjectServer.</p>
CustomConfigFile <i>string</i>	<code>-customconfigfile</code> <i>string</i>	Specifies a comma-separated list of paths to SQL import files (.sql) that can be used to update the database schema with additional configuration when the ObjectServer is created.
DesktopFile <i>string</i>	<code>-desktopfile</code> <i>string</i>	The desktop SQL definition file, which inserts default values into the desktop tables, including default colors, conversions, tools, and menus. The default file is <code>\$NCHOME/omnibus/etc/desktop.sql</code> .

Table 67. Properties and command-line options for `nco_dbinit` (continued)

Property	Command-line option	Description
DesktopPrimaryServer <i>string</i>	<code>-dsdprimary</code> <i>string</i>	Indicates the name of the primary ObjectServer in a dual-server architecture. This value is entered in the MasterServer field of the master.national table. If the DesktopServer property is not set to TRUE, this property is ignored.
DesktopServer TRUE FALSE	<code>-desktopserver</code>	Indicates that the ObjectServer should be created as a desktop ObjectServer.
DesktopServerDualWrite 0 1	<code>-dsddualwrite</code>	Indicates that the desktop ObjectServer should be created with dual-write mode enabled. This value is entered in the DualWrite field of the master.national table. If the DesktopServer property is not set to TRUE, this property is ignored.
DesktopServerFile <i>string</i>	<code>-desktopserverfile</code> <i>string</i>	Configures the ObjectServer as a desktop ObjectServer using this SQL definition file, which creates the master.national table for the desktop server and the corresponding MasterSerial column in the alerts.status table. The default file is \$NCHOME/omnibus/etc/desktopserver.sql.
Force TRUE FALSE	<code>-force</code>	Forces existing database files to be overwritten. Attention: All modifications are lost.
N/A	<code>-help</code>	Displays help on the command-line options and exits.
Memstore.DataDirectory <i>string</i>	<code>-memstoredatadirectory</code> <i>string</i>	Specifies the path for the ObjectServer to use for database files. The default is \$NCHOME/omnibus/db.
MessageLevel <i>string</i>	<code>-messagelevel</code> <i>string</i>	Specifies the message logging level. Possible values are: fatal, error, warn, info, and debug. The default level is info. Messages that are logged at each level are as follows: <ul style="list-style-type: none"> fatal : fatal only. error: fatal and error. warn: fatal, error, and warn. info: fatal, error, warn, and info. debug: fatal, error, warn, info, and debug. Tip: The value of <i>string</i> can be in uppercase, lowercase, or mixed case.
MessageLog <i>string</i>	<code>-messagelog</code> <i>string</i>	Specifies where messages are logged. The default is stderr.
N/A	<code>-propsfile</code> <i>string</i>	Specifies the path to the nco_dbinit properties file. The default is \$NCHOME/omnibus/etc/nco_dbinit.props.

Table 67. Properties and command-line options for `nco_dbinit` (continued)

Property	Command-line option	Description
ObjectServerPropsFile <i>string</i>	<code>-objservpropsfile</code> <i>string</i>	Specifies the path to the source ObjectServer properties file. The default is <code>\$NCHOME/omnibus/etc/initial/NCOMS.props</code> .
Props.CheckNames TRUE FALSE	N/A	When TRUE, <code>nco_dbinit</code> does not run if any specified property is invalid. The default is TRUE.
RestrictPasswords TRUE FALSE	<code>-restrictpasswords</code> TRUE FALSE	When TRUE, passwords must conform to the following restrictions: <ul style="list-style-type: none"> • The password must consist of at least eight characters. • The password must contain at least one numeric character. • The password must contain at least one alphabetic character. The default is FALSE.
Sec.AuditLevel <i>string</i>	<code>-secauditlevel</code> <i>string</i>	Specifies the security audit level. The default is warn.
Sec.AuditLog <i>string</i>	<code>-secauditlog</code> <i>string</i>	Specifies the location to log the security audit trail. The default is stdout.
SecurityFile <i>string</i>	<code>-securityfile</code> <i>string</i>	Specifies the path to the security definition file, which defines the operator and administrator roles. The default is <code>\$NCHOME/omnibus/etc/security.sql</code> .
Server <i>string</i>	<code>-server</code> <i>string</i>	Specifies the name of ObjectServer to initialize. The default is NCOMS.
SystemFile <i>string</i>	<code>-systemfile</code> <i>string</i>	Specifies security database and tables, and system users, groups, roles, and permissions. The default is <code>\$NCHOME/omnibus/etc/system.sql</code> . Attention: Do not modify this file.
N/A	<div>Windows</div> <code>-utf8enabled</code> TRUE FALSE	Controls the encoding of data that is passed into, or generated by, this application on Windows. Set the value of <code>-utf8enabled</code> to TRUE to run the application in the UTF-8 encoding. The default is FALSE, which causes the default system code page to be used. Important: Although a <code>UTF8Enabled</code> property is available, an attempt to enable UTF-8 encoding by setting this property to TRUE will have no effect. To run in a UTF-8 encoding on Windows, you must always use the <code>-utf8enabled</code> command-line option.
N/A	<code>-version</code>	Displays version information for <code>nco_dbinit</code> and exits.

Note: The `nco_dbinit` utility includes advanced properties that must be used only under direction from IBM Software Support. These properties are not documented.

After creating an ObjectServer

After you have created a new ObjectServer, you must use the Server Editor to add the communication details for the ObjectServer on the host machine and on every machine that needs to connect to the ObjectServer.

You can start the Server Editor as follows:

- **UNIX** **Linux** Enter `$NCHOME/omnibus/bin/nco_xigen` on the command line.
- **Windows** Click **Start > Programs > Netcool Suite > System Utilities > Servers Editor**.

After you have added the communication details in the Server Editor, you can then start the ObjectServer.

Additionally, you can perform the following tasks:

- Modify ObjectServer objects and data by using Netcool/OMNIBus Administrator or the `nco_sql` utility.
- Modify property settings by using Netcool/OMNIBus Administrator or the `ALTER SYSTEM SET SQL` command.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Related tasks

“Starting an ObjectServer”

Starting an ObjectServer

You must have an ObjectServer running before you can use the components of Tivoli Netcool/OMNIBus.

You can start an ObjectServer:

- Automatically, using process control on UNIX and Windows
- Automatically, using services on Windows
- Manually, from the command line

Starting an ObjectServer by using process control

If started by the process agent, the ObjectServer automatically restarts if it fails. By starting the process agent when the system starts, you can make the ObjectServer start automatically on either UNIX or Windows.

An ObjectServer can be started as a process, by using the process agent. The ObjectServer must be defined as a process or part of a service.

You can start the ObjectServer from a remote computer. The name that you specify with the `-server` option is compared to the process agent names that are configured in the Server Editor. The host computer and port are identified and the command is sent to the correct process agent.

To start an ObjectServer as a process, enter the following command:

```
nco_pa_start -process ObjectServer
```

In this example, the ObjectServer has been defined as a process called ObjectServer.

For information about process control and process agents, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Starting an ObjectServer by using services (Windows)

On Windows, you can optionally install and run the ObjectServer as a Windows service. When the service is set to automatic, the ObjectServer starts when the computer starts up.

If you have installed the ObjectServer as a Windows service, you can start the service either from the Services window in the Control Panel, or from the command line.

To start the ObjectServer from the Services window, perform the following actions:

1. Open the Windows Control Panel.
2. Double-click **Administrative Tools** and double-click the **Services** icon.
3. Double-click **Netcool/OMNIBus Object Server**. The properties window for this service opens.
4. In the **Start parameters** field on the **General** tab, type the command-line options for the local ObjectServer. For example, enter:
-name NCOMS
5. Ensure that the value in the **Startup type** field is set to Automatic.
6. Click the **Start** button to start the ObjectServer as a Windows service. When the service starts, click **OK** to close the properties window.

Example

From the command line, you can start the ObjectServer service by running the following command:

```
net start service_name
```

Where *service_name* is the service name of the ObjectServer, as defined in the Services window; for example, **NCOObjectServer**.

Related tasks

“Setting up Tivoli Netcool/OMNIBus components as Windows services” on page 161

Starting an ObjectServer manually

Use the **nco_objserv** command to start the ObjectServer manually.

To start an ObjectServer:

From the command line, enter the appropriate command for your operating system:

Operating system	Command
UNIX	<code>\$NCHOME/omnibus/bin/nco_objserv [-name servername]</code>
Windows	<code>%NCHOME%\omnibus\bin\nco_objserv [-name servername]</code>

In this command, *servername* is the ObjectServer name. If you do not specify the -name command-line option, **nco_objserv** attempts to start the NCOMS

ObjectServer. You can start the ObjectServer with additional command-line options. For details of these command-line options, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Note: An ObjectServer that is started from the command line is not under process control, and must be restarted manually if it is shut down. On startup, the ObjectServer attempts to open the `$NCHOME/omnibus/etc/servername.props` properties file, where *servername* is the name of the specified ObjectServer.

Configuring a running ObjectServer

When the ObjectServer is running, you can use ALTER SYSTEM commands and Netcool/OMNIBus Administrator to make changes to the configuration.

For further details, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Stopping an ObjectServer

You can stop an ObjectServer by using process control on UNIX and Windows. If you have set up the ObjectServer as a Windows service, you can stop the ObjectServer by using services on Windows. You can also stop an ObjectServer from the SQL interactive interface.

Stopping an ObjectServer by using process control

On both UNIX and Windows, an ObjectServer can be stopped, as a process, by using the process agent. The ObjectServer must be defined as a process or part of a service.

To stop an ObjectServer as a process, enter the following command:

```
nco_pa_stop -process ObjectServer
```

In this example, the ObjectServer has been defined as a process called ObjectServer.

You can stop the ObjectServer from a remote computer as follows:

```
nco_pa_stop -server NAME_PA -process ObjectServer
```

In this example, the *NAME_PA* value that you specify with the `-server` option is compared to the process agent names that are configured in the Server Editor. The host machine and port are identified, and the command is sent to the correct process agent on a remote computer.

For information about process control and process agents, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Stopping an ObjectServer by using services (Windows)

If you have set up the ObjectServer to run as a Windows service, you can stop the ObjectServer by stopping the service.

To stop an ObjectServer that is running as a Windows service, perform the following actions:

1. Open the Windows Control Panel.
2. Double-click **Administrative Tools** and double-click the **Services** icon.
3. Double-click **Netcool/OMNIbus Object Server**. The properties window for this service opens.
4. From the **General** tab, click the **Stop** button to stop the ObjectServer service.

Example

From the command line, you can stop the ObjectServer service by running the following command:

```
net stop service_name
```

Where *service_name* is the service name of the ObjectServer, as defined in the Services window; for example, **NCOObjectServer**.

Related tasks

“Setting up Tivoli Netcool/OMNIbus components as Windows services” on page 161

Stopping an ObjectServer from the SQL interactive interface

If you manually started an ObjectServer from the command line, you must manually stop the ObjectServer by using the SQL interactive interface. You must have the appropriate permissions to stop the ObjectServer.

To stop an ObjectServer that was started manually, perform the following actions:

1. Connect to an ObjectServer by running the appropriate command for your operating system:

Operating system	Command
UNIX	<code>\$NCHOME/omnibus/bin/nco_sql [-server servername] [-user username]</code>
Windows	<code>%NCHOME%\omnibus\bin\isql -S servername -U username</code>

In these commands, *servername* is the name of a local or remote ObjectServer and *username* is a valid user name. If you do not specify the `-server` command-line option, the SQL interactive interface connects to the NCOMS ObjectServer. If you do not specify a user name, the default is the user running the command.

Tip: On Windows you must specify the ObjectServer name and user name.

2. Provide the requested password.
3. When the SQL prompt is displayed, enter the following commands:
1> alter system shutdown;
2> go

Results

Note: The `nco_sql` command does not allow whitespace preceding the `go` keyword. Any whitespace causes the SQL statements to fail.

If an ObjectServer is started under process control, the process agent restarts it automatically after a manual shutdown. In this case, you must shut down the ObjectServer using process control.

For information about using the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIbus Administration Guide*.

Configuring server communication details in the Server Editor

When you install or change a server component on any host in your Tivoli Netcool/OMNIbus system, you must configure the component to communicate with other components by using the Server Editor.

Use the Server Editor to maintain communication information for the following server components:

- ObjectServers
- Gateways
- Process agents
- Proxy servers

Creating and maintaining server entries after installation

After installation, you must use the Server Editor to update the server communication information on the host machine and on every computer that needs to connect to the server component.

You must continue to maintain the server communication information on the host computer and on every computer that needs to connect to the server component, each time that your system configuration changes.

Default Server Editor entries on UNIX

A default Server Editor entry is created on the host computer for each server as part of the Tivoli Netcool/OMNIbus installation. After installation, the host is set to `omnihost` for all server entries. You must change these `omnihost` settings to the name of the server host computer.

Additionally, you must create a client entry on each computer from which you will connect to the server with values for the host and port that match those on the host computer.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

Related tasks

“Configuring server communication information” on page 242

Default Server Editor entries on Windows

Two default Server Editor entries are created on the host computer for each server as part of the Tivoli Netcool/OMNIBus installation. The Listener responds to client requests. Additionally, a client entry is created so that local clients can connect to the server.

Note: Local connections do not need to be encrypted.

You must create a client entry on each computer from which you will connect to the server with values for the host and port that match those on the host computer.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

Related tasks

“Configuring server communication information” on page 242

Gateway server definition entry

You must create a gateway server definition entry for each gateway running on the current host.

The default name of the gateway server is NCO_GATE. This uses the properties file `NCHOME/omnibus/etc/NCO_GATE.props`.

Each gateway can also be configured to run with its own gateway server.

For more information about gateways, including how to configure and run them, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*, and the individual guides for the gateways that you are using.

Process agent server definition entry

The process agent allows you to use external procedures in the automations system. This means that you can issue commands on other host machines.

The process agent must have a server and client entry in the Server Editor. These entries are automatically created during installation.

The default process agent server is called NCO_PA. The default port number is 4200.

For more information about process control, including how to configure it to manage processes and to run external procedures in automations, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Related tasks

“Configuring server communication information” on page 242

Configuring server communication information

You can use the Server Editor to create and modify communication details, test server activity, and add backup (failover) servers and listeners. You can also delete server definitions when they are no longer part of your system configuration.

To configure server communication using the Server Editor:

1. Perform the appropriate action for your operating system:

Table 68. Starting the Server Editor

Operating system	Action
UNIX	Perform either of the following actions: <ul style="list-style-type: none">• Click the Interfaces button on the UNIX Conductor.• Enter <code>\$NCHOME/omnibus/bin/nco_xigen</code> on the command line.
Windows	Perform either of the following actions: <ul style="list-style-type: none">• Click Start > Programs > Netcool Suite > System Utilities > Servers Editor.• Open the Conductor menu by right-clicking the bus icon in the desktop tray and clicking Servers.

The Server Editor window opens, showing the list of defined servers.

2. Complete this window as follows:

Server list

The server list shows existing server components and their settings:

Server This column displays the name of each server that is defined for this workstation. The default server names are:

- NCOMS: The default name for ObjectServers
- NCO_PROXY: The default name for proxy servers
- NCO_GATE: The default name for gateways
- NCO_PA: The default name for process agents

On Windows, the host workstation for the server must have a *listener* entry, and the host workstation and any other workstation that connects to that server must have a *client* entry.

Backup ObjectServers appear below the primary ObjectServer in the Server Editor. To hide the list of backup ObjectServers, double-click the primary ObjectServer name or server icon. The backup ObjectServers are hidden and the letter C appears in the server icon. Double-click again to display the backup ObjectServers. The server icon returns to normal.

Hostname

This column displays the host name or IP address of the workstation on which the server component is installed.

Note: After installation, the host is set to omnihost for all servers (on UNIX). You must change this to the name of the server host workstation. On Windows, the default is the host name.

Port This column displays the port on which the server component listens for unencrypted connections.

SSL On UNIX, this column displays the port on which the server component listens for encrypted connections. On UNIX systems, a server component can have a standard port, an SSL port, or both, defined.

On Windows, this column displays the word yes for encrypted connections. On Windows systems, the same port number is used for both encrypted and unencrypted connections.

Server entry fields

Use the server entry fields to enter or edit the details of each server component. If you are editing the details of an existing server component, you must first select the relevant row in the server list, so that the details are shown within the server entry fields.

Name Type the name of a new server component or edit the name of an existing component. On Windows, you can also use the drop-down list to select a name. Use the following suffixes when naming components: `_PROXY` for proxy servers, `_GATE` for gateways, and `_PA` for process agents.

Note: The name of a server entry must consist of 29 or fewer uppercase letters and cannot begin with an integer.

Host Type or edit the host name or IP address of the workstation on which the server component is installed. (For new server components on UNIX, the name is set to `omnihost` by default, and must be changed to the actual host name or IP address.)

If you type an IP address, you can specify an IPv4 or IPv6 address. For example:

- 192.168.0.1
- 2094:82a:2a6e:123:503:badd:fe43:f552

Port On UNIX, if you want clients using unencrypted connections to be able to connect to the server, type a valid, unused port number in this field. To disallow unencrypted connections, do not set the port.

On Windows, type a valid, unused port number in this field.

SSL On UNIX, type a valid, unused port number in this field if you want clients using encrypted connections to be able to connect to the server. To disallow encrypted connections, do not set the port.

On Windows, select this check box to indicate that the port accepts encrypted connections from clients that use SSL.

Note: On UNIX systems, a server component can have a standard port, an SSL port, or both ports defined. On Windows systems, the same port number is used for either type of connection.

Listener (Windows only)

Select this check box if this is a listener entry on the host workstation for the server. Clear the **Listener** check box if this is a client entry.

Add Click this button to add new (and unique) server details to the server list.

Remove/Update

This button changes based on whether you have made any changes to the server component that is currently selected. Click **Remove** if you want to remove the server component that is currently selected from the server list. Click **Update** to refresh the server list with updated details of an existing server component.

Test Click this button to test that you can connect to the server that is selected in the server list. The Server Editor attempts to contact the server on the specified host and port. A window shows the result of the test command.

Priority fields

The server priority fields enable you to raise or lower the priority of server components that are configured as failover systems. For example, suppose an ObjectServer called NCOMS_PRI is configured with a backup called NCOMS_BAK. Using the **Raise** or **Lower** buttons, you can raise the priority of NCOMS_BAK to be the primary ObjectServer and NCOMS_PRI to be the backup.

Raise Click this button to raise the priority of the selected server component by one level.

Lower Click this button to lower the priority of the selected server component by one level.

Generate All (UNIX only)

Click this button to generate interfaces files for all UNIX operating systems. When you apply your changes (using the **Apply** button), this generates interfaces files named \$NCHOME/etc/interfaces.*arch*, where *arch* represents individual UNIX platform names; for example, interfaces.hpux11 and interfaces.solaris2.

Show Groups (Windows only)

On Windows, select this check box to group each server by name within the server list, with backup servers and listeners identified following the client entry.

Apply (UNIX only)

Click this button to apply the changes to the interfaces file. Any server components that were added to the server list, removed from the server list, or edited, will be saved to this file.

OK (Windows only)

Click this button to save your changes and close this window.

Close (UNIX)/Cancel (Windows)

Click this button to either close this window after saving your changes with the **Apply** button, or to close this window without saving your changes.

Import (UNIX only)

Click this button to import communication details.

Results

The ObjectServer retrieves its own server definition to identify its host name and port number. It then accepts connection requests made to this location. Probes, gateways, and desktop clients have properties or command line options that

specify the ObjectServer to connect to. You can also specify a backup ObjectServer to be used if the primary ObjectServer is not available.

Related concepts

Chapter 23, “IPv6 configuration,” on page 399

Related tasks

“Adding a backup ObjectServer”

Adding a backup ObjectServer

You can specify a backup ObjectServer for each ObjectServer defined in the Server Editor. If a connection to the primary ObjectServer fails, the clients attempt to connect to the backup ObjectServer.

To add a backup ObjectServer, follow these steps:

1. Create the new backup ObjectServer.
2. Create the server definition entries.
3. Create the backup client entry.
4. Distribute the interfaces file (UNIX only).

Step 1: Creating the new backup ObjectServer

Create the ObjectServer that will act as the backup. The backup ObjectServer must have a unique name and port number, and is usually installed on a different host computer.

For example, if you already have a primary ObjectServer called NCOMS, you could create a new ObjectServer called NCOMS_BAK.

Note: A backup ObjectServer installed on a different host than the primary ObjectServer must run under process control. If not already present, you must also add server definitions for the process agent to both host systems.

You can create more than one backup ObjectServer.

Related tasks

“Creating an ObjectServer” on page 231

Step 2: Creating the server definition entries

From the Server Editor, create a server entry for each backup ObjectServer.

On UNIX, specify values in the **Name**, **Host**, **Port**, and **SSL** fields. Then click **Add** to add the new server, and click **Apply** to apply the changes to the interfaces file.

On Windows, specify values in the **Name**, **Host**, **Port**, **Listener**, and **SSL** fields. Then click **Add** to add the new server.

Related tasks

“Configuring server communication information” on page 242

Step 3: Creating the backup client entry

If the clients cannot connect to the primary ObjectServer, they look for a backup entry. Probes, gateways, and desktops identify the backup ObjectServer by looking for an ObjectServer that matches the primary ObjectServer name.

To create this client entry in the Server Editor:

1. Click the entry for the primary ObjectServer, for example NCOMS.

Note: *Do not* change the ObjectServer name.

2. Enter the host name for the backup ObjectServer. For example, if you created a backup ObjectServer called NCOMS_BAK, specify the host machine on which NCOMS_BAK is running.
3. Enter the port number for the backup ObjectServer. For example, if you created a backup ObjectServer called NCOMS_BAK, specify the port number for NCOMS_BAK.
4. Click the **Add** button.
5. On UNIX systems, click the **Apply** button to apply the changes to the interfaces file.

Results

The backup ObjectServer is shown as indented below the primary ObjectServer, and displays the host and port details of the backup ObjectServer.

Related tasks

“Configuring server communication information” on page 242

Step 4: Distributing the interfaces file (UNIX only)

The backup ObjectServer usually runs on a different host than the primary ObjectServer, although this is not mandatory. You must have server entries on each host for all servers in your configuration.

To distribute the interfaces file, which contains these entries, to all host computers in your configuration, copy the relevant architecture-specific interfaces file to the \$NCHOME/etc directory on each of the host computers.

For example, if you have three Tivoli Netcool/OMNIbus installations on Sun workstations, copy the file \$NCHOME/etc/interfaces.solaris2 to the \$NCHOME/etc directory on each of the Sun workstations.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

Related reference

“Generating the interfaces file on UNIX” on page 379

Changing the priority of servers

If you have one or more backup servers, you might want to change their priority.

To change the priority of servers:

1. From the Server Editor, select a server.
2. Click either the **Raise** or **Lower** button to give the server a higher or lower priority.
3. On UNIX systems, click the **Apply** button to apply the changes to the interfaces file.

Related tasks

“Configuring server communication information” on page 242

Hiding backup ObjectServers in the Server Editor (UNIX only)

Backup ObjectServers are displayed below the primary ObjectServer in the Server Editor.

To hide the list of backup ObjectServers:

1. From the Server Editor, double-click the primary ObjectServer name or server icon. The backup ObjectServers are hidden and the letter C is displayed in the server icon.
2. Double-click again to display the backup ObjectServers. The server icon returns to normal.

Related tasks

“Configuring server communication information” on page 242

Testing a server

To test the availability of a server in the Server Editor, select a server name from the list and click the **Test** button.

The Server Editor attempts to contact the server on the specified host and port. A window shows the result of the test command.

Related tasks

“Configuring server communication information” on page 242

Manually editing the connections data file

The connections data file is used to create the interfaces file for Tivoli Netcool/OMNIbus communications. There might be occasions when you need to edit the connections file directly; for example, on UNIX systems that do not have a graphical interface.

Note: Where possible, you must use the Server Editor to modify connection information rather than editing the connections data file directly.

The connections data file is called:

```
$NCHOME/etc/omni.dat
```

The following is an example omni.dat file:

```
[NCOMS]
{
Primary: sfo 4100
Backup: dfw 4100
```

```

Backup: lax 4100
}
[NCO_PA]
{
Primary: sfo 4200
}
[NCO_GATE]
{
Primary: sfo 4300
}

```

If you need to manually edit the `omni.dat` file, use this format.

Note: Port numbers must be unique for each server entry.

After you edit the connections data file, you must generate the interfaces file. If you do not want to use the Server Editor, you can run the **nco_igen** command-line utility to generate the interfaces file.

Related tasks

“Generating the interfaces file for multiple platforms (UNIX only)” on page 249

“Configuring server communication information” on page 242

Setting up distributed installations

You can run different Tivoli Netcool/OMNIBus components on multiple systems in your network. For example, you can have an ObjectServer running on one computer, a gateway on another, and a proxy server on another.

To create a distributed installation, you must perform the following steps:

1. Install the required Tivoli Netcool/OMNIBus components on each computer.
2. Configure component communications.
3. Distribute the communications information to each Tivoli Netcool/OMNIBus system (UNIX only).

Step 1: Installing Tivoli Netcool/OMNIBus components

Install the required components on the designated computers in your environment.

Tip: You can also create duplicate ObjectServer configurations by using the **nco_confpack** utility, which is used for importing and exporting ObjectServer configurations.

Related concepts

Chapter 11, “Importing and exporting ObjectServer configurations,” on page 301

Step 2: Configuring component communications

After installing the Tivoli Netcool/OMNIBus components on each machine, you must ensure that the components can communicate with each other.

To do this:

1. Configure server communications on *one* UNIX Tivoli Netcool/OMNIBus workstation.
2. Configure server communications on *all* Windows workstations.
3. For UNIX systems only, generate communications information for multiple operating systems, if necessary.

Configuring server communications on the computers

To configure server communications, use the Server Editor.

Related tasks

“Configuring server communication information” on page 242

Generating the interfaces file for multiple platforms (UNIX only)

After using the Server Editor to set up component communications, the communications information is saved in an *interfaces* file.

For example, the Server Editor generates the following interfaces file on a Solaris workstation:

```
$NCHOME/etc/interfaces.solaris2
```

If you are running Tivoli Netcool/OMNIBus components on more than one UNIX platform, you must generate compatible interfaces files before distributing them.

You can use the Server Editor to generate interfaces files for each platform, or you can generate interfaces files from the command line.

To generate interfaces files for all the available platforms, in the Server Editor:

1. Select the **Generate All** check box.
2. Click **Apply**. This generates interfaces files named `$NCHOME/etc/interfaces.arch`, where *arch* is the UNIX platform name.

To generate an interfaces file for a single platform, enter the following command:

```
$NCHOME/bin/nco_igen -arch platform
```

Where *platform* can be:

- solaris2
- hpux11
- aix5
- linux2x86
- linux2s390
- java
- hpux11hpie

For example, to generate an interfaces file for an AIX system, enter the following command:

```
$NCHOME/bin/nco_igen -arch aix5
```

The following file is created:

```
$NCHOME/etc/interfaces.aix5
```

To generate interfaces files for all available UNIX platforms, enter the command:

```
$NCHOME/bin/nco_igen -all
```

This generates an interfaces file named `$NCHOME/etc/interfaces.arch` for each platform, where *arch* is the UNIX platform name.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

Related tasks

“Configuring server communication information” on page 242

Related reference

“Generating the interfaces file on UNIX” on page 379

Step 3: Distributing the interfaces files (UNIX only)

After generating interfaces files for each UNIX operating system in your Tivoli Netcool/OMNIbus system, you can distribute the interfaces files. This enables you to easily duplicate communications settings for every UNIX Tivoli Netcool/OMNIbus computer.

To do this, copy the relevant architecture-specific interfaces file to the \$NCHOME/etc directory on each of the host computers.

For example, if you have three Tivoli Netcool/OMNIbus installations on Sun workstations, copy the file \$NCHOME/etc/interfaces.solaris2 to the \$NCHOME/etc directory on each of the Sun workstations.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

Related reference

“Generating the interfaces file on UNIX” on page 379

Chapter 8. Configuring and deploying a multitiered architecture

Tivoli Netcool/OMNIBus can be deployed in a multitiered configuration to increase performance and event handling capacity. In a multitiered environment, the control of the event flow between ObjectServers must be carefully managed to preserve data integrity and to ensure that race conditions do not occur.

Tivoli Netcool/OMNIBus ships with a set of configurations that offer a baseline configuration for one-, two-, and three-tiered systems. You can use these configurations to rapidly deploy a multitiered architecture without issue, and in a standardized way. The set of configurations is based on the *standard architecture* from the Event Services Framework (ESF) that was previously released by the IBM Tivoli Netcool Advanced Architecture Group. The multitiered configuration release that is supplied with Tivoli Netcool/OMNIBus contains only the components of the ESF that relate to event flow control, data integrity, and performance.

Important: To ensure a smooth and trouble-free deployment, read through all the information provided in its entirety to familiarize yourself with the concepts before attempting to configure and deploy a multitiered architecture.

Before you begin

Before installing and deploying a multitiered architecture, read this information to understand how to set up the standard architecture, the naming conventions for the components in the architecture, component resourcing considerations, severity handling, and the location of the configuration files.

Overview of the standard multitiered architecture

To minimize the impact of computer failure, it is good practice to use more than one computer in the standard multitiered architecture. Any of the components can, however, be installed and run on any computer, and all the components can even be configured to run on a single computer.

The following figure shows the standard multitiered architecture. The components in the architecture sit within three tiers (or layers): collection, aggregation, and display. Each layer shows the physical computers on which the ObjectServers and associated ObjectServer Gateways reside.

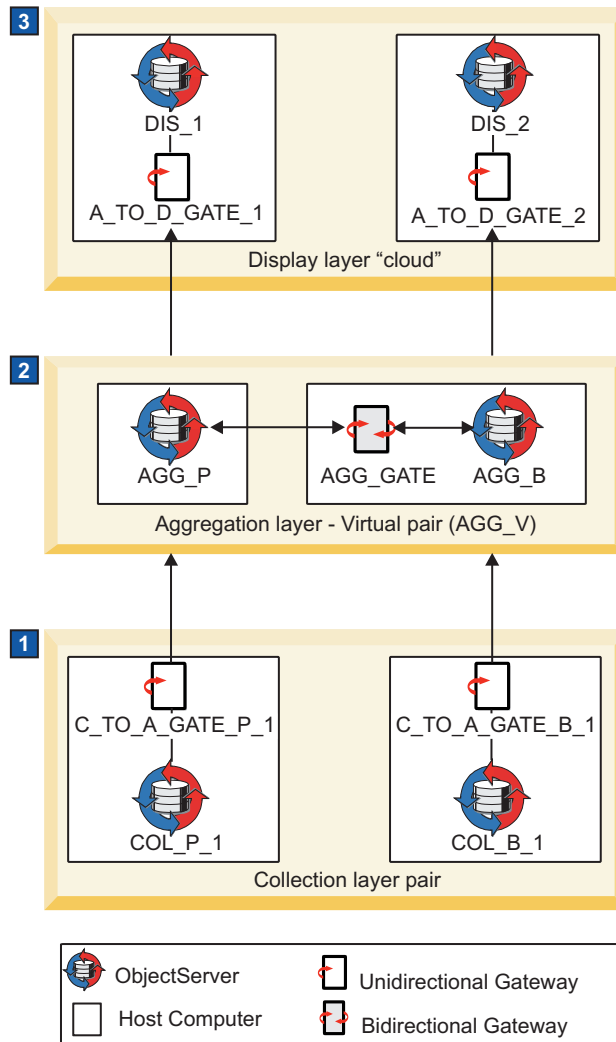


Figure 4. Standard multitiered architecture

In the figure, the unidirectional gateways transfer data in the direction of the arrow. The end of the gateway that connects to the source ObjectServer is known as the reader because it reads data from the source. The end of the gateway that connects to the destination is known as the writer because it writes data to the destination. The bidirectional gateway in the aggregation layer has a reader and a writer at each end because data is flowing in both directions.

1 Collection layer

The collection layer includes a primary and backup pair of ObjectServers to which probes connect. The configuration shows one pair of collection layer ObjectServers, but further pairs can be added if required. (Details about how to configure additional pairs of collection layer ObjectServers are included as an extension of the standard multitiered architecture - see the related links at the end of this topic for further information.)

Each collection layer ObjectServer has its own dedicated unidirectional ObjectServer Gateway that connects the ObjectServer to the aggregation layer. Each collection gateway reader connects, and is fixed to, its dedicated collection ObjectServer, whereas each gateway's writer connects to the *virtual* aggregation ObjectServer pair. Therefore, although the writers

can *fail over* and *fail back* between the primary and backup aggregation layer ObjectServers, the reader stays connected only to its dedicated collection ObjectServer.

2 Aggregation layer

The aggregation layer includes one pair of ObjectServers that are connected by a bidirectional ObjectServer Gateway to keep them synchronized. Note that the bidirectional ObjectServer Gateway runs on the backup host.

All incoming collection gateway writers and all outgoing display gateway readers connect to the virtual aggregation pair (named AGG_V) so that the writers and readers can fail over and fail back if the primary aggregation ObjectServer computer becomes unavailable.

3 Display layer

The display layer includes two standalone display ObjectServers to which both desktop event list users and Web GUI users connect. The configuration includes two display layer ObjectServers, but further display ObjectServers can be added if required. (Details about how to configure additional display layer ObjectServers are included as an extension of the standard multitiered architecture - see the related links at the end of this topic for further information.)

Each display layer ObjectServer has its own dedicated unidirectional ObjectServer Gateway that connects the ObjectServer to the aggregation layer. Each display gateway reader connects to the virtual aggregation pair whereas each gateway writer connects, and is fixed, to its dedicated display ObjectServer. Therefore, although the readers can fail over and fail back between the primary and backup aggregation layer ObjectServers, the writer stays connected only to its dedicated display ObjectServer. (These gateway connections are the opposite of the gateway connections in the collection layer.)

Related concepts

“Naming conventions for the multitiered architecture”

“Adding a second pair of collection ObjectServers” on page 270

“Adding an additional display ObjectServer” on page 275

Related tasks

“Setting up the standard multitiered environment” on page 259

Naming conventions for the multitiered architecture

A naming convention has been devised to help you identify related components in each layer of the multitiered architecture, and the data flow within and across the layers.

Important: Ensure that you use these naming conventions for your ObjectServers and ObjectServer Gateways. The SQL import files and gateway configuration files, which are supplied for setting up the architecture, rely on compliance with these naming conventions. In particular, ensure that the primary ObjectServer names end in _P* and the backup ObjectServer names end in _B*.

The naming conventions used for the standard multitiered architecture are shown in the following table. For each component shown in the first column, the table shows the following details:

- Description: Shows what the component represents.

- Convention: Shows the naming convention for that component, where *n* represents an integer.
- Name provided: Lists examples of the component name, as provided in the standard architecture.
- Additional name: Provides suggested names (where applicable) for additional components, when added. Note that additional names are not required for aggregation ObjectServers and the aggregation gateway because there should only ever be one aggregation ObjectServer pair and one aggregation gateway in any multitiered environment.

Table 69. Naming conventions for the multitiered architecture

Component	Description	Convention	Name provided	Additional name
Collection ObjectServers	Collection <i>primary</i> Objectserver <i>n</i>	COL_P_ <i>n</i>	COL_P_1	COL_P_2 COL_B_2
	Collection <i>backup</i> Objectserver <i>n</i>	COL_B_ <i>n</i>	COL_B_1	COL_P_3 COL_B_3 ...
Collection-to-Aggregation Gateways	Collection <i>primary</i> Objectserver Gateway <i>n</i>	C_TO_A_GATE_P_ <i>n</i>	C_TO_A_GATE_P_1	C_TO_A_GATE_P_2 C_TO_A_GATE_B_2
	Collection <i>backup</i> Objectserver Gateway <i>n</i>	C_TO_A_GATE_B_ <i>n</i>	C_TO_A_GATE_B_1	C_TO_A_GATE_P_3 C_TO_A_GATE_B_3 ...
Aggregation ObjectServers	Aggregation <i>primary</i> ObjectServer	AGG_P	AGG_P	Not applicable
	Aggregation <i>backup</i> ObjectServer	AGG_B	AGG_B	Not applicable
	Aggregation <i>virtual</i> pair	AGG_V	AGG_V	Not applicable
Aggregation Gateway	Aggregation Gateway	AGG_GATE	AGG_GATE	Not applicable
Aggregation-to-Display Gateways	Display Objectserver Gateway <i>n</i>	A_TO_D_GATE_ <i>n</i>	A_TO_D_GATE_1 A_TO_D_GATE_2	A_TO_D_GATE_3 A_TO_D_GATE_4 ...
Display ObjectServers	Display ObjectServer <i>n</i>	DIS_ <i>n</i>	DIS_1 DIS_2	DIS_3 DIS_4 ...

Related concepts

“Overview of the standard multitiered architecture” on page 251

“Multitiered configuration file locations” on page 258

Component resourcing: determining the number of ObjectServers needed

The standard multitiered architecture is based around the aggregation layer. A single-tiered environment in which a primary and a backup ObjectServer are connected by a bidirectional ObjectServer Gateway, is essentially an aggregation pair with no collection or display layers connected.

The modular design of the standard multitiered architecture means that any system can start with a single pair of ObjectServers as an aggregation pair, and then have collection or display components added at any time in the future.

A pragmatic approach to resourcing an architecture design is to start with an aggregation pair. As technical requirements are implemented on the system (for example, probes deployed, triggers built, Netcool/Impact deployed, and users added), additional components can be added based on the profiling information that is produced by the ObjectServers in the architecture.

All ObjectServers in the standard multitiered configuration have profiling enabled to measure the amount of time spent running SQL queries for client connections. Profiling information is automatically recorded in the `$NCHOME/omnibus/log/servername_profiler_report.logn` file, provided the **Profile** property of the ObjectServer is set to TRUE. For example, the profiling log for the primary ObjectServer (COL_P_1) in the collection layer is called `COL_P_1_profiler_report.log1`.

By default, the granularity of the ObjectServer is 60 seconds. This means that every 60 seconds, the ObjectServer records a breakdown of its activities in the last 60 seconds to the profiling log. Sample output is as follows:

```
Thu Nov 20 14:02:50 2008: Individual user profiles:
Thu Nov 20 14:02:50 2008: 'Administrator' (uid = 0) time on IBM-ADAF9B5BAFC: 0.020000s
Thu Nov 20 14:02:50 2008: 'PROBE' (uid = 0) time on devtest12.hursley.ibm.com: 0.000000s
Thu Nov 20 14:02:50 2008: 'isql' (uid = 0) time on devtest12.hursley.ibm.com: 0.000000s
Thu Nov 20 14:02:50 2008: 'GATEWAY' (uid = 0) time on devtest12.hursley.ibm.com: 0.000000s
Thu Nov 20 14:02:50 2008: Grouped user profiles:
Thu Nov 20 14:02:50 2008: Execution time for all connections whose application name is 'Administrator': 0.020000s
Thu Nov 20 14:02:50 2008: Execution time for all connections whose application name is 'PROBE': 0.000000s
Thu Nov 20 14:02:50 2008: Execution time for all connections whose application name is 'isql': 0.000000s
Thu Nov 20 14:02:50 2008: Execution time for all connections whose application name is 'GATEWAY': 0.000000s
Thu Nov 20 14:02:50 2008: Total time in the report period (59.989325s): 0.020000s
```

This preceding sample output shows that very little activity is occurring in this ObjectServer; the only perceivable load is coming from Netcool/OMNIBus Administrator (0.02 seconds).

As an example, if the total time consumed by probes (under the Grouped user profiles section of the log file) increases over time and causes the Total time value to approach the 60-second mark, additional collection ObjectServers could be considered. If the Total time value exceeds the granularity or report period, it is possible that the ObjectServer is falling behind on processing its workload. If the Total time values are only occasionally higher than 60 seconds, the ObjectServer will eventually catch up with its processing. If, however, the Total time values constantly exceed 60 seconds, it is possible that the ObjectServer will get further behind over time.

When running under normal conditions, it is not prudent for the ObjectServer to be consistently using close to the granularity time because all systems should have

contingency for event storm occurrences. A fault management system is of limited value if it is overwhelmed in the event of a network crisis.

Note: It is not just software provisioning that protects against event storm situations. Other aspects such as hardware resourcing, and rules file configuration to discard non-essential alerts, should be considered.

Similar principles apply to the provisioning of display layer ObjectServers. The profiling data can show when the time taken to collectively execute client requests, such as event list updates, causes the total time to approach the granularity period. This information can indicate when it is appropriate to deploy display layer ObjectServers.

Another influencing factor for deploying display layer ObjectServers could be user reports of slow response times. The number of users that an ObjectServer can support largely depends on the number of events in the ObjectServer, the filters being used, and the number of alerts being displayed. As a general rule, when the number of users exceeds five, consider deploying display ObjectServers to ensure good user response times. Each ObjectServer in the display layer should then support up to 30 users. Take note that this number is again dependent on factors such as how the events are being displayed and the number of events.

Severity handling

Before deploying the multitiered configuration, consideration should be given to the way in which the Severity field is handled on deduplication. Note that this applies only to deduplication at the aggregation layer.

In the multitiered configuration, the default setting is to *always* update the Severity field on deduplication. This means that any incoming recurrence of the same event (that is, an event with the same Identifier value) will always be updated with the incoming severity value. Some implications of this are as follows:

- Generic clear by deduplication can be used for significant performance gains.
- Any severity updates from the end point will be applied.
- Any severity changes made to an event by a user *might be lost* on deduplication.

This default setting gives precedence of an event's severity value to the value coming from the end point. In other words, the event should always take on the severity value coming from the end point if the event recurs. This setting has been selected as the default because it is widely used and is necessary to enable generic-clear by deduplication to work.

An alternative approach is the **Reawaken closed on deduplication** setting, which was previously provided in Netcool/OMNIBus V3.x. This method accepts an incoming severity value only if the current severity is clear (that is, set to zero). Otherwise, the severity is not updated. Note that this means generic-clear by deduplication cannot be used. It does, however, mean that user-initiated severity changes to an event are not lost if the event deduplicates. No single approach is more correct than another because it is dependent on customer requirements.

In the deduplication trigger provided in the \$NCHOME/omnibus/extensions/multitier/objectserver/aggregation.sql file, the code fragment that handles the Severity field is contained within the trigger agg_deduplication and is as follows:

```
-----  
-- HANDLE SEVERITY UPDATE ON DEDUPLICATION  
-----
```

```

-- DEFAULT - UPDATE ALWAYS
set old.Severity = new.Severity;
-----
-- REAWAKEN CLOSED ONLY - UPDATE ONLY IF CLEAR
-- if ((old.Severity = 0) and (new.Severity > 0)) then
--
-- set old.Severity = new.Severity;
-- end if;
-----

```

Note that all lines except for the default handling of the Severity field are commented out (that is, preceded by two hyphen characters). To use the **Reawaken closed on deduplication** handling of severity, modify the lines in the preceding code fragment as follows. The changes are highlighted in bold text, and comment out the default code, while uncommenting the **Reawaken closed on deduplication** code.

```

-----
-- HANDLE SEVERITY UPDATE ON DEDUPLICATION
-----
-- DEFAULT - UPDATE ALWAYS
-- set old.Severity = new.Severity;
-----
-- REAWAKEN CLOSED ONLY - UPDATE ONLY IF CLEAR
if ((old.Severity = 0) and (new.Severity > 0)) then

    set old.Severity = new.Severity;
end if;
-----

```

Note: Modify this code *before* applying the aggregation.sql file to the aggregation ObjectServers. Note, however, that the deduplication trigger can be modified at any time after application of the multitiered configuration by using Netcool/OMNIBus Administrator (**nco_config**).

In some cases, your requirements might necessitate the implementation of a custom severity-handling method. As noted in “Creating custom triggers” on page 279, it is important to keep custom code separate from vendor-released code. One main reason for this is so that custom code is not lost when future vendor updates are applied to existing code.

If an alternative custom severity-handling scheme is to be implemented therefore, comment out both default options in the default code as follows, and instead create a separate SQL file with a separate reinsert trigger:

```

-----
-- HANDLE SEVERITY UPDATE ON DEDUPLICATION
-----
-- DEFAULT - UPDATE ALWAYS
-- set old.Severity = new.Severity;
-----
-- REAWAKEN CLOSED ONLY - UPDATE ONLY IF CLEAR
-- if ((old.Severity = 0) and (new.Severity > 0)) then
--
-- set old.Severity = new.Severity;
-- end if;
-----

```

Any custom SQL files should be applied after the SQL files provided in the multitiered configuration by using the **nco_sql** or **isql** utility in the same way that the multitiered configuration files are applied.

The following code shows an example custom reinsert trigger. In this example, the Severity field is updated only if the incoming value is higher than the existing one. (Note that this would not allow generic-clear-by-deduplication to work.)

```
-- CREATE CUSTOM TRIGGER GROUP

CREATE TRIGGER GROUP widgetcom_triggers;
go

-- CREATE CUSTOM REINSERT TRIGGER

CREATE OR REPLACE TRIGGER widgetcom_deduplication
GROUP widgetcom_triggers
PRIORITY 1
COMMENT 'Widgetcom reinsert trigger (alerts.status) to handle the Severity field.'
BEFORE REINSERT ON alerts.status
FOR EACH ROW
begin

    -- UPDATE SEVERITY ONLY IF INCOMING VALUE IS GREATER THAN THE EXISTING VALUE
    if (old.Severity < new.Severity) then

        set old.Severity = new.Severity;
    end if;
end;
go
```

For a complete reference on ObjectServer SQL, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Related concepts

“Multitiered configuration file locations”

Related tasks

“Setting up the standard multitiered environment” on page 259

Related reference

“Creating custom triggers” on page 279

Multitiered configuration file locations

All of the configuration files that are provided for building the multitiered architecture are located in the \$NCHOME/omnibus/extensions/multitier directory.

The gateway subdirectory holds the following customized map definition files (.map), properties files (.props), and table replication definition files (.tblrep.def), which can be used to configure the ObjectServer Gateways in the collection, aggregation, and display layers:

- A_TO_D_GATE.map
- A_TO_D_GATE.tblrep.def
- A_TO_D_GATE_1.props
- A_TO_D_GATE_2.props
- AGG_GATE.map
- AGG_GATE.props
- AGG_GATE.tblrep.def
- C_TO_A_GATE.map
- C_TO_A_GATE_B_1.props
- C_TO_A_GATE_B_1.tblrep.def
- C_TO_A_GATE_P_1.props

- C_TO_A_GATE_P_1.tblrep.def

These gateway files are preconfigured with the required settings, and should be used as provided. The gateway files also require compliance with the naming conventions for ObjectServers and ObjectServer Gateways in the multitiered architecture.

The objectserver subdirectory holds the following customized SQL import files (.sql), which you can apply to the ObjectServers in the collection, aggregation, and display layers, in order to update the database schema with the required multitiered configuration:

- aggregation.sql
- aggregation_rollback.sql
- collection.sql
- collection_rollback.sql
- display.sql
- display_rollback.sql

The SQL files provide automations that require compliance with the naming conventions, and typically:

- Add relevant columns to the database tables.
- Create the automations that control the behavior of the ObjectServers, ObjectServer Gateways, and clients, and the flow of events across these components.
- Enable and disable triggers, as appropriate for the ObjectServers.
- Assign permissions to the triggers.
- Create conversions.
- Roll back the applied schema changes, if required.

All of the files are provided in read-only format. When setting up your multitiered environment, you are required to run commands that reference some of the files, or to make copies of some files for editing.

Use these files as directed in the relevant procedures.

Related concepts

“Naming conventions for the multitiered architecture” on page 253

Setting up the standard multitiered environment

Use this information to set up the environment for the standard multitiered architecture.

If you do not require collection layer ObjectServers, the steps for setting up collection layer ObjectServers and their associated gateways can be skipped. Similarly, if display layer ObjectServers are not required, skip the steps for setting up display layer ObjectServers and their associated gateways. You can add collection and display ObjectServers to the solution at any time, as needed.

The procedure is as follows:

1. Set up the interfaces file.
2. Install the primary aggregation ObjectServer.
3. Install the backup aggregation ObjectServer.

4. Configure the bidirectional aggregation ObjectServer Gateway.
5. Install the primary collection ObjectServer.
6. Configure the unidirectional primary collection ObjectServer Gateway.
7. Install the backup collection ObjectServer.
8. Configure the unidirectional backup Collection ObjectServer Gateway.
9. Install the display ObjectServer 1.
10. Configure the unidirectional display ObjectServer 1 Gateway.
11. Install the display ObjectServer 2.
12. Configure the unidirectional display ObjectServer 2 Gateway.

Related concepts

“Overview of the standard multitiered architecture” on page 251

Configuring server communication information (multitiered architecture)

Each host computer on which the components run must be configured with server communication information that enables the components in the architecture to run and communicate with one another.

On UNIX or Linux, update the communication information for all the server components in your deployment by manually editing the connections data file `$NCHOME/etc/omni.dat`, which is used to create the interfaces file. A suggested good practice is to add all the components in the entire deployment to a single `omni.dat` file, which can then be distributed to all the computers in the deployment. You can then generate the interfaces file from each computer by running the `$NCHOME/bin/nco_igen` command, as described in later procedures. (Interfaces files are named `$NCHOME/etc/interfaces.arch`, where *arch* is the operating system name.) Sample `omni.dat` files are provided to show the configuration for the servers in the aggregation layer only, and in the standard three-tier architecture.

On Windows, configure server communication information on each computer by using the Server Editor, which is available by clicking **Start > All Programs > NETCOOL Suite > System Utilities > Servers Editor**. The information is saved in the connections data file `%NCHOME%\ini\sql.ini`.

Important: You must continue to maintain the server communication information on the host computer and on every computer that needs to connect to the server component, each time that your system configuration changes.

Related concepts

“Naming conventions for the multitiered architecture” on page 253

Related tasks

“Manually editing the connections data file” on page 247

Related reference

“Sample `omni.dat` files” on page 283

Installing the primary aggregation ObjectServer

Use the following steps to install the primary aggregation ObjectServer AGG_P, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the aggregation SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the `$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini` file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer AGG_P and include the SQL import file to be applied to this ObjectServer:

```
$NCHOME/omnibus/bin/nco_dbinit -server AGG_P -customconfigfile  
$NCHOME/omnibus/extensions/multitier/objectserver/aggregation.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied.

5. Start the ObjectServer AGG_P:

```
$NCHOME/omnibus/bin/nco_objserv -name AGG_P &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer

To apply the SQL customization when the ObjectServer is already installed and running, apply the aggregation SQL file against the ObjectServer AGG_P, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server AGG_P -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/  
aggregation.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S AGG_P -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\aggregation.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `aggregation_rollback.sql` script is provided to roll back the changes that the `aggregation.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the `nco_sql` or `isql` utility with the syntax shown for applying the `aggregation.sql` script.

Installing the backup aggregation ObjectServer

Use the following steps to install the backup aggregation ObjectServer AGG_B, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the aggregation SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the `$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini` file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer AGG_B and include the SQL import file to be applied to this ObjectServer:

```
$NCHOME/omnibus/bin/nco_dbinit -server AGG_B -customconfigfile  
$NCHOME/omnibus/extensions/multitier/objectserver/aggregation.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied. If the ObjectServer name ends in `_B` (as per the naming conventions), the **BackupObjectServer** property is automatically set to `TRUE` and the corresponding automations required by the backup ObjectServer are enabled.

5. Start the ObjectServer AGG_B:

```
$NCHOME/omnibus/bin/nco_objserv -name AGG_B &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer

To apply the SQL customization when the ObjectServer is already installed and running, apply the aggregation SQL file against the ObjectServer AGG_B, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server AGG_B -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/  
aggregation.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S AGG_B -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\aggregation.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `aggregation_rollback.sql` script is provided to roll back the changes that the `aggregation.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the `nco_sql` or `isql` utility with the syntax shown for applying the `aggregation.sql` script.

Configuring the bidirectional aggregation ObjectServer Gateway

Use the following steps to configure the bidirectional aggregation ObjectServer Gateway AGG_GATE. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the backup aggregation ObjectServer AGG_B.

To configure the gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/AGG_GATE.*
$NCHOME/omnibus/etc/.
```

The following files are copied to \$NCHOME/omnibus/etc:

- AGG_GATE.map
- AGG_GATE.props
- AGG_GATE.tblrep.def

Windows You can use Windows Explorer to copy these files from %NCHOME%\omnibus\extensions\multitier\gateway and paste them to %NCHOME%\omnibus\etc.

2. Start the gateway AGG_GATE:

```
$NCHOME/omnibus/bin/nco_g_objserv_bi -propsfile $NCHOME/omnibus/etc/
AGG_GATE.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing the primary collection ObjectServer

Use the following steps to install the primary collection ObjectServer COL_P_1, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the collection SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the \$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer COL_P_1 and include the SQL import file to be applied to this ObjectServer:

```
$NCHOME/omnibus/bin/nco_dbinit -server COL_P_1 -customconfigfile
$NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied.

5. Start the ObjectServer COL_P_1:

```
$NCHOME/omnibus/bin/nco_objserv -name COL_P_1 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer

To apply the SQL customization when the ObjectServer is already installed and running, apply the collection SQL file against the ObjectServer COL_P_1, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server COL_P_1 -user root -password password < $NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S COL_P_1 -U root -P password -i "%NCHOME%\omnibus\extensions\multitier\objectserver\collection.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the \$NCHOME/omnibus/extensions/multitier/objectserver directory, the collection_rollback.sql script is provided to roll back the changes that the collection.sql script makes to the ObjectServer, if required. You can apply this rollback script by using the **nco_sql** or **isql** utility with the syntax shown for applying the collection.sql script.

Configuring the unidirectional primary collection ObjectServer Gateway

Use the following steps to configure the unidirectional primary collection ObjectServer Gateway C_TO_A_GATE_P_1. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the primary collection ObjectServer COL_P_1.

To configure the gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE.map $NCHOME/omnibus/etc/.
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE_P_1.* $NCHOME/omnibus/etc/.
```

The following files are copied to \$NCHOME/omnibus/etc:

- C_TO_A_GATE.map
- C_TO_A_GATE_P_1.props
- C_TO_A_GATE_P_1.tblrep.def

Windows You can use Windows Explorer to copy these files from %NCHOME%\omnibus\extensions\multitier\gateway and paste them to %NCHOME%\omnibus\etc.

2. Start the gateway C_TO_A_GATE_P_1:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/C_TO_A_GATE_P_1.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing the backup collection ObjectServer

Use the following steps to install the backup collection ObjectServer COL_B_1, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the collection SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the `$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini` file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer COL_B_1 and include the SQL import file to be applied to this ObjectServer:

```
$NCHOME/omnibus/bin/nco_dbinit -server COL_B_1 -customconfigfile  
$NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied.

5. Start the ObjectServer COL_B_1:

```
$NCHOME/omnibus/bin/nco_objserv -name COL_B_1 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer

To apply the SQL customization when the ObjectServer is already installed and running, apply the collection SQL file against the ObjectServer COL_B_1, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server COL_B_1 -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S COL_B_1 -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\collection.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `collection_rollback.sql` script is provided to roll back the changes that the `collection.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the `nco_sql` or `isql` utility with the syntax shown for applying the `collection.sql` script.

Configuring the unidirectional backup collection ObjectServer Gateway

Use the following steps to configure the unidirectional backup collection ObjectServer Gateway C_TO_A_GATE_B_1. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the backup collection ObjectServer COL_B_1.

To configure the gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE.map  
$NCHOME/omnibus/etc/.  
  
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE_B_1.*  
$NCHOME/omnibus/etc/.
```

The following files are copied to \$NCHOME/omnibus/etc:

- C_TO_A_GATE.map
- C_TO_A_GATE_B_1.props
- C_TO_A_GATE_B_1.tblrep.def

Windows You can use Windows Explorer to copy these files from %NCHOME%\omnibus\extensions\multitier\gateway and paste them to %NCHOME%\omnibus\etc.

2. Start the gateway C_TO_A_GATE_B_1:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/  
C_TO_A_GATE_B_1.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing the display ObjectServer 1

Use the following steps to install the first display ObjectServer DIS_1, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the display SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the \$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer DIS_1 and include the SQL import file to be applied to this ObjectServer. The additional command-line options -desktopserver, -dsddualwrite, and -dsdprimary are required for the initialization of display layer ObjectServers. Notice that the -dsdprimary command-line option is set to the name of the virtual ObjectServer pair in the aggregation layer.

```
$NCHOME/omnibus/bin/nco_dbinit -server DIS_1 -desktopserver  
-dsddualwrite -dsdprimary AGG_V -customconfigfile $NCHOME/omnibus/  
extensions/multitier/objectserver/display.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The ObjectServer is created as a desktop ObjectServer with dual-write mode enabled. The SQL customization is also applied.

5. Start the ObjectServer DIS_1:

```
$NCHOME/omnibus/bin/nco_objserv -name DIS_1 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Related concepts

Chapter 12, “Setting up desktop ObjectServers,” on page 327

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Applying the SQL customization to a running ObjectServer

When creating the ObjectServer, you must have run the **nco_dbinit** command with the **-desktopserver**, **-dsdualwrite**, and **-dsdprimary** command-line options.

To apply the SQL customization when the ObjectServer is already installed and running, apply the display SQL file against the ObjectServer DIS_1, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server DIS_1 -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/display.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S DIS_1 -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\display.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `display_rollback.sql` script is provided to roll back the changes that the `display.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the **nco_sql** or **isql** utility with the syntax shown for applying the `display.sql` script.

Configuring the unidirectional display ObjectServer 1 Gateway

Use the following steps to configure the unidirectional ObjectServer Gateway `A_TO_D_GATE_1` for the display ObjectServer DIS_1. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the first display ObjectServer DIS_1.

To configure the unidirectional display ObjectServer Gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.map  
$NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.tblrep.def  
$NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE_1.props  
$NCHOME/omnibus/etc/.
```

The following files are copied to `$NCHOME/omnibus/etc`:

- `A_TO_D_GATE.map`

- A_TO_D_GATE.tblrep.def
- A_TO_D_GATE_1.props

Windows You can use Windows Explorer to copy these files from %NCHOME%\omnibus\extensions\multitier\gateway and paste them to %NCHOME%\omnibus\etc.

2. Start the gateway A_TO_D_GATE_1:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/A_TO_D_GATE_1.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing the display ObjectServer 2

Use the following steps to install the second display ObjectServer DIS_2, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the display SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the \$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer DIS_2 and include the SQL import file to be applied to this ObjectServer. The additional command-line options -desktopserver, -dsddualwrite, and -dsdprimary are required for the initialization of display layer ObjectServers. Notice that the -dsdprimary command-line option is set to the name of the virtual ObjectServer pair in the aggregation layer.

```
$NCHOME/omnibus/bin/nco_dbinit -server DIS_2 -desktopserver -dsddualwrite -dsdprimary AGG_V -customconfigfile $NCHOME/omnibus/extensions/multitier/objectserver/display.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The ObjectServer is created as a desktop ObjectServer with dual-write mode enabled. The SQL customization is also applied.

5. Start the ObjectServer DIS_2:

```
$NCHOME/omnibus/bin/nco_objserv -name DIS_2 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Related concepts

Chapter 12, “Setting up desktop ObjectServers,” on page 327

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Applying the SQL customization to a running ObjectServer

When creating the ObjectServer, you must have run the **nco_dbinit** command with the **-desktopserver**, **-dsdualwrite**, and **-dsdprimary** command-line options.

To apply the SQL customization when the ObjectServer is already installed and running, apply the display SQL file against the ObjectServer DIS_2, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server DIS_2 -user root -password password < $NCHOME/omnibus/extensions/multitier/objectserver/display.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S DIS_2 -U root -P password -i "%NCHOME%\omnibus\extensions\multitier\objectserver\display.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `display_rollback.sql` script is provided to roll back the changes that the `display.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the **nco_sql** or **isql** utility with the syntax shown for applying the `display.sql` script.

Configuring the unidirectional display ObjectServer 2 Gateway

Use the following steps to configure the unidirectional ObjectServer Gateway `A_TO_D_GATE_2` for the display ObjectServer `DIS_2`. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the second display ObjectServer `DIS_2`.

To configure the gateway:

- UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.map $NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.tblrep.def $NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE_2.props $NCHOME/omnibus/etc/.
```

The following files are copied to `$NCHOME/omnibus/etc`:

- `A_TO_D_GATE.map`
- `A_TO_D_GATE.tblrep.def`
- `A_TO_D_GATE_2.props`

Windows You can use Windows Explorer to copy these files from `%NCHOME%\omnibus\extensions\multitier\gateway` and paste them to `%NCHOME%\omnibus\etc`.

- Start the gateway `A_TO_D_GATE_2`:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/A_TO_D_GATE_2.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing additional ObjectServers

You can add collection and display ObjectServers to the standard multitiered architecture. Analyze the requirements for your environment and then consider whether to add collection or display ObjectServers.

Adding a second pair of collection ObjectServers

If the loading of the collection ObjectServers starts to approach its capacity, you can deploy additional pairs of collection ObjectServers to share the load. You can monitor the profiling data that is recorded to determine whether the total time used by each ObjectServer is approaching the granularity period.

The following figure shows the standard multitiered architecture with an additional pair of collection ObjectServers and associated ObjectServer Gateways. Take note that the ObjectServers and gateways follow the naming convention established earlier.

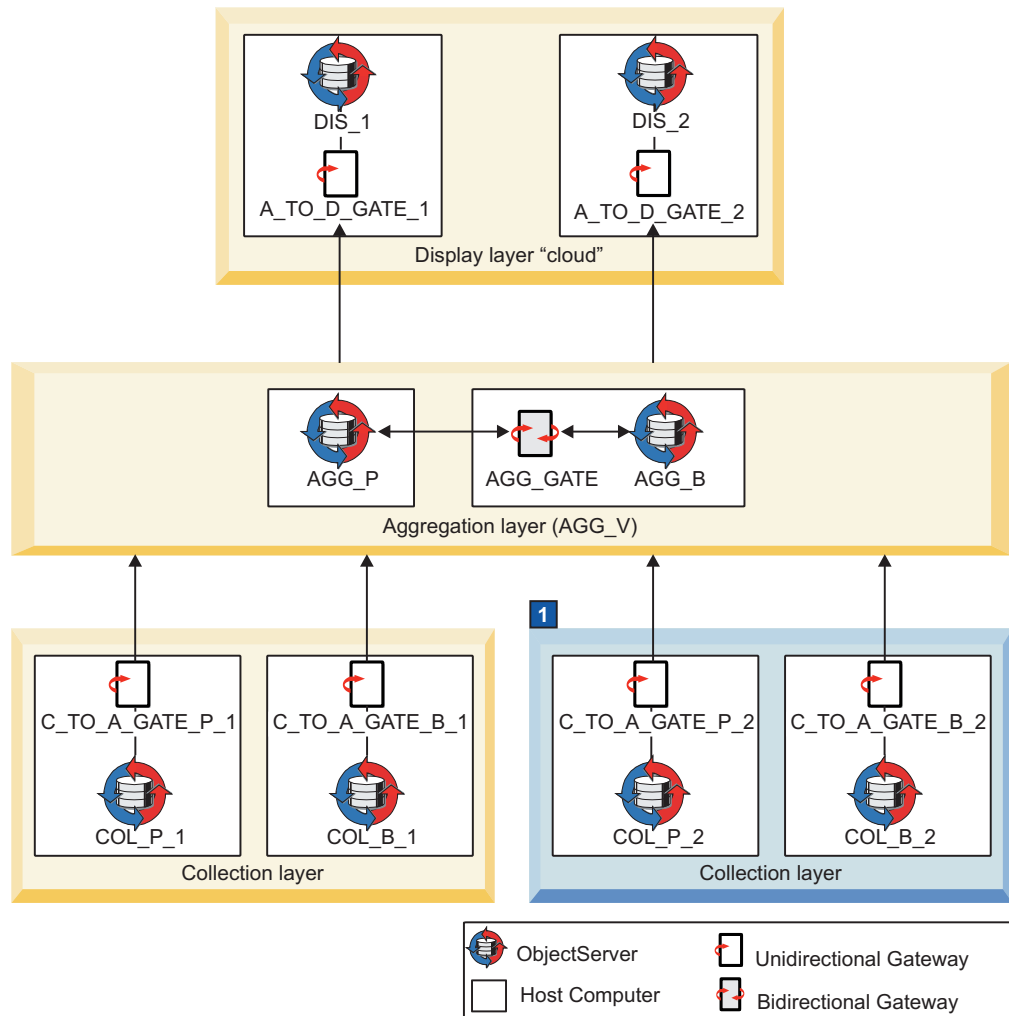


Figure 5. Second pair of collection ObjectServers and gateways added to the standard multitiered architecture

1 Second pair of ObjectServers and unidirectional ObjectServer Gateways in the collection layer

The ObjectServers are shown as a primary and backup pair in the

collection layer at the bottom right of the figure. This deployment of a second pair of collection ObjectServers follows a similar process to the initial pair in the standard architecture. Each ObjectServer has its own dedicated unidirectional ObjectServer Gateway that connects the ObjectServer to the aggregation layer.

You can deploy as many ObjectServer pairs in the collection layer as necessary to meet your requirements.

Related concepts

“Overview of the standard multitiered architecture” on page 251

“Naming conventions for the multitiered architecture” on page 253

Installing an additional primary collection ObjectServer

Use the following steps to install the additional primary collection ObjectServer COL_P_2, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the collection SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the `$NCHOME/etc/omni.dat` file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer COL_P_2 and include the SQL import file to be applied to this ObjectServer:

```
$NCHOME/omnibus/bin/nco_dbinit -server COL_P_2 -customconfigfile  
$NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied.

5. Start the ObjectServer COL_P_2:

```
$NCHOME/omnibus/bin/nco_objserv -name COL_P_2 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer:

To apply the SQL customization when the ObjectServer is already installed and running, apply the collection SQL file against the ObjectServer COL_P_2, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server COL_P_2 -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S COL_P_2 -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\collection.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `collection_rollback.sql` script is provided to roll back the changes that the `collection.sql` script makes to the ObjectServer, if required. You can apply this

rollback script by using the **nco_sql** or **isql** utility with the syntax shown for applying the `collection.sql` script.

Configuring an additional unidirectional primary collection ObjectServer Gateway

Use the following steps to configure the additional unidirectional primary collection ObjectServer Gateway `C_TO_A_GATE_P_2`. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the additional primary collection ObjectServer `COL_P_2`.

To configure the additional gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE.map
$NCHOME/omnibus/etc/.

cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE_P_1.*
$NCHOME/omnibus/etc/.
```

The following files are copied to `$NCHOME/omnibus/etc`:

- `C_TO_A_GATE.map`
- `C_TO_A_GATE_P_1.props`
- `C_TO_A_GATE_P_1.tblrep.def`

Windows You can use Windows Explorer to copy these files from `%NCHOME%\omnibus\extensions\multitier\gateway` and paste them to `%NCHOME%\omnibus\etc`.

2. Remove the read-only permissions from the three files, and then rename the following files:

```
cd $NCHOME/omnibus/etc
mv C_TO_A_GATE_P_1.props C_TO_A_GATE_P_2.props
mv C_TO_A_GATE_P_1.tblrep.def C_TO_A_GATE_P_2.tblrep.def
```

The files in the `$NCHOME/omnibus/etc` directory should now be called: `C_TO_A_GATE.map`, `C_TO_A_GATE_P_2.props`, and `C_TO_A_GATE_P_2.tblrep.def`.

3. Edit the `C_TO_A_GATE_P_2.props` file and change only the following lines:

MessageLog	: '\$OMNIHOME/log/C_TO_A_GATE_P_2.log'
Name	: 'C_TO_A_GATE_P_2'
Gate.Reader.Server	: 'COL_P_2'
Gate.Reader.TblReplicateDefFile	: '\$OMNIHOME/etc/C_TO_A_GATE_P_2.tblrep.def'
Gate.Writer.SAFFile	: '\$OMNIHOME/var/objserv_uni/C_TO_A_GATE_P_2.store'

4. Edit the `C_TO_A_GATE_P_2.tblrep.def` file and change only the following line:

```
CACHE FILTER 'SourceServerName = \'COL_P_2\'';
```

5. Start the gateway `C_TO_A_GATE_P_2`:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/
C_TO_A_GATE_P_2.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Installing an additional backup collection ObjectServer

Use the following steps to install the additional backup collection ObjectServer COL_B_2, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the collection SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIbus and ensure that all components are selected for installation.
2. Ensure that the `$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini` file is configured with all the component details.
3. **UNIX** Generate the interfaces file as follows:
`$NCHOME/bin/nco_igen`
4. Initialize the ObjectServer COL_B_2 and include the SQL import file to be applied to this ObjectServer:
`$NCHOME/omnibus/bin/nco_dbinit -server COL_B_2 -customconfigfile $NCHOME/omnibus/extensions/multitier/objectserver/collection.sql`
The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The SQL customization is also applied.
5. Start the ObjectServer COL_B_2:
`$NCHOME/omnibus/bin/nco_objserv -name COL_B_2 &`
The ObjectServer is confirmed as initialized and entering a RUN state.

Applying the SQL customization to a running ObjectServer:

To apply the SQL customization when the ObjectServer is already installed and running, apply the collection SQL file against the ObjectServer COL_B_2, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server COL_B_2 -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/collection.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S COL_B_2 -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\collection.sql"
```

It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `collection_rollback.sql` script is provided to roll back the changes that the `collection.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the `nco_sql` or `isql` utility with the syntax shown for applying the `collection.sql` script.

Configuring an additional unidirectional backup collection ObjectServer Gateway

Use the following steps to configure the additional unidirectional backup collection ObjectServer Gateway C_TO_A_GATE_B_2. Note that installation of Tivoli Netcool/OMNIbus is not necessary because the gateway is configured on the same host computer as the additional backup collection ObjectServer COL_B_2.

To configure the additional gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE.map  
$NCHOME/omnibus/etc/.
```

```
cp $NCHOME/omnibus/extensions/multitier/gateway/C_TO_A_GATE_B_1.*  
$NCHOME/omnibus/etc/.
```

The following files are copied to \$NCHOME/omnibus/etc:

- C_TO_A_GATE.map
- C_TO_A_GATE_B_1.props
- C_TO_A_GATE_B_1.tblrep.def

Windows You can use Windows Explorer to copy these files from %NCHOME%\omnibus\extensions\multitier\gateway and paste them to %NCHOME%\omnibus\etc.

2. Remove the read-only permissions from the three files, and then rename the following files:

```
cd $NCHOME/omnibus/etc  
mv C_TO_A_GATE_B_1.props C_TO_A_GATE_B_2.props  
mv C_TO_A_GATE_B_1.tblrep.def C_TO_A_GATE_B_2.tblrep.def
```

The files in the \$NCHOME/omnibus/etc directory should now be called: C_TO_A_GATE.map, C_TO_A_GATE_B_2.props, and C_TO_A_GATE_B_2.tblrep.def.

3. Edit the C_TO_A_GATE_B_2.props file and change only the following lines:

```
MessageLog           : '$OMNIHOME/log/C_TO_A_GATE_B_2.log'  
Name                 : 'C_TO_A_GATE_B_2'  
Gate.Reader.Server   : 'COL_B_2'  
Gate.Reader.TblReplicateDefFile : '$OMNIHOME/etc/C_TO_A_GATE_B_2.tblrep.def'  
Gate.Writer.SAFile    : '$OMNIHOME/var/objserv_uni/C_TO_A_GATE_B_2.store'
```

4. Edit the C_TO_A_GATE_B_2.tblrep.def file and change only the following line:

```
CACHE FILTER 'ServerName = \'COL_B_2\'';
```

5. Start the gateway C_TO_A_GATE_B_2:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/  
C_TO_A_GATE_B_2.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Adding an additional display ObjectServer

If the loading of the display ObjectServers starts to approach its capacity, you can deploy additional display ObjectServers to share the load. You can monitor the profiling data that is recorded to determine whether the total time used by each ObjectServer is approaching the granularity period. You can also choose to deploy additional ObjectServers if users are reporting slow response times.

The following figure shows a multitiered architecture with an additional display ObjectServer and associated ObjectServer Gateway. Take note that the ObjectServer and gateway follow the naming convention established earlier.

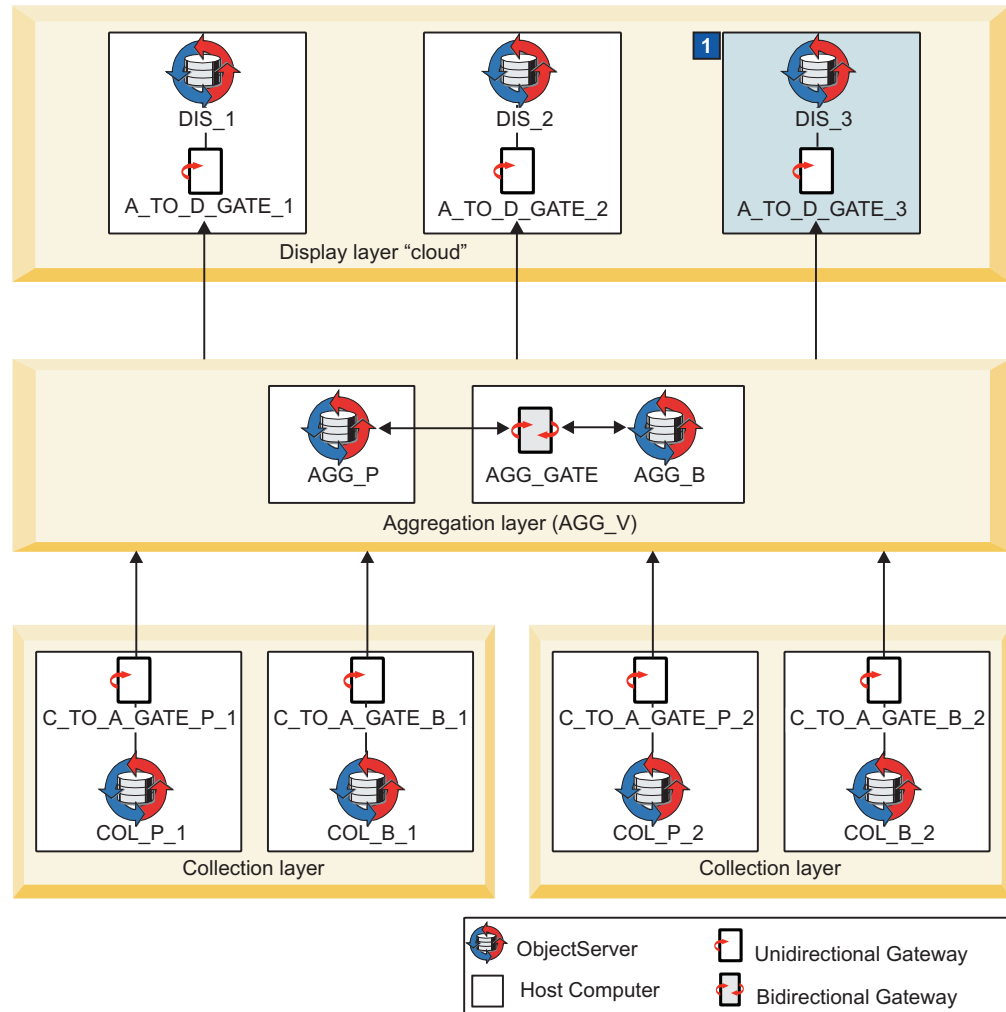


Figure 6. Additional display ObjectServer and gateway added to the multitiered architecture

1 Additional ObjectServer and unidirectional ObjectServer Gateway in the display layer

The ObjectServer is shown in the display layer at the top right of the figure. This deployment of an additional display ObjectServer follows a similar process to the initial ones in the architecture. The display ObjectServer has its own dedicated unidirectional ObjectServer Gateway that connects the ObjectServer to the aggregation layer.

You can deploy as many ObjectServers in the display layer as necessary to meet your requirements.

Related concepts

“Overview of the standard multitiered architecture” on page 251

“Naming conventions for the multitiered architecture” on page 253

Installing an additional display ObjectServer

Use the following steps to install an additional display ObjectServer DIS_3, and apply the SQL customization. If the ObjectServer is already installed and running, you can apply the SQL customization to the ObjectServer by using the display SQL file provided.

To install and configure the ObjectServer:

1. Install Tivoli Netcool/OMNIBus and ensure that all components are selected for installation.
2. Ensure that the \$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini file is configured with all the component details.

3. **UNIX** Generate the interfaces file as follows:

```
$NCHOME/bin/nco_igen
```

4. Initialize the ObjectServer DIS_3 and include the SQL import file to be applied to this ObjectServer. The additional command-line options -desktopserver, -dsddualwrite, and -dsdprimary are required for the initialization of display layer ObjectServers. Notice that the -dsdprimary command-line option is set to the name of the virtual ObjectServer pair in the aggregation layer.

```
$NCHOME/omnibus/bin/nco_dbinit -server DIS_3 -desktopserver  
-dsddualwrite -dsdprimary AGG_V -customconfigfile $NCHOME/omnibus/  
extensions/multitier/objectserver/display.sql
```

The properties file, and default database tables, data, users, groups, and roles are created for the ObjectServer. The ObjectServer is created as a desktop ObjectServer with dual-write mode enabled. The SQL customization is also applied.

5. Start the ObjectServer DIS_3:

```
$NCHOME/omnibus/bin/nco_objserv -name DIS_3 &
```

The ObjectServer is confirmed as initialized and entering a RUN state.

Related concepts

Chapter 12, “Setting up desktop ObjectServers,” on page 327

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Applying the SQL customization to a running ObjectServer:

When creating the ObjectServer, you must have run the **nco_dbinit** command with the -desktopserver, -dsddualwrite, and -dsdprimary command-line options.

To apply the SQL customization when the ObjectServer is already installed and running, apply the display SQL file against the ObjectServer DIS_3, as follows:

```
UNIX $NCHOME/omnibus/bin/nco_sql -server DIS_3 -user root -password  
password < $NCHOME/omnibus/extensions/multitier/objectserver/display.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S DIS_3 -U root -P password -i  
"%NCHOME%\omnibus\extensions\multitier\objectserver\display.sql"
```


It is assumed you are logged on as root with a preferred password.

Tip: In the `$NCHOME/omnibus/extensions/multitier/objectserver` directory, the `display_rollback.sql` script is provided to roll back the changes that the `display.sql` script makes to the ObjectServer, if required. You can apply this rollback script by using the `nco_sql` or `isql` utility with the syntax shown for applying the `display.sql` script.

Configuring an additional unidirectional display ObjectServer Gateway

Use the following steps to configure an additional unidirectional ObjectServer Gateway `A_TO_D_GATE_3` for the display ObjectServer `DIS_3`. Note that installation of Tivoli Netcool/OMNIBus is not necessary because the gateway is configured on the same host computer as the additional display ObjectServer `DIS_3`.

To configure the unidirectional display ObjectServer Gateway:

1. **UNIX** Copy the multitiered property files for the gateway, to the default location where configuration and properties files are held:

```
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.map  
$NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE.tblrep.def  
$NCHOME/omnibus/etc/.  
cp $NCHOME/omnibus/extensions/multitier/gateway/A_TO_D_GATE_1.props  
$NCHOME/omnibus/etc/.
```

The following files are copied to `$NCHOME/omnibus/etc`:

- `A_TO_D_GATE.map`
- `A_TO_D_GATE.tblrep.def`
- `A_TO_D_GATE_1.props`

Windows You can use Windows Explorer to copy these files from `%NCHOME%\omnibus\extensions\multitier\gateway` and paste them to `%NCHOME%\omnibus\etc`.

2. Remove the read-only permissions from the three files, and then rename the following file:

```
cd $NCHOME/omnibus/etc  
mv A_TO_D_GATE_1.props A_TO_D_GATE_3.props
```

The files in the `$NCHOME/omnibus/etc` directory should now be called: `A_TO_D_GATE.map`, `A_TO_D_GATE.tblrep.def`, and `A_TO_D_GATE_3.props`.

3. Edit the `A_TO_D_GATE_3.props` file and change only the following lines:

MessageLog	:	'\$OMNIHOME/log/A_TO_D_GATE_3.log'
Name	:	'A_TO_D_GATE_3'
Gate.Writer.Server	:	'DIS_3'
Gate.Writer.SAFFile	:	'\$OMNIHOME/var/objserv_uni/A_TO_D_GATE_3.store'

4. Start the gateway `A_TO_D_GATE_3`:

```
$NCHOME/omnibus/bin/nco_g_objserv_uni -propsfile $NCHOME/omnibus/etc/  
A_TO_D_GATE_3.props &
```

The gateway is confirmed as initialized and entering a RUN state.

Automatic load balancing of event list clients

The display ObjectServers can be configured to automatically load balance event lists that connect to them. Once configured, users are automatically load balanced over the available display ObjectServers, regardless of the display ObjectServer that was selected when the users logged in.

Load balancing is implemented by populating the master.servergroups table in the display ObjectServers. The master.servergroups table of each display ObjectServer contains the same information: one row for each display ObjectServer in the architecture design. When users connect, the event list queries the contents of the table, selects an available display ObjectServer (based on an internal algorithm), and then connects to that display ObjectServer. The load balancing that results is not fully distributed evenly, but is approximately even.

Note: The contents of the master.servergroups table are ignored unless the master.national table contains an entry. The master.national table is automatically populated for the display layer ObjectServers by the configuration file \$NCHOME/omnibus/extensions/multitier/display.sql.

The aggregation SQL file \$NCHOME/omnibus/extensions/multitier/objectserver/aggregation.sql contains the following lines of code, which are designed to populate the master.servergroups table with details for the two display ObjectServers in the standard multitiered architecture:

```
-----
-- INITIALISE LOAD-BALANCING FOR THE DISPLAY OBJECTSERVERS
-- NOT ENABLED BY DEFAULT
DELETE FROM master.servergroups;
go
-- INSERT INTO master.servergroups VALUES('DIS_1',1,1);
-- INSERT INTO master.servergroups VALUES('DIS_2',1,1);
-- go
```

The data is inserted at the aggregation layer because the master.servergroups table is configured to automatically replicate out from the aggregation ObjectServers to the display ObjectServers. The advantage of this is that the information needs to be updated in only one place if additional display ObjectServers are added at a later date. Also, it helps to ensure that all display ObjectServers contain the same data, thereby reducing the possibility of errors.

To enable automatic load balancing for event lists:

1. Copy the \$NCHOME/omnibus/extensions/multitier/objectserver/aggregation.sql file to another location, and then remove the read-only permissions.
2. Edit the aggregation.sql file by uncommenting the two INSERT statements and the go keyword, as follows:

```
-----
-- INITIALISE LOAD-BALANCING FOR THE DISPLAY OBJECTSERVERS
-- NOT ENABLED BY DEFAULT
DELETE FROM master.servergroups;
go
INSERT INTO master.servergroups VALUES('DIS_1',1,1);
INSERT INTO master.servergroups VALUES('DIS_2',1,1);
go
```

In each INSERT statement, the first value (DIS_1 or DIS_2) populates the ServerName column in the table. The second value (1) populates the GroupID column, and the third value (1) populates the Weight column. If you want

twice as many users connecting to ObjectServer DIS_1 than DIS_2, for example, set the weighting on DIS_1 to 2 and set the weighting on DIS_2 to 1.

3. Save and close the file.

You must now apply the file to the aggregation ObjectServers by using the **nco_sql** (UNIX) or **isql** (Windows) commands.

Including additional display ObjectServers in the load balancing configuration

If additional display ObjectServers are added to the configuration, you must insert additional lines into the edited `aggregation.sql` file to include the additional display ObjectServers in the load-balancing. For example, if ObjectServer DIS_3 is added to the display layer, add a third INSERT statement as follows:

```
-----  
-- INITIALISE LOAD-BALANCING FOR THE DISPLAY OBJECTSERVERS  
-- NOT ENABLED BY DEFAULT  
DELETE FROM master.servergroups;  
go  
INSERT INTO master.servergroups VALUES('DIS_1',1,1);  
INSERT INTO master.servergroups VALUES('DIS_2',1,1);  
INSERT INTO master.servergroups VALUES('DIS_3',1,1);  
go
```

You can apply the `aggregation.sql` file to the aggregation ObjectServers multiple times:

- You can add additional values to the file and then apply the file to the primary aggregation ObjectServer.
- You can modify the `master.servergroups` table on the primary aggregation ObjectServer by using Netcool/OMNIBus Administrator (**nco_config**) or the SQL interactive interface (**nco_sql** or **isql**).

Any additions or changes are automatically propagated to the backup aggregation ObjectServer and all the display ObjectServers through the gateways.

Note: Subsequent applications of the `aggregation.sql` file can generate errors. These errors occur because the SQL is either attempting to create fields that already exist, or is attempting to insert rows that already exist. It is safe to ignore these errors.

Related concepts

“Load balanced mode” on page 333

“Overview of the standard multitiered architecture” on page 251

Creating custom triggers

The multitiered architecture configuration works by carefully controlling insert, reinsert and update operations in the ObjectServers. The triggers have been intentionally set with a priority of 2 so that any custom insert, reinsert or update operations that are required can be implemented in separate triggers with a priority of 1. This priority setting ensures that the custom triggers are executed first.

Guidelines for creating custom triggers include:

- Always create a new trigger group for custom triggers; for example, `customer_x_triggers`.
- Do not modify the default triggers; create new ones and keep them separate.

Tip: The main reason for creating separate trigger groups and triggers for custom functionality is to eliminate the risk of overwriting custom functionality. If the default triggers have not been modified, they can be safely replaced with updated versions, as required in future product releases.

Example 1

You want to add a new custom field that you want to update on deduplication. Typical steps that can be performed are as follows:

1. Add custom fields to all ObjectServers.
2. Add custom fields to all gateway mapping files.
3. Create a new trigger group on the collection and aggregation ObjectServers; for example, x_triggers.
4. Create a new reinsert trigger on the collection and aggregation ObjectServers. Set the priority of the trigger to 1 and assign it to the newly-created trigger group. Update the trigger action to include the lines of code that are needed to update the field. For example:

```
set old.MyField = new.MyField;
```

Example 2

You now want to add a new trigger that performs some custom correlation. Typical steps that can be performed are as follows:

1. Create a new temporal trigger on both the primary and backup aggregation ObjectServers.
2. Set the priority of the trigger to 1, choose a suitable prime number timing (for example, 61 seconds) and assign the trigger to the primary_only trigger group to ensure the trigger gets enabled or disabled correctly on the backup ObjectServer.

Triggers that perform correlation should run only on the primary ObjectServer or the backup ObjectServer. Therefore, the new temporal trigger is assigned to the primary_only trigger group because this trigger group is automatically enabled or disabled if the primary aggregation ObjectServer fails or starts up.

Note: When designing any trigger, consider whether it should run concurrently on both the primary and backup aggregation ObjectServers, or whether it should run only on the acting primary aggregation ObjectServer.

The performance triggers

When setting up a multitiered environment, the supplied SQL scripts add triggers and fields that enable event list users to see how long it takes for alerts to reach the display layer ObjectServer to which they are connected. This performance data provides useful feedback about the overall health of the system.

The following figure depicts an event list that includes one of the new fields (TimeToDisplay) that is added to display layer ObjectServers. The TimeToDisplay field shows the number of seconds calculated from the time when an event was initially inserted into any ObjectServer, to the time of insertion into the current display ObjectServer.

In addition, a synthetic event is generated on each display ObjectServer to inform users of the average time to display *all* events that are currently in the display ObjectServer to which the users are connected. (In the figure, the synthetic event is shown as the first row in the event list.)

Note: The average TimeToDisplay value is typically between 20 and 40 seconds in a three-tiered environment.

Node	Summary	Type	TimeToDisplay
DIS_1	Average time to display events since Gateway connect: 33 seconds	Information	0
Test2	Test event 2	Problem	27
Test3	Test event 3	Problem	27
Test4	Test event 4	Problem	27
Test5	Test event 5	Problem	27
myhost	An isql process running on myhost has connected	Problem	27
myhost	An isql process running on myhost has disconnected	Resolution	27
Test1	Test event 1	Problem	30
Test2	Test event 2	Problem	30
Test3	Test event 3	Problem	30
Test4	Test event 4	Problem	30
Test5	Test event 5	Problem	30

Figure 7. Event list showing TimeToDisplay field and synthetic event

The performance data can occasionally get skewed upwards, when, for example, a display gateway is restarted. When the gateway is restarted, a full resynchronization is initiated, and the corresponding display ObjectServer is refreshed with the event data from the aggregation layer. Because the display timestamp is set to the time when the event was inserted into the display ObjectServer, these timestamps are all refreshed to show the current time.

When the TimeToDisplay metric is subsequently calculated, the value will be incorrect (that is, skewed upwards) because the calculation compares the current time with the time when the event was inserted in the aggregation layer. (The insertion into the aggregation layer might have occurred some time ago.) The following figure depicts skewed values in the first two rows of an event list.

Node	Summary	Type	TimeToDisplay
myhost	A GATEWAY process display_gate running on myhost ...	Resolution	15
myhost	A GATEWAY process display_gate running on myhost ...	Problem	22
Test3	Test event 3	Problem	1307
Test4	Test event 4	Problem	1312
Test5	Test event 5	Problem	1307
Test6	Test event 6	Problem	1308
Test7	Test event 7	Problem	1312
Test1	Test event 1	Problem	1308
Test2	Test event 2	Problem	1312
Test3	Test event 3	Problem	1310
Test4	Test event 4	Problem	1311
Test5	Test event 5	Problem	1308

Figure 8. Event list showing skewed TimeToDisplay field values after failback

To counteract this effect, the `calculate_time_to_display` trigger, which calculates the average `TimeToDisplay` value, only includes in its calculations events whose `LastOccurrence` time occurs after the time when the aggregation-to-display gateway connected. This is important because the event might be fairly old (that is, the `CollectionFirst` or `AggregationFirst` timestamps might have occurred some time ago). However, the `DisplayFirst` value is always shown as the time when the event was initially inserted into the display `ObjectServer`; this value will be new each time the gateway restarts or reconnects. (The `DisplayFirst` field is set on initial insert into the display `ObjectServer` by an insert database trigger and therefore gets reset each time the gateway either restarts or resynchronizes when failover or fallback occurs.)

Events that occurred before the connection time of the gateway are therefore excluded from the average calculation and the `TimeToDisplay` field for those events is set to `N/A` (that is, not applicable) to enable those events to be easily identified in the event list, as shown in the following figure.

Node	Summary	Type	TimeToDisplay
DIS_1	Average time to display events: 52 seconds	Information	0
Test2	Test event 2	Problem	N/A
Test3	Test event 3	Problem	N/A
Test4	Test event 4	Problem	N/A
Test5	Test event 5	Problem	N/A
Test6	Test event 6	Problem	N/A
Test7	Test event 7	Resolution	N/A
Test1	Test event 1	Problem	N/A
Test2	Test event 2	Problem	N/A
Test3	Test event 3	Problem	N/A
Test4	Test event 4	Problem	N/A
Test5	Test event 5	Problem	N/A

Figure 9. Event list showing corrected `TimeToDisplay` values

Resynchronization Complete synthetic events

Triggers are available in the aggregation layer `ObjectServers` to create synthetic events that indicate when gateways complete resynchronization.

These events are displayed in the event list, as shown in the following figure.

The Resynchronization Complete events have an expiry time of 86,400 seconds (or 24 hours). Such events are informational only and hence should not remain in the `ObjectServer` indefinitely. When the events become 24 hours old, the `expire` trigger clears them by setting the `Severity` to 0. The events are subsequently deleted by the `delete_clears` trigger.

Node	Summary	Type	TimeToDisplay	ExpireTime
AGG_P	Failover Gateway resynchronisation complete on myhost	Information	40	86400
AGG_P	Display Gateway resynchronisation complete on myhost	Information	22	86400
AGG_P	Display Gateway resynchronisation complete on myhost	Information	20	86400
AGG_P	Collection Gateway resynchronisation complete on myhost	Information	33	86400
AGG_P	Collection Gateway resynchronisation complete on myhost	Information	47	86400

Figure 10. Event list showing Resynchronisation Complete synthetic events

Resynchronization Complete events are a useful way of indicating that gateways have reconnected and successfully completed the resynchronization process after a failover and failback operation, or a disconnection and reconnection.

Final steps

After all the components are installed and configured in the multitiered environment, set up the components to run under process control.

For further information about process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Sample omni.dat files

Two sample connections data files \$NCHOME/etc/omni.dat are provided here with communication details of all the components in a basic failover configuration (aggregation layer only), and in the standard multitiered architecture. In these files, the components are all shown as installed on the same host.

omni.dat file for a failover configuration

```
#
# omni.dat file as prototype for interfaces file
#
# Ident: $Id: omni.dat 1.5 1999/07/13 09:34:20 chris Development $
#

[AGG_P]
{
    Primary: myhost_name.ibm.com 4100
}

[AGG_B]
{
    Primary: myhost_name.ibm.com 4150
}

[AGG_V]
{
    Primary: myhost_name.ibm.com 4100
    Backup: myhost_name.ibm.com 4150
}

[AGG_GATE]
{
    Primary: myhost_name.ibm.com 4105
}
```

omni.dat file for the standard multitiered architecture

```
#
# omni.dat file as prototype for interfaces file
#
# Ident: $Id: omni.dat 1.5 1999/07/13 09:34:20 chris Development $
#

[AGG_P]
{
    Primary: myhost_name.ibm.com 4100
}

[AGG_B]
{
    Primary: myhost_name.ibm.com 4150
}

[AGG_V]
{
    Primary: myhost_name.ibm.com 4100
    Backup: myhost_name.ibm.com 4150
}

[COL_P_1]
{
    Primary: myhost_name.ibm.com 4101
}

[COL_B_1]
{
    Primary: myhost_name.ibm.com 4151
}

[DIS_1]
{
    Primary: myhost_name.ibm.com 4102
}

[DIS_2]
{
    Primary: myhost_name.ibm.com 4152
}

[C_TO_A_GATE_P_1]
{
    Primary: myhost_name.ibm.com 4103
}

[C_TO_A_GATE_B_1]
{
    Primary: myhost_name.ibm.com 4153
}

[A_TO_D_GATE_1]
{
    Primary: myhost_name.ibm.com 4104
}

[A_TO_D_GATE_2]
{
    Primary: myhost_name.ibm.com 4154
}

[AGG_GATE]
{
    Primary: myhost_name.ibm.com 4105
}
```

Chapter 9. Configuring high availability

When Tivoli Netcool/OMNIBus is configured for high availability, event loss is minimized, data integrity is improved, and performance is increased.

The multitiered architecture provides the backdrop for a high availability setup in which ObjectServers are deployed in a one-, two-, or three-tiered configuration. The multitiered architecture enables you to start your deployment in the aggregation layer or tier, and to add ObjectServer resources to the collection and display layers, as suitable for your requirements. The one-, two-, or three-tiered configuration is a prerequisite for configuring high availability; at a minimum, your system must be configured for failover and failback in the aggregation layer.

Your Tivoli Netcool/OMNIBus installation includes a set of customizations, which you can apply to your ObjectServers and ObjectServer Gateways to configure each layer. These customizations are provided in the `$NCHOME/omnibus/extensions/multitier` and `$NCHOME/omnibus/extensions/control_shutdown` directories.

Related concepts

Chapter 8, “Configuring and deploying a multitiered architecture,” on page 251

Failover configuration

The failover configuration is a requirement for high availability, and is based around the aggregation layer of the standard multitiered architecture. In its simplest configuration, the failover configuration consists of a primary and a backup ObjectServer that are connected by a bidirectional ObjectServer Gateway in the aggregation layer, with no collection or display layers connected.

The following figure illustrates a failover configuration.

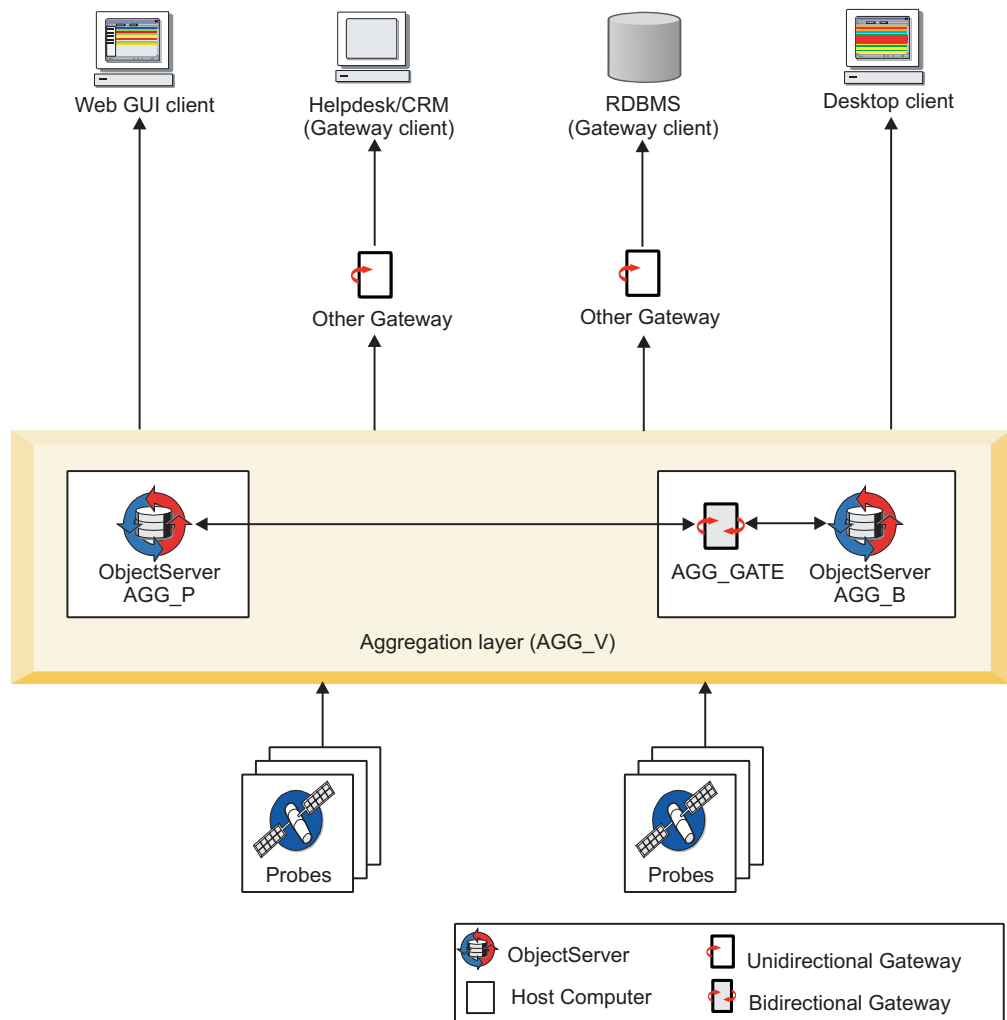


Figure 11. Basic failover configuration

In the figure, the aggregation pair of ObjectServers is connected by a bidirectional ObjectServer Gateway to keep the ObjectServers synchronized, and the bidirectional ObjectServer Gateway runs on the backup host. Probes connect directly to the virtual aggregation pair (AGG_V) to facilitate fail over and fail back if the primary aggregation ObjectServer computer becomes unavailable. Alternative targets to which alerts can be forwarded from the aggregation layer are also shown:

- A dedicated unidirectional ObjectServer Gateway for a display layer ObjectServer can connect to the virtual aggregation pair, and alerts can be forwarded to the desktop or Web GUI clients.
- Other gateways can connect to the virtual aggregation pair and forward alerts to clients such as a helpdesk or Customer Relationship Management (CRM) system, and a relational database management system (RDBMS).
- Alerts can be forwarded directly to the desktop or Web GUI clients.

If you want to set up the failover configuration shown in the aggregation layer of the preceding figure, only a subset of the steps that are required for setting up the standard multitiered architecture apply. The required steps for configuring failover in the aggregation layer are as follows:

1. “Configuring server communication information (multitiered architecture)” on page 260
2. “Installing the primary aggregation ObjectServer” on page 261
3. “Installing the backup aggregation ObjectServer” on page 262
4. “Configuring the bidirectional aggregation ObjectServer Gateway” on page 263

Related concepts

“Overview of the standard multitiered architecture” on page 251

Configuring controlled failback of clients

To minimize event loss, which can occur if clients fail back to a primary ObjectServer before resynchronization is completed, client failback behavior must be controlled by a failover pair of ObjectServers instead of the clients themselves. The configuration for controlled failback is based around the aggregation layer in the multitiered architecture.

Before you begin

Multitiered setup: The default multitiered architecture is preconfigured with the required values for controlled failback of the northbound gateways; that is, the unidirectional gateways that connect the collection layer to the aggregation layer, and the aggregation layer to the display layer, are configured for controlled failback.

Other failover pair setup: If you have set up the failover pair in the aggregation layer as directed, the automations that are required for controlled failback will be in place:

- The `backup_startup`, `backup_counterpart_down`, and `backup_counterpart_up` triggers are enabled in the backup ObjectServer.
- The `disconnect_all_clients` trigger is also enabled in the backup ObjectServer.

To configure controlled failback for clients that connect to the failover pair, perform the following steps for the client types:

- **Probes:** Probes connecting to the collection failover pair, must use the standard failover and failback property settings. In the probe properties file:
 - Set **Server** to `NCOMS_P`.
 - Set **ServerBackup** to `NCOMS_B`.
 - Set **NetworkTimeout** to 15
 - Set **PollServer** to 60.

Where `NCOMS_B` is the backup ObjectServer for `NCOMS_P`, and the **PollServer** value must be greater than the **NetworkTimeout** value.

- **Unidirectional ObjectServer Gateways:**
 - If unidirectional gateways are configured to connect from the collection to the aggregation (virtual) pair, disable failback in the unidirectional collection layer gateways (`C_TO_A_GATE_P_1` and `C_TO_A_GATE_B_1`). In each collection ObjectServer Gateway properties file:
 - Set **Gate.Writer.Server** to `AGG_V`.
 - Set **Gate.Writer.FailbackEnabled** to `FALSE`.
 - If unidirectional gateways are configured to connect the aggregation (virtual) pair to the display ObjectServers, disable failback in the unidirectional display

layer gateways (A_TO_D_GATE_P_1 and A_TO_D_GATE_B_1). In each display ObjectServer Gateway properties file:

- Set **Gate.Reader.Server** to AGG_V.
- Set **Gate.Reader.FailbackEnabled** to FALSE.
- **Bidirectional ObjectServer Gateways:** Although not part of the proposed multitiered configuration, if bidirectional ObjectServer Gateways are used between the collection layer and aggregation layer, disable failback in the bidirectional ObjectServer Gateways by using the following properties:
 - Set **Gate.ObjectServerB.Server** to AGG_V.
 - Set **Gate.ObjectServerB.FailbackEnabled** to FALSE.
- **Event lists:** If event lists are connecting to the aggregation failover pair, disable failback for event lists by setting the `-failbackpolltime` command-line option to 0 when running **nco_event** on UNIX and Linux, or **NC0Event.exe** on Windows. If event lists are configured to connect to the display layer ObjectServers, and the event lists make a dual-write connection to the aggregation failover pair, start the event lists with the `-failbackpolltime` command-line option set to 0 so that they exhibit controlled failback behavior.

Results

When failback is disabled for the clients, they remain connected to the backup ObjectServer until the backup ObjectServer forcefully disconnects them when resynchronization is completed.

To indicate that resynchronization is complete, the ObjectServer Gateway sends a `gw_resync_finish` signal to both the primary and backup ObjectServers. On receipt of this signal, the backup ObjectServer disconnects the clients so that they can connect to the resynchronized primary ObjectServer.

Related concepts

Chapter 8, “Configuring and deploying a multitiered architecture,” on page 251
“Naming conventions for the multitiered architecture” on page 253

Reducing event loss on ObjectServer failure during resynchronization

To minimize event loss when an ObjectServer fails during resynchronization, an ObjectServer property **ActingPrimary** is used to define which ObjectServer was acting as the primary if the last resynchronization was not successful. The bidirectional ObjectServer Gateway determines the direction of the resynchronization by using the **ActingPrimary** property of the ObjectServer. This property setting is updated solely by automations, and requires no user intervention.

For further information about how the ObjectServer Gateway determines which ObjectServer is the master of the resynchronization, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*. Go to the *IBM Tivoli Network Management* Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>. Locate the *IBM Tivoli Netcool/OMNIBus* node in the left navigation pane and go to the *Tivoli Netcool/OMNIBus gateways* node.

Reducing resynchronization time

To reduce the time taken to resynchronize the contents of one ObjectServer to another after the recovery of a primary or backup ObjectServer, or a bidirectional ObjectServer Gateway, you can configure the gateway to resynchronize only those events that have changed since the failure occurred.

You can use the **Gate.Resync.Type** property to specify the type of resynchronization that is required. Set **Gate.Resync.Type** to **Minimal** to configure the gateway to resynchronize only events that were inserted or updated into the source ObjectServer after the other ObjectServer or the gateway failed.

For further information about how the ObjectServer Gateway performs resynchronization, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*. Go to the *IBM Tivoli Network Management Information Center* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>. Locate the *IBM Tivoli Netcool/OMNIBus* node in the left navigation pane and go to the *Tivoli Netcool/OMNIBus gateways* node.

Details about the last time at which IDUC changes were passed to an IDUC client prior to a failure are stored in the `iduc_system.iduc_stats` table.

Configuring controlled shutdown of an ObjectServer

You can configure a controlled shutdown of any ObjectServer such that pending changes are forwarded to IDUC clients before the ObjectServer shuts down. This minimizes the possibility of data loss on shutdown.

To enable controlled shutdown, the ObjectServer schema must be updated with a set of triggers and procedures that are provided in an SQL import file `control_shutdown.sql`, which is stored in the `$NCHOME/omnibus/extensions/control_shutdown` directory. The ObjectServer must also be set up to run under process control because the **nco_pa_stop** utility needs to be called from an external procedure to shut down the ObjectServer.

The triggers and procedures that are provided in the `control_shutdown.sql` file will orchestrate a controlled shutdown. The ObjectServer is first brought to a restricted state. Connections identified for non-IDUC clients (such as **nco_sql** and **nco_config**) are dropped, and an IDUC FLUSH command is initiated to send pending changes to all identified IDUC clients (such as gateways and event lists). Any new connection requests to the ObjectServer are blocked. If store and forward is enabled for probes, any new alerts are stored in a store-and-forward file until the probe can successfully reconnect to an ObjectServer. When the data retrieval is completed for the IDUC clients, the **nco_pa_stop** utility is used to shut down the ObjectServer process that is running under process control. For further information about store and forward, see the *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*. For information about configuring the ObjectServer to run under process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Sections of the `control_shutdown.sql` file must be edited to specify information that is required for setting up the configuration.

To configure and perform a controlled shutdown for an ObjectServer:

1. Go to the `$NCHOME/omnibus/extensions/control_shutdown` directory, and copy the `control_shutdown.sql` file to the `$NCHOME/omnibus/etc` directory, or to another preferred location.

- Remove the default read-only permissions from your copy of the `control_shutdown.sql` file and review the file to familiarize yourself with its contents. Then edit the file as follows:

- Locate the following section of code for the `ext_shutdown` procedure:

```
-----
-- External procedure to shutdown OS using nco_pa_stop
...
-----

create or replace procedure ext_shutdown (in process_name Char(255),
    in username Char(255), in pass Char(255), in paserver Char(255))
executable '$OMNIBUS/bin/nco_pa_stop'
host 'nchost1'
user user1 group grp1
arguments ' -process ' + process_name + ' -user ' + username + ' -password ' + pass
    + ' -server ' + paserver
go
```

Replace the `nchost1` placeholder with the name of the host on which you want to run the **nco_pa_stop** utility to shut down the ObjectServer. Replace `user1` with the appropriate user ID and replace `grp1` with the group ID under which to run **nco_pa_stop**. The `ext_shutdown` procedure is called from the `control_shutdown` procedure and the final shutdown trigger in the `control_shutdown.sql` file, so these sections of code also need to be amended.

- Locate the following section of code for the `control_shutdown` procedure:

```
-----
-- Procedure to drop and flush connections for controlled shutdown
-----

create or replace procedure control_shutdown()
declare
    iduc_clients int;
...
...
...
if ( iduc_clients = 0 )
then
-- do nothing , simply shutdown
-- call external procedure to shutdown OS using nco_pa_stop
-- Replace 'MasterObjectServer' with ObjectServer process name in PA conf file
-- Replace 'user1' with username to execute nco_pa_stop.
-- Replace 'pass1' with password to execute nco_pa_stop.
-- Replace 'AGG_PA' with PA server name to connect.
execute procedure ext_shutdown ( 'MasterObjectServer', 'user1', 'pass1', 'AGG_PA' );
else
-- Enable trigger to check if GET IDUC is finished for all the clients.
execute procedure enable_control_shutdown;
end if;
```

On the `execute procedure ext_shutdown` line, replace the `MasterObjectServer`, `user1`, `pass1`, and `AGG_PA` placeholders with the ObjectServer process name defined in the process agent configuration file, the user credentials for running **nco_pa_stop**, and the name of the process agent that the ObjectServer uses to run the external automation.

- Locate the following section of code for the `final_shutdown` trigger:

```
-----
-- Trigger to reset Pending flag on GET IDUC from all clients
-----

create or replace trigger final_shutdown
group control_shutdown_triggers
...
...
...
if( pending_cnt = 0 ) then
-- disable this trigger group and shutdown ObjectServer.
execute procedure disable_control_shutdown;
-- call external procedure to shutdown OS using nco_pa_stop
-- Replace 'MasterObjectServer' with ObjectServer process name in PA conf file
-- Replace 'user1' with username to execute nco_pa_stop.
-- Replace 'pass1' with password to execute nco_pa_stop.
-- Replace 'AGG_PA' with PA server name to connect.
execute procedure ext_shutdown ( 'MasterObjectServer', 'user1', 'pass1', 'AGG_PA' );
end if;
```

On the execute procedure `ext_shutdown` line, replace the `MasterObjectServer`, `user1`, `pass1`, and `AGG_PA` placeholders with the ObjectServer process name defined in the process agent configuration file, the user credentials for running `nco_pa_stop`, and the name of the process agent that the ObjectServer uses to run the external automation.

3. Apply the controlled shutdown customization to a new or existing ObjectServer as follows:

- When creating the ObjectServer by using the `nco_dbinit` command, apply the customization to the ObjectServer database:

```
$NCHOME/omnibus/bin/nco_dbinit -server server_name -customconfigfile
$NCHOME/omnibus/extensions/control_shutdown/control_shutdown.sql
```

- If the ObjectServer already exists, apply the customization as follows:

```
UNIX Linux $NCHOME/omnibus/bin/nco_sql -server server_name
-user user_name -password password < $NCHOME/omnibus/extensions/
control_shutdown/control_shutdown.sql
```

```
Windows "%NCHOME%\omnibus\bin\isql" -S server_name -U user_name -P
password -i "%NCHOME%\omnibus\extensions\control_shutdown\
control_shutdown.sql"
```

In these commands, `server_name` is the name of the ObjectServer, `user_name` is a valid user name used to log on to the ObjectServer, and `password` is the corresponding password.

4. Set up the ObjectServer to run under process control. Within the ObjectServer properties file, set the following properties for process control:

- Set **PA.Name** to the name of the process agent that the ObjectServer uses to run external automations.
- Set **PA.Username** and **PA.Password** to a valid user name and password combination required for connecting to a process agent to run the `ext_shutdown` procedure.

For information about configuring the ObjectServer to run under process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

5. Start the process agent that runs the ObjectServer process under process control.
6. Assuming clients (both IDUC and non-IDUC) have been started within your environment, you can perform a controlled shutdown at any time by running the following SQL commands from the SQL interactive interface:

```
execute procedure control_shutdown;
go
```

Note: After flushing all the IDUC clients, the ObjectServer waits for a GET IDUC response from the notified clients. If any IDUC client does not respond, the ObjectServer remains in a restricted state. In this state, only the SQL interactive interface utility (`nco_sql`) is allowed to connect. You can query the ObjectServer by using `nco_sql` to check which client is not responding. If a client does not respond within the expected time, you can forcefully disconnect the client and try to execute the `control_shutdown` procedure again. For example:

```
select ConnectionId, AppName, AppDesc from iduc_system.temp_connections where Pending =1 ;
alter system drop connection 'connectionid';
execute procedure control_shutdown;
go
```

For information about the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Sample process agent configuration file: AGG_PA.conf

This example shows sample values in a process agent configuration file named AGG_PA.conf, which map to the placeholder values that you need to complete in the control_shutdown.sql file (as discussed in the preceding steps).

The code shows sample values in the AGG_PA.conf file, for the process agent named AGG_PA. In the file, an ObjectServer process called MasterObjectServer has been set up to run the AGG_P ObjectServer under process control. The user credentials user1 and pass1 will be used to connect to AGG_PA in order to run the external procedure ext_shutdown to shut down the ObjectServer, which is running on host nchost1.

```
=====
Example Process agent config file  AGG_PA.conf
=====
#
# Process Agent Daemon Configuration File 1.1
#
#
# List of Processes.
#
nco_process 'MasterObjectServer'
{
    Command '$OMNIHOME/bin/nco_objserv -name AGG_P -pa AGG_PA -pausername user1 -papassword pass1 run as 0
    Host      =      'nchost1'
    Managed   =      True
    RestartMsg =      '${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
}

nco_service 'InactiveProcesses'
{
    ServiceType =      Non-Master
    ServiceStart =      Non-Auto
}

#
# Routing Table Entries.
#
# 'user'      - (optional) only required for secure mode PAD on target host
#              'user' must be member of UNIX group 'ncoadmin'
# 'password'  - (optional) only required for secure mode PAD on target host
#              use nco_pa_crypt to encrypt.
nco_routing
{
    host 'nchost1' 'AGG_PA' 'user1' 'pass1'
}
}
```

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Configuring proxy servers for failover

The proxy server failover setup requires the Tivoli Netcool/OMNIbus basic failover architecture, and the following additional components: a primary proxy server and a backup proxy server.

The following figure shows the configuration for proxy server failover.

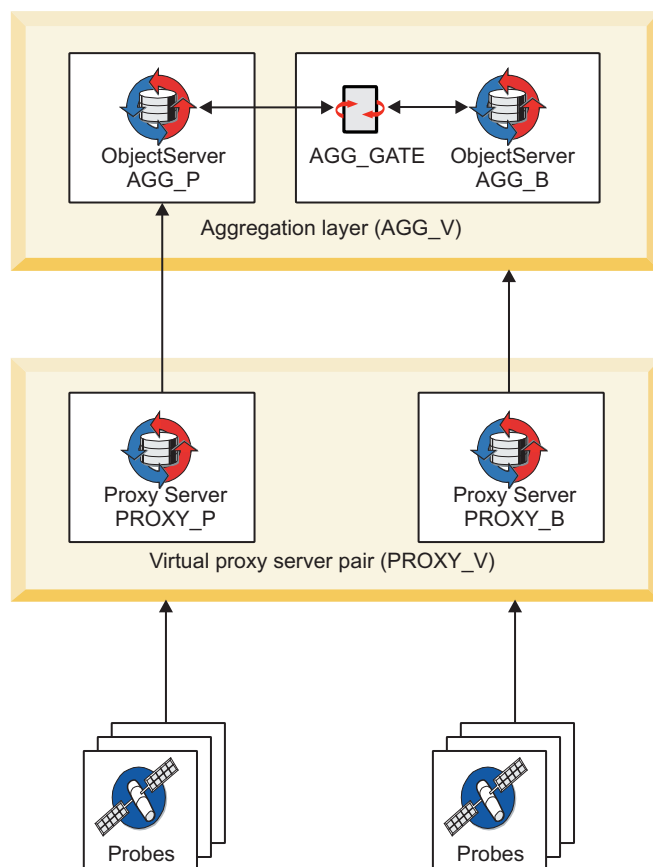


Figure 12. Proxy server failover setup

In the basic failover setup, alert data from the primary aggregation ObjectServer is replicated in the backup aggregation ObjectServer through a bidirectional ObjectServer Gateway. If a connection to the primary aggregation ObjectServer fails, the clients attempt to connect to the backup aggregation ObjectServer. As shown in the figure, you must set up a virtual proxy server pair to which probes can connect. Set up the primary proxy server PROXY_P to have a single connection to the primary aggregation ObjectServer AGG_P. Set up the backup proxy server PROXY_B for failover by configuring PROXY_B to connect to the virtual ObjectServer pair AGG_V.

Using the architecture shown in the preceding figure, configure the proxy servers for failover as follows:

- In the connections data file (\$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini), set up the server communications details, using the following sample configuration as a guideline:

```
[AGG_P]
{
    Primary:      nchost1 10000
```

```

}
[AGG_B]
{
    Primary:      nchost2 10001
}
[AGG_GATE]
{
    Primary:      nchost2 10002
}
[AGG_V]
{
    Primary:      nchost1 10000
    Backup:       nchost2 10001
}

}
[PROXY_P]
{
    Primary:      nchost1 10003
}
[PROXY_B]
{
    Primary:      nchost2 10004
}
[PROXY_V]
{
    Primary:      nchost1 10003
    Backup:       nchost2 10004
}
}

```

- Configure probes to connect to the proxy servers. In the probe properties file:
 - Set **Server** to PROXY_V.
 - Set **ServerBackup** to "".
- Configure the proxy servers by setting the following properties:
 - In the primary proxy server PROXY_P properties file, set **RemoteServer** to AGG_P.
 - In the backup proxy server PROXY_B properties file, set **RemoteServer** to AGG_V.

Results

With this configuration, if AGG_P fails, PROXY_P also fails, but the probes are automatically connected to PROXY_B, which will in turn connect to AGG_B. If only PROXY_P fails, the probes will automatically connect to PROXY_B, and events will be sent to AGG_P, which is still up and running as the primary ObjectServer.

Chapter 10. Configuring FIPS 140–2 support for the server components

You can run the following server components in FIPS 140–2 mode: ObjectServers, process agents, proxy servers, and ObjectServer gateways. In this mode, the cryptographic functions of Tivoli Netcool/OMNIbus use cryptographic modules that have been FIPS 140–2 approved.

To operate Tivoli Netcool/OMNIbus in FIPS 140–2 mode, you must create a FIPS configuration file within your installation and then configure the server components for FIPS 140–2 mode.

If you want to use SSL for client and server communications, you must additionally enable FIPS 140–2 mode for the SSL communications.

Related tasks

“Notes for SSL connections in FIPS 140–2 mode” on page 380

Related reference

“Switching your installation to FIPS 140-2 mode” on page 299

Creating the FIPS configuration file

A FIPS configuration file is required for FIPS initialization. This file is called `fips.conf`, and is required on each computer where a server component is installed.

Before running the server components in FIPS 140–2 mode, create the FIPS configuration file as follows:

1. Create an empty text file called `fips.conf`.
2. Save this file in the relevant directory for your operating system:
 - UNIX: `$NCHOME/etc/security`
 - Windows: `%NCHOME%\ini\security`

What to do next

You must now configure the server components for FIPS 140–2 mode.

Configuring the server components for FIPS 140–2 mode

If the server components are configured with the required FIPS 140–2 settings, all connecting clients must connect with plain text passwords in order to meet the requirements for FIPS 140–2 mode. If a client uses property value encryption, the relevant encryption algorithm for FIPS 140–2 mode must also be used.

When you run server components that are configured for FIPS 140–2, they verify the existence of the `fips.conf` file and then verify that their relevant properties are set to the values required for FIPS 140–2 mode. Error messages are logged to the server log files if any properties are found to have non-FIPS 140–2 settings. In debug logging mode, confirmation for FIPS 140–2 mode is also logged.

When running a server component in both FIPS 140–2 mode and secure mode, authentication passwords for client applications are typically stored as follows:

- Proxy servers and probes store authentication passwords by using the **AuthPassword** property in the proxy server and probe properties files.
- Unidirectional ObjectServer gateways store authentication passwords by using the **Gate.Writer.Password** and **Gate.Reader.Password** properties in the properties file.
- Bidirectional ObjectServer gateways store authentication passwords by using the **Gate.ObjectServerA.Password** and **Gate.ObjectServerB.Password** properties in the properties file.
- The process agent configuration file can also store passwords for secure connections.

In FIPS 140–2 mode, you can either specify plain text passwords within these files, or specify passwords that are encrypted by running the **\$NCHOME/omnibus/bin/nco_aes_crypt** utility with a key file and specific cryptographic algorithm. If you are using encrypted passwords, you must also set properties that define the key file and algorithm within the proxy server, probe, and gateway properties files; these values are required for decrypting the passwords at run time, so that they can be sent to the server as plain text. In the case of the process agent, which does not make use of properties, you must specify command-line options for decrypting the passwords in the configuration file when you run **\$NCHOME/omnibus/bin/nco_pad**.

Note: Do not use the **nco_g_crypt**, **nco_pa_crypt**, and **nco_sql_crypt** utilities to encrypt passwords when running in FIPS 140-2 mode.

ObjectServer configuration for FIPS 140–2

To run an ObjectServer in FIPS 140–2 mode, the following configuration is required:

- Set the **PasswordEncryption** property of the ObjectServer to the AES setting.
- If you want to run the ObjectServer in secure mode, and want to encrypt passwords within the proxy server, probe, or gateway properties files, encrypt the passwords by running the **nco_aes_crypt** utility and use the **-c** command-line option to specify AES_FIPS as the encryption algorithm.

Process agent configuration for FIPS 140–2

To run a process agent in FIPS 140–2 mode, the following configuration is required:

- On UNIX, only Pluggable Authentication Modules (PAM) are supported for external authentication in FIPS 140–2 mode. When running the process agent with the **nco_pad** command, set the **-authenticate** command-line option to the PAM setting if you want to verify the credentials of a user or a remote process agent daemon.

On Windows, process agent connections are authenticated against the Windows user accounts, and no additional configuration is required for FIPS 140–2 mode.

- If you want to run process control utilities (such as **\$NCHOME/omnibus/bin/nco_pa_status**) with the **-user** and **-password** command-line options (login credentials), specify the passwords in plain text.
- If you want to run the process agent in secure mode, you are typically required to specify the following login credentials within the routing definition section of the process agent configuration file (**\$NCHOME/omnibus/etc/nco_pa.conf**):

- User name and password credentials for each host that connects to the process agent
- User name and password credentials for logging into a remote process agent (if required)

If you want to encrypt the passwords in the configuration file, run the **nco_aes_crypt** utility and use the **-c** command-line option to specify **AES_FIPS** as the encryption algorithm.

Proxy server configuration for FIPS 140–2

To run a proxy server in FIPS 140–2 mode, the following configuration is required:

- If you want to run the proxy server in secure mode, and want to encrypt passwords within the probe properties files, encrypt the passwords by running the **nco_aes_crypt** utility and use the **-c** command-line option to specify **AES_FIPS** as the encryption algorithm.
- Additionally, if the proxy server is connecting to an ObjectServer that is running in secure mode, and you want to encrypt the password within the proxy server properties file, encrypt the password by running the **nco_aes_crypt** utility and use the **-c** command-line option to specify **AES_FIPS** as the encryption algorithm.

Gateway configuration for FIPS 140–2

To run gateways in FIPS 140–2 mode, the following configuration is required:

- On UNIX, only Pluggable Authentication Modules (PAM) are supported for external authentication in FIPS 140–2 mode. When running a gateway, set the **Gate.UsePamAuth** property of the gateway to **TRUE** to use PAM authentication.
- If a gateway is connecting to an ObjectServer that is running in secure mode, and you want to encrypt the password within the gateway properties file, encrypt the password by running the **nco_aes_crypt** utility and use the **-c** command-line option to specify **AES_FIPS** as the encryption algorithm.

A note about the encryption algorithm options

When in FIPS 140–2 mode, you must use the **AES_FIPS** algorithm when encrypting passwords with the **nco_aes_crypt** utility. You can specify the algorithm as either **AES_FIPS**, or use its synonym **AES_CBC**, which yields the same result. For simplicity, only **AES_FIPS** is specified in the documentation.

When in non-FIPS 140–2 mode, you can specify an additional algorithm, **AES** or **AES_CFB1**. These are synonyms and yield the same result; for simplicity, only **AES** and **AES_FIPS** are specified in the documentation. The **AES** option is primarily for compatibility with the **AES** property encryption that is available in Tivoli Netcool/OMNIBUS V7.2, and use of the **AES_FIPS** algorithm is preferred.

Related reference

“Property value encryption” on page 356

Configuration requirements for connecting V7.2 or earlier clients to V7.2.1 or later servers in FIPS 140–2 mode

Tivoli Netcool/OMNIBus V7.2.1, or later, maintains backward compatibility with existing client applications when running in non-FIPS 140–2 mode. To operate in FIPS 140–2 mode, some configuration is required for V7.2 or earlier clients that require connection to servers running in secure mode.

The following table shows the compatibility between V7.2 or earlier clients, and V7.2.1 or later servers running in secure mode, and the configuration changes required for FIPS 140–2 mode.

Table 70. Compatibility between V7.2 or earlier clients, and V7.2.1 or later FIPS 140–2 servers in secure mode

V7.2, or earlier client	Compatible	Configuration changes for connecting in FIPS 140–2 mode
Unidirectional and bidirectional gateways	Yes	Gateways can authenticate and connect to a V7.2.1 or later ObjectServer running in secure mode, without any changes.
Process control client (nco_pad) and process control utilities (nco_pa_shutdown , nco_pa_start , nco_pa_status , and nco_pa_stop)	Yes	Clients can connect to a V7.2.1 or later process agent running in secure mode if the client application is started with the -nosecure option and a plain text password.
Conductor (nco) and event lists (nco_event and nco_elct)	No	Clients cannot connect to a V7.2.1 or later ObjectServer running in secure mode.
Probes	Yes	Probes can connect to a V7.2.1 or later ObjectServer running in secure mode if they are started with the -nosecurelogin option and a plain text password. Additionally, the AuthPassword property setting in the probe properties file must not be encrypted with the nco_crypt or nco_g_crypt utility.
SQL interactive interface (nco_sql)	Yes	Clients can connect to a V7.2.1 or later ObjectServer running in secure mode if they are started with the -nosecure option. Authentication fails if the -nosecure option is not specified.
Proxy server client (nco_proxyserv)	Yes	Proxy servers can connect to a V7.2.1 or later ObjectServer running in secure mode, without any changes.
Process agent client (nco_pad)	Yes	Process agents can connect to a V7.2.1 or later ObjectServer running in secure mode, without any changes.
Other clients	-	When the ObjectServer is in FIPS 140–2 mode, clients supplying encrypted passwords cannot connect to the ObjectServer.

Switching your installation to FIPS 140-2 mode

If you want to change your V7.3.1 installation to operate in FIPS 140-2 mode, you must follow the steps outlined in the FIPS 140-2 configuration checklist.

Note: Switching your V7.3.1 installation to operate in FIPS 140-2 mode will automatically change the scheme used to encrypt passwords from DES to the Advanced Encryption Standard (AES).

If the user passwords in your system are currently encrypted by using the DES algorithm, or if you are using property value encryption to encrypt string values in properties files, the configuration steps for FIPS 140-2 are described here.

Changing the encryption scheme for DES-encrypted user passwords

When in FIPS 140-2 mode, the Advanced Encryption Standard (AES) algorithm must be used to encrypt user passwords that are stored in the ObjectServer. If your existing installation uses DES encryption for passwords, you must change the encryption scheme to AES.

To establish whether your passwords are DES encrypted, check the value of the ObjectServer **PasswordEncryption** property to see whether it is set to DES or to AES.

To change the encryption scheme to AES:

1. Change the setting of the ObjectServer **PasswordEncryption** property to AES.
2. Ensure that all user passwords are changed or reset. The passwords are now AES encrypted. (See the information that follows for guidelines about how to change or reset passwords.)
3. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140-2 mode.
4. Restart Tivoli Netcool/OMNIBus.

Guidelines for changing or resetting passwords

You can use the SQL interactive interface (**nco_sql**) for changing or resetting passwords.

If you ask users to change their passwords, you must verify that the changes have been made, and will most likely have to send out reminders. To verify whether all passwords have been changed or to identify which ones still need to be changed, perform either of the following steps:

- Start the SQL interactive interface and then enter the following command:

```
select UserName,Passwd from security.users;
```

Check the length of the encrypted passwords returned. Passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

For information about starting the SQL interactive interface, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

- From Netcool/OMNIBus Administrator:
 1. Connect to the relevant ObjectServer. Then click the **System** menu button and click **Databases** to open the Databases, Tables and Columns pane.

2. Select the **security** database and the **users** table, and then click the **Data View** tab in the Databases, Tables and Columns pane to view user data. In the **Passwd** column, passwords that are still DES encrypted have 11 characters, whereas AES-encrypted passwords have 24 characters.

A system administrator can reset user passwords from the SQL interactive interface as follows:

```
alter user 'username' set password 'password';
```

Where *username* is the name of the user and *password* is their new password.

Changing property value encryption

When in FIPS 140–2 mode, property value encryption must be performed by using an algorithm and mode of operation defined as AES_FIPS. Property value encryption is used to encrypt string values in a properties file or configuration file so that the strings cannot be read without a key.

If your existing installation uses property value encryption with the AES algorithm, or uses the **nco_g_crypt** and **nco_pa_crypt** utilities to encrypt passwords, these encrypted values do not meet the requirements for FIPS 140–2 operation. To run your system in FIPS 140–2 mode, you must decrypt these values and then encrypt them again by using the AES_FIPS algorithm. You must perform this task for each ObjectServer, proxy server, process agent, probe, and gateway that uses encrypted property values, including passwords.

To change property value (and password) encryption for FIPS 140–2 mode, follow these guidelines:

1. In your existing installation, identify any keys that were generated by using the command-line key generator **nco_keygen**.

Tip: The **nco_keygen** utility stores keys within key files. You should be able to identify any key files used by checking the **ConfigKeyFile** property settings in your properties files.

2. Using the keys in your existing installation, decrypt all encrypted properties and passwords in your properties and configuration files by running the **nco_aes_crypt** utility with the **-d** command-line option.
3. Configure Tivoli Netcool/OMNIBus to operate in FIPS 140–2 mode.
4. Encrypt the values again by using the **nco_keygen** utility to generate one or more new keys, and then running the **nco_aes_crypt** utility with the relevant key file setting.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related reference

“Configuring the JRE for FIPS 140–2 mode (UNIX and Linux)” on page 102

“Configuring the JRE for FIPS 140–2 mode (Windows)” on page 158

“Property value encryption” on page 356

“nco_aes_crypt command-line options” on page 358

Chapter 11. Importing and exporting ObjectServer configurations

Tivoli Netcool/OMNIBus provides two utilities, called **nco_confpack** and **nco_osreport**, both of which you can use to import and export ObjectServer configurations.

The **nco_confpack** and **nco_osreport** utilities require the appropriate version of the Java Runtime Environment (JRE) to be installed on your system. The features that install these utilities use the JRE that is provided with Tivoli Netcool/OMNIBus.

nco_osreport

You can use the **nco_osreport** utility to perform the following tasks:

- Export the configuration of an ObjectServer to a series of SQL files that can be input into a new ObjectServer created by running the **nco_dbinit** command. By exporting the contents of an ObjectServer to SQL files, you can create a copy of the ObjectServer on different operating systems than the source ObjectServer. You can view and modify the SQL files before using them to create a new ObjectServer; you can also use the files to archive the ObjectServer contents in a form that is independent of operating system considerations. Additionally, you can use the exported SQL files to submit the contents of the ObjectServer, in a human-readable form, to a support team.
- Export the contents of ObjectServer tables to an HTML file to capture a snapshot of an ObjectServer configuration, for example to submit the configuration to a support team.
- Export the contents of ObjectServer tables to an XML file that can, for example, be used for programming tasks.

nco_confpack

You can use the **nco_confpack** utility to extract a subset of configuration objects (for example, event list menus, tool, triggers and procedures, and class numbers) from ObjectServers and import the objects into other existing ObjectServers. The **nco_confpack** utility is not suitable for importing entire ObjectServer configurations. To extract a configuration, you must run the **nco_confpack** utility on the installation of Tivoli Netcool/OMNIBus that contains the ObjectServer from which you want to extract the configuration. To import a configuration, you must run the **nco_confpack** utility on the installation of Tivoli Netcool/OMNIBus that contains the ObjectServer into which you want to import the configuration. The source and target ObjectServers can be on different installations of Tivoli Netcool/OMNIBus. You can export the configuration from one installation, send it to an installation on another server, and import the configuration to an ObjectServer on that installation.

Related concepts

"JRE requirements" on page 14

Exporting and importing ObjectServer configurations by using the `nco_osreport` utility

Use the **`nco_osreport`** utility to extract the content of ObjectServer tables into HTML, XML, or SQL files. You can use the extracted SQL files for cloning ObjectServers.

About the `nco_osreport` utility

The **`nco_osreport`** utility outputs the content of the tables of an ObjectServer into an HTML or XML file. It can also be used to export the configuration of an ObjectServer to a series of SQL files that can be used to create the initial contents of a new ObjectServer.

After you have run the utility with the `-dbinit` command-line option, you can use the **`nco_dbinit`** command to create a new ObjectServer with the contents of the exported ObjectServer.

The full path to the **`nco_osreport`** utility is `$NCOME/omnibus/bin/nco_osreport`.

SQL files created by the `nco_osreport` utility

After you have run the **`nco_osreport`** utility with the `-dbinit` command-line option, the following files are generated:

- `system.sql`: This file specifies the security database and tables, and system users, groups, roles, and permissions. You must not edit this file.
- `application.sql`: This file creates the initial tables for the alerts and tools databases.
- `alertsdata.sql`: The contents of non-system tables are exported to this file.

If you use the **`nco_dbinit`** utility to import the SQL files into a new ObjectServer, this file is used instead of the `application.sql` to create the default tables for the alerts and tools databases. If you do not want all the data in this file to be imported into a new ObjectServer, you can edit this file. Specify *both* the `-alertsdata` and the `-alertsdatafile alertsdata.sql` arguments. If you do not include both of these arguments, many tables will be left empty. The files generated by the **`nco_osreport`** utility include `alertsdata.sql` so that cross table references are maintained. It is common for cross table references to be based on columns of type INCR. The **`nco_dbinit`** utility usually reassigns values to INCR columns because it populates tables; this operation is suspended while it is reading `alertsdata.sql`.

- `desktop.sql`: This file specifies initial values for the desktop tables, including default colors, conversions, tools, and menus. This file is empty but is provided for completeness, because it is required by the **`nco_dbinit`** utility.
- `automation.sql`: This file creates the initial trigger groups, triggers, and procedures.

In this file, triggers and internal procedures are defined twice, once with an empty body and once with the value defined in the exported ObjectServer configuration. These duplicates ensure that when a reference from one automation to another is imported, the referenced automation is known to the target ObjectServer (that is, the ObjectServer to be created by the **`nco_dbinit`** utility).

- **security.sql**: This file specifies additional operator and administrator roles. The owners of entities in the source ObjectServer are assigned only permissions on the entities that they own in the exported ObjectServer, ownership passes to the root user.

Examples

To generate an HTML file that contains the content of the tables of an ObjectServer that is not defined in the interfaces file, and output the file into a specific directory, run the **nco_osreport** utility as follows:

```
$NCOME/omnibus/bin/nco_osreport -html -server NCOMS -user root
-password ' ' -directory /home/output/html
```

Related tasks

“Exporting ObjectServer configurations and cloning ObjectServers”

Related reference

“Command-line options for the **nco_osreport** command” on page 304

Exporting ObjectServer configurations and cloning ObjectServers

Use the command-line options of the **nco_osreport** utility to select the form in which the utility exports the contents of an ObjectServer. Contents that are exported into SQL files can be used to create a new ObjectServer by running the **nco_dbinit** command.

To export ObjectServer configurations, or use the exported files to create new ObjectServers:

- To export the tables of an ObjectServer to a single HTML file, run the **nco_osreport** utility with the **-html** option.
- To export the tables of an ObjectServer to a single XML file, run the **nco_osreport** utility with the **-xml** option.
- To export the configuration of an ObjectServer to a series of SQL files, and use the files for the creation of a new ObjectServer:
 1. Run the **nco_osreport** utility with the **-dbinit** option.
 2. Change to the directory into which the SQL files were output.
 3. To create a new ObjectServer based on the configuration of the exported ObjectServer, run the **nco_dbinit** utility as follows:

```
$NCHOME/omnibus/bin/nco_dbinit -server SERVERNAME
-systemfile system.sql -applicationfile application.sql
-alertsdata -alertsdatafile alertsdata.sql
-desktopfile desktop.sql -automationfile automation.sql
-securityfile security.sql
```

In this command, *SERVERNAME* is the name of the new ObjectServer that you want to create, and the *.sql* arguments are the names and paths of the files that are to be read by the **nco_dbinit** utility.

Important:

4. Add the communications details for the newly-created ObjectServer by running the **nco_xigen** utility on UNIX, or the Servers Editor on Windows.
5. Start the new ObjectServer.

What to do next

If you created a new ObjectServer, use the **nco_config** utility, the **nco_sql** utility, **nco_event** utility, or the Web GUI to check some of the data in that ObjectServer. If it appears that no data has been included, you might have forgotten to use both the **-alertsdata** and the **-alertsdatefile** *alertsdata.sql* arguments to the **nco_dbinit** utility.

Related concepts

"About the nco_osreport utility" on page 302

Related tasks

"After creating an ObjectServer" on page 236

"Starting an ObjectServer" on page 236

"Configuring server communication information" on page 242

Related reference

"Command-line options for the nco_osreport command"

"Properties and command-line options for nco_dbinit" on page 232

Command-line options for the nco_osreport command

Use the command-line options of the **nco_osreport** utility to specify the type of output required, and the ObjectServer that you want to export.

The following table describes the command-line options of the **nco_osreport** command. The command-line options **-dbinit**, **-xml**, and **-html** are mutually exclusive.

Table 71. Command-line options of the **nco_osreport** command

Command-line option	Description
-directory <i>string</i>	Specifies a directory into which the output of the utility is stored. If you do not use this option to specify a directory, the file or files are output to the current working directory
-dbinit	Default: Extracts the configuration of the specified ObjectServer and stores the configuration in a series of SQL files. The nco_dbinit utility can use these SQL files to import the configuration of the specified ObjectServer into a new ObjectServer.
-help	Displays help information about the command-line options.
-host <i>string</i>	If the required ObjectServer is not defined in the interfaces file, use this command-line option, together with the -port option, to specify the fully-qualified name of server on which the ObjectServer is installed.
-html	Extracts the contents of the tables of the specified ObjectServer into a single HTML file. By default, the output file is called <i>osreport.html</i> .
-password <i>string</i>	The password of the specified user.
-port <i>string</i>	The port number on which the ObjectServer installed on the server specified by the -hostname option listens for events.

Table 71. Command-line options of the `nco_osreport` command (continued)

Command-line option	Description
<code>-server string</code>	The name of the ObjectServer, as specified in the interfaces file, from which to extract the configuration or tables. Note: Either specify the <code>-server</code> command-line option, or the combination of the <code>-host</code> and <code>-port</code> command-line options.
<code>-user string</code>	The user name to connect to the ObjectServer specified by the <code>-server</code> command-line option.
<code>-timeout string</code>	Specifies the time, in milliseconds, that the utility waits for a response from the ObjectServer, where <i>string</i> is the amount of time. The default time is 6000 milliseconds (one minute).
<code>-version</code>	Displays software version information and exits.
<code>-xml</code>	Extracts the contents of the tables of the specified ObjectServer into a single XML file. By default, the output file is called <code>osreport.xml</code> .

Related concepts

“About the `nco_osreport` utility” on page 302

Related tasks

“Exporting ObjectServer configurations and cloning ObjectServers” on page 303

Exporting and importing ObjectServer configurations using the `nco_confpack` utility

Use the `nco_confpack` utility to extract ObjectServer configurations, deploy duplicate ObjectServer configurations, and back up existing ObjectServers.

Import and export terminology

A number of terms are used in the instructions for importing and exporting ObjectServer configurations.

These terms are as follows:

- *Source* and *target* ObjectServers: You export configuration objects from the source ObjectServer and import objects into the target ObjectServer.
- *Configuration list files*: These are text files that itemize the objects you can export from a ObjectServer. You can then select which objects you want to export from, or import into, an ObjectServer.
- *Configuration package*: When you export configuration objects from an ObjectServer, the objects are saved in a configuration package. You use the configuration package to import objects into a target ObjectServer. Configuration packages are saved as Java archive (`.jar`) files.

Importable and exportable objects

You can use the **nco_confpack** utility to import and export a number of ObjectServer objects.

These objects include:

- Triggers
- Trigger groups
- Procedures
- User-defined signals
- Menus
- Tools
- Prompts
- Classes
- Conversions
- Column visuals
- Colors
- Users
- Groups
- Roles
- Tables
- Indexes
- Views
- Restriction filters
- ObjectServer file definitions

Note: The **nco_confpack** utility does not import or export the granted permissions of tables which are considered to be system objects.

Tables considered to be system objects to roles created in the source ObjectServer, will not contain their granted permissions in the target ObjectServer, when they are imported or exported using the **nco_confpack** utility. The following standard tables are considered system objects:

- All standard tables in the catalog database
- All standard tables in the persist database
- All standard tables in the security database
- All standard tables in the transfer database.

Tables not considered to be system objects in the source ObjectServer, will still contain their granted permissions in the target ObjectServer, when they are imported or exported using the **nco_confpack** utility. The following standard tables are not considered system objects:

- All standard tables in the alerts database
- All standard tables in the iduc_system database
- All standard tables in the master database
- All standard tables in the precision database
- All standard tables in the service database
- All standard tables in the tools database.

Note:

- Standard tables are created when the ObjectServer is initialized.
- Trigger groups and prompts are exported indirectly, based on their association with triggers and tools. When triggers are exported, trigger groups are automatically exported. Likewise, when tools are exported, prompts are automatically exported.
- System objects, which include system users, system groups, system roles, and system signals, cannot be exported or imported.
- The ownership of the object in the source ObjectServer is not imported into the target ObjectServer. The user importing the object becomes the owner of the object in the target ObjectServer.

nco_confpack properties and command-line options

The **nco_confpack** utility includes a number of properties and command-line options. You need to specify additional *subcommands* for most command-line options.

The following table lists the properties and command-line options for **nco_confpack**.

Table 72. Command-line options and properties for **nco_confpack**

Command-line option	Property	Description
-contents <i>-subcommand parameter, ...</i>	N/A	Lists the contents of a configuration package.
-dumpprops	N/A	Displays system and nco_confpack properties.
-export <i>-subcommand parameter, ...</i>	N/A	Exports selected configuration objects from a source ObjectServer into a configuration package.
-help	N/A	Displays help for nco_confpack and exits.
-import <i>-subcommand parameter, ...</i>	N/A	Extracts objects from a configuration package and imports them into a target ObjectServer.
-list <i>-subcommand parameter, ...</i>	N/A	Creates a list of all exportable configuration objects in a source ObjectServer.
N/A	nc.home <i>string</i>	The full path to the Netcool home location. This property takes its value from NCHOME. The value can be overridden, but this is not recommended. The default is /opt/netcool.
N/A	omni.home <i>string</i>	The full path to the Tivoli Netcool/OMNIBus installation. This property takes its value from NCHOME/omnibus. The value can be overridden, but this is not recommended. The default is /opt/netcool/omnibus.
-version	N/A	Displays the program version and exits.

The default **nco_confpack** properties file is `$NCHOME/omnibus/etc/nco_confpack.props`. You can use a properties file with the `-list`, `-export`, `-contents`, and `-import` command-line options.

Tip: You can use the properties file as an alternative to entering subcommands on the command line. This is useful if, for example, you need to frequently export the same ObjectServer configuration.

In an unedited properties file, all properties are listed with their default values, commented out with a hash symbol (`#`) at the beginning of the line. A property and its corresponding value are separated by a colon (`:`). String values are surrounded by single, straight quotes.

You can use the properties file as a template and modify it for different purposes. For example, you may have one properties file for creating a list file, one for exporting configurations, and one for importing configurations. You can edit the property values using a text editor. To override the default, change a setting in the properties file and remove the hash symbol.

Note: Start comments on a new line; otherwise, property values will not be read correctly.

If you specify a setting on the command line, this overrides both the default value and the setting in the properties file.

Example: Using command-line options to list, export, and import configuration objects

This example gives an overview of how to use **nco_confpack**.

The following command creates the configuration list file `confpack.list`, which itemizes the objects in the ObjectServer named MASTER.

```
nco_confpack -list -file confpack.list -server MASTER -user root
```

Edit the configuration list file to remove objects you do not want to export. For example, if you only want to export menus and tools, remove all objects except menus and tools.

Next, create a configuration package. The following command exports the objects itemized in the configuration list file `confpack.list` and produces a configuration package called `menutools.jar`.

```
nco_confpack -export -file confpack.list -package menutools.jar -user root
```

You do not need to specify the ObjectServer name in the preceding command because the name is specified in the list file.

The following command imports the objects in the `menutools.jar` package into the TEST ObjectServer.

```
nco_confpack -import -package menutools.jar -user root -server TEST -nowarn
```

Related reference

“Example: Configuration list file” on page 313

Creating and editing configuration list files

Use configuration list files to view the exportable objects in an ObjectServer, and to select the objects that you want to export from, or import into, an ObjectServer.

When exporting a configuration, the list file determines which of the exportable objects is included in the configuration package. To select the objects that you want to export into a configuration package, you must edit the list file.

Related tasks

“Editing configuration list files” on page 312

Creating configuration list files

To create a configuration list file for an ObjectServer, enter the following command:

```
$NCHOME/omnibus/bin/nco_confpack -list [ -subcommand parameter, ... ]
```

In this command, *-subcommand parameter* can be any of the subcommands in the following table.

Table 73. Subcommands and corresponding properties for *nco_confpack -list*

Subcommand	Property	Description
-file <i>string</i>	confpack.list.name <i>string</i>	Path and file name of the output configuration list. The default is stdout.
-memstoredatadirectory <i>OSname:string</i> , <i>OSname2:string</i> , ...	objectserver.OSname .memstoredatadirectory <i>string</i>	Specifies an alternative database directory for each ObjectServer, where: <ul style="list-style-type: none">• <i>OSname</i> is the name of the ObjectServer.• <i>string</i> is the path that contains the ObjectServer database files. The default is <code>\$NCHOME/omnibus/db</code>.• <i>OSname</i> can be substituted with an asterisk (*) to indicate a general path for all ObjectServers for which an explicit path is not supplied.• If the path is the same for all ObjectServers, <i>OSname</i> can be omitted. For example: <code>-memstoredatadirectory string</code> You can have multiple entries in the same properties file for different ObjectServers. For example: objectserver.NCOMSA. memstoredatadirectory : path1 objectserver.NCOMSB. memstoredatadirectory : path2 Note: On Windows, if you want to specify a path that includes a drive letter, <i>do not</i> omit the <i>OSname</i> value because the drive letter will be interpreted as an ObjectServer name. For example, specifying <code>-memstoredatadirectory C:\MyDir</code> causes C to be interpreted as the ObjectServer name.

Table 73. Subcommands and corresponding properties for `nco_confpack -list` (continued)

Subcommand	Property	Description
<code>-password OSname:string, OSname2:string, ...</code>	<code>objectserver.OSname.password</code> <i>string</i>	<p>Login password for the ObjectServers, where:</p> <ul style="list-style-type: none"> <i>OSname</i> is the name of the ObjectServer <i>string</i> is the login password If the password is the same for all ObjectServers, <i>OSName</i> can be omitted <i>OSName</i> can be substituted with an asterisk (*) followed by a password to be used for all ObjectServers for which a password is not supplied If a user name and password are defined, but not for all ObjectServers, the remaining ObjectServers default to the current system user with no password <p>The default <i>OSname</i> is all ObjectServers running on the local machine.</p> <p>The default password is ''.</p> <p>You can have multiple entries in the same properties file for logging into different ObjectServers. For example:</p> <pre>objectserver.NCOMSA.password : pass1 objectserver.NCOMSB.password : pass2</pre>
<code>-propsfile string</code>	N/A	<p>Specifies the nco_confpack properties file. You can use the properties file instead of entering individual subcommands on the command line.</p> <p>The default properties file is <code>\$NCHOME/omnibus/etc/nco_confpack.props</code>.</p>
<code>-server OSname1, OSname2, ...</code>	<code>confpack.omnibus.servers</code> <i>OSname, OSname, ...</i>	<p>The ObjectServers from which to retrieve configuration information. You can only retrieve information from ObjectServers that are running on the local machine.</p> <p>The default is all ObjectServers running on the local machine.</p>
<code>-timeoutOSname:string, OSname2:string,...</code>	<code>objectserver.OSname.timeout</code> <i>string</i>	<p>Specifies the time, in milliseconds, that the utility waits for a response from the ObjectServer, where <i>OSname</i> is the name of the ObjectServer, and <i>string</i> is the amount of time.</p> <p>The default time is 6000 milliseconds (one minute).</p>

Table 73. Subcommands and corresponding properties for `nco_confpack -list` (continued)

Subcommand	Property	Description
<code>-user OSname:string, OSname2:string, ...</code>	<code>objectserver.OSname.user string</code>	<p>Login user name for the ObjectServers, where:</p> <ul style="list-style-type: none"> <code>OSname</code> is the name of the ObjectServer <code>string</code> is the login user name If the user name is the same for all ObjectServers, <code>OSName</code> can be omitted <code>OSName</code> can be substituted with an asterisk (*) to indicate a general user name for all ObjectServers for which an explicit user name is not supplied If a user name and password are defined, but not for all ObjectServers, the remaining ObjectServers default to the current system user with no password <p>The default <code>OSname</code> is all ObjectServers running on the local machine.</p> <p>The default user name is the current operating system user.</p> <p>You can have multiple entries in the same properties file for logging into different ObjectServers. For example:</p> <pre>objectserver.NCOMSA.user : fred objectserver.NCOMSB.user : rob</pre>

Example: Creating configuration list files:

This example depicts various ways in which you can create configuration list files.

The following command logs into the ObjectServer NCOMS as the current system user with no password and creates the configuration list file `/tmp/NCOMS_conf.txt`.

```
nco_confpack -list -server NCOMS -file /tmp/NCOMS_conf.txt
```

The following command logs into all running ObjectServers as the user fred with the password secret and creates the configuration list file `/tmp/NCOMS_conf.txt`.

```
nco_confpack -list -user fred -password secret -file /tmp/NCOMS_conf.txt
```

The following command logs into the ObjectServers NCOMS and NYC as the current system user with no password. The configuration list file displays on-screen (stdout).

```
nco_confpack -list -server NCOMS, NYC
```

For the following examples, assume there are three active ObjectServers: NCOMS1, NCOMS2, and NCOMS3.

The following command logs into ObjectServer NCOMS1 with the user name user1 and the password pass1, ObjectServer NCOMS2 with the user name user2 and the password pass2, and ObjectServer NCOMS3 as the current system user with no password.

```
nco_confpack -list -user NCOMS1:user1,NCOMS2:user2 -password NCOMS1:pass1,NCOMS2:pass2
```

The following command logs into ObjectServer NCOMS1 with the user name user1 and the password pass1, ObjectServer NCOMS2 with the user name seth and the password secret, and ObjectServer NCOMS3 with the user name seth and the password muse.

```
nco_confpack -list -user "NCOMS1:user1,*:seth" -password NCOMS1:pass1,NCOMS2:secret,NCOMS3:muse
```

The following command logs into ObjectServer NCOMS1 with the user name user1 and no password, ObjectServer NCOMS2 as the current system user with no password, and NCOMS3 as the current system user with no password.

```
nco_confpack -list -user NCOMS1:user1
```

The following command logs into all of the ObjectServers as the current system user with the password sesame.

```
nco_confpack -list -password sesame
```

Tip: Even if the password is specified on the command line, it does not appear in ps command output.

Example: Properties file for creating a configuration list file:

The following example properties file creates a configuration list file called NCOMS_NY_export_list.txt for the ObjectServer NCOMS_NY.

```
nc.home           : '/opt/netcool'
omni.home         : '/opt/netcool/omnibus'
license.file      : '27000@licenseA_NY&27000@licenseB_NY'
objectserver.NCOMS_NY.user : 'joe_ny'
objectserver.NCOMS_NY.password : 'jOE_4_NY'
confpack.list.name : 'NCOMS_NY_export_list.txt'
confpack.package.name : ''
confpack.omnibus.servers : 'NCOMS_NY'
```

Editing configuration list files

You can edit configuration list files to specify the objects to export from a source ObjectServer or import into a target ObjectServer.

To edit a configuration list file:

1. Create the configuration list file using the `-list` command-line option with **nco_confpack**.
2. Edit the file to remove the objects that you do not want to include in the configuration package. For example, you can remove all the Menu objects if you do not want to export menus from the source ObjectServer.
3. Save the file.

Results

You can use the edited configuration list file with the `-file` subcommand for the `-export` command-line option or the `-select` subcommand for the `-import` command-line option.

Related reference

“Creating configuration list files” on page 309

“Example: Configuration list file” on page 313

“Exporting configurations” on page 313

“Importing configurations” on page 320

Example: Configuration list file:

The following example partial configuration list file is created from a Tivoli Netcool/OMNIBus installation with two ObjectServers (NCOMS1 and NCOMS2) running on the same host.

ObjectServer	NCOMS1	Menu	AlertsMenu
ObjectServer	NCOMS1	Menu	AlertsMenu->&Ownership
ObjectServer	NCOMS1	Menu	AlertsMenu->&Prioritize
ObjectServer	NCOMS1	Menu	AlertsMenu->&Resolve
ObjectServer	NCOMS1	Menu	AlertsMenu->&Tools
ObjectServer	NCOMS1	Menu	AlertsMenu->Related E&vents
ObjectServer	NCOMS1	Menu	AlertsMenu->Related E&vents->&Far-End Events
ObjectServer	NCOMS1	Menu	AlertsMenu->Related E&vents->&Near-End Events
ObjectServer	NCOMS1	Menu	AlertsMenu->Task &List
ObjectServer	NCOMS1	Menu	ConductorMenu
ObjectServer	NCOMS1	Menu	MainEventListMenu
ObjectServer	NCOMS1	Menu	SubEventListMenu
ObjectServer	NCOMS1	Menu	SymbolToolsMenu
ObjectServer	NCOMS2	Tool	Acknowledged Action
ObjectServer	NCOMS2	Tool	Add to Task List
ObjectServer	NCOMS2	Tool	Assign Action
ObjectServer	NCOMS2	Tool	Change Severity
ObjectServer	NCOMS2	Tool	Deacknowledged Action
ObjectServer	NCOMS2	Tool	Delete Action
ObjectServer	NCOMS2	Tool	Group Action
ObjectServer	NCOMS2	Tool	Ping Tool
ObjectServer	NCOMS2	Tool	Prompted Ping Tool
ObjectServer	NCOMS2	Tool	Prompted Telnet Tool
ObjectServer	NCOMS2	Tool	Remove from Task List
ObjectServer	NCOMS2	Tool	Sample Tool
ObjectServer	NCOMS2	Tool	Show Related FE Node
ObjectServer	NCOMS2	Tool	Show Related FE Object
ObjectServer	NCOMS2	Tool	Show Related NE Node
ObjectServer	NCOMS2	Tool	Show Related NE Object
ObjectServer	NCOMS2	Tool	Suppress/Escalate
ObjectServer	NCOMS2	Tool	Takeownership Action
ObjectServer	NCOMS2	Tool	Telnet Tool

Exporting configurations

To extract a configuration, you must run the **nco_confpack** utility on the installation of Tivoli Netcool/OMNIBus that contains the ObjectServer from which you want to extract the configuration. An exported configuration is saved in a configuration package.

The source and target ObjectServers can be on different installations of Tivoli Netcool/OMNIBus. You can export the configuration from one installation, send it to an installation on another server, and import the configuration to an ObjectServer on that installation.

To create a configuration package, run the following command:

```
$NCHOME/omnibus/bin/nco_confpack -export -subcommand parameter, ...
```

In this command, *-subcommand parameter* can be any of the subcommands in the following table.

Table 74. Subcommands and corresponding properties for `nco_confpack -export`

Subcommand	Property	Description
<code>-file string</code>	confpack.list.name <i>string</i>	<p>Name of the configuration list file that itemizes the objects to export. If you use this subcommand, you cannot use the <code>-server</code> subcommand or confpack.omnibus.servers property.</p> <p>The default is to read input from stdin if you do not use either the <code>-file</code> or <code>-server</code> subcommand with <code>nco_confpack -export</code>.</p>
<code>-memstoredatadirectory</code> <i>OSname:string,</i> <i>OSname2:string, ...</i>	objectserver.OSname .memstoredatadirectory <i>string</i>	<p>Specifies an alternative database directory for each ObjectServer, where:</p> <ul style="list-style-type: none"> <i>OSname</i> is the name of the ObjectServer. <i>string</i> is the path that contains the ObjectServer database files. The default is <code>\$NCHOME/omnibus/db</code>. <i>OSname</i> can be substituted with an asterisk (*) to indicate a general path for all ObjectServers for which an explicit path is not supplied. If the path is the same for all ObjectServers, <i>OSname</i> can be omitted. For example: <code>-memstoredatadirectory string</code> <p>You can have multiple entries in the same properties file for different ObjectServers. For example:</p> <pre>objectserver.NCOMSA. memstoredatadirectory : path1 objectserver.NCOMSB. memstoredatadirectory : path2</pre> <p>Note: On Windows, if you want to specify a path that includes a drive letter, <i>do not</i> omit the <i>OSname</i> value because the drive letter will be interpreted as an ObjectServer name. For example, specifying <code>-memstoredatadirectory C:\MyDir</code> causes <code>C</code> to be interpreted as the ObjectServer name.</p>
<code>-package string</code>	confpack.package.name <i>string</i>	<p>Path and file name to which the configuration package is to be exported.</p> <p>The default is stdout.</p>

Table 74. Subcommands and corresponding properties for `nco_confpack -export` (continued)

Subcommand	Property	Description
<code>-password OSname:string, OSname2:string, ...</code>	objectserver.OSname.password <i>string</i>	<p>Login password for the ObjectServers, where:</p> <ul style="list-style-type: none"> <i>OSname</i> is the name of the ObjectServer. <i>string</i> is the login password. If the password is the same for all ObjectServers, <i>OSName</i> can be omitted. <i>OSName</i> can be substituted with an asterisk (*) to indicate a general password for all ObjectServers for which an explicit password is not supplied. If a user name and password are defined, but not for all ObjectServers, the remaining ObjectServers default to the current system user with no password. <p>The default <i>OSname</i> is all ObjectServers running on the local computer.</p> <p>The default password is ''.</p> <p>You can have multiple entries in the same properties file for logging into different ObjectServers. For example:</p> <pre>objectserver.NCOMSA.password : pass1 objectserver.NCOMSB.password : pass2</pre>
<code>-propsfile string</code>	N/A	<p>Specifies the nco_confpack properties file. You can use the properties file instead of entering individual subcommands on the command line.</p> <p>The default properties file is <code>\$NCHOME/omnibus/etc/nco_confpack. props</code>.</p>
<code>-rename OSname:OSpack, OSname2:OSpack,...</code>	confpack.export.rename <i>OSname:OSpack, OSname2:OSpack2, ...</i>	<p>Renames the source ObjectServers in the configuration package.</p> <ul style="list-style-type: none"> <i>OSname</i> is the name of the source ObjectServer. <i>OSpack</i> is the corresponding ObjectServer name in the configuration package. <p>For example you could rename the source ObjectServers NCOMS1 and NCOMS2 to PRIMARY and SECONDARY in the configuration package.</p>
<code>-server OSname1, OSname2, ...</code>	confpack.omnibus.servers <i>OSname, OSname, ...</i>	<p>ObjectServers from which to export configuration objects. If you use this subcommand, you cannot use the <code>-file</code> subcommand (or config.list.name property).</p> <p>You can only export data from ObjectServers that are running on the local machine.</p> <p>This subcommand exports all objects from the selected ObjectServers.</p>

Table 74. Subcommands and corresponding properties for `nco_confpack -export` (continued)

Subcommand	Property	Description
<code>-timeout OSname:string, OSname2:string,...</code>	<code>objectserver.OSname.timeout</code> <i>string</i>	<p>Specifies the time, in milliseconds, that the utility waits for a response from the ObjectServer, where <i>OSname</i> is the name of the ObjectServer, and <i>string</i> is the amount of time.</p> <p>The default time is 6000 milliseconds (one minute).</p>
<code>-user OSname:string, OSname2:string,...</code>	<code>objectserver.OSname.user</code> <i>string</i>	<p>Login user name for the ObjectServers, where:</p> <ul style="list-style-type: none"> • <i>OSname</i> is the name of the ObjectServer. • <i>string</i> is the login user name. • If the user name is the same for all ObjectServers, <i>OSname</i> can be omitted. • <i>OSname</i> can be substituted with an asterisk (*) to indicate a general user name for all ObjectServers for which an explicit user name is not supplied. • If a user name and password are defined, but not for all ObjectServers, the remaining ObjectServers default to the current system user with no password. <p>The default <i>OSname</i> is all ObjectServers running on the local computer.</p> <p>The default user name is the current operating system user.</p> <p>You can have multiple entries in the same properties file for logging into different ObjectServers. For example:</p> <pre>objectserver.NCOMSA.user : fred objectserver.NCOMSB.user : rob</pre>

Related concepts

“Creating and editing configuration list files” on page 309

Export considerations

When exporting a configuration package, take note of a number of considerations.

These considerations are as follows:

- Do not use the character combination `->` in menu names.
- The order in which submenu names are displayed in the configuration list file determines the order in which the names are displayed in the event list. To change the order, you can edit the list file.
- You must include all of the submenus of a menu, in the configuration list file; otherwise, an error occurs when you attempt to import the menu.
- You cannot export system objects, which include system users, system groups, system roles, and system signals.

- Any tools referring to prompts that do not have an entry in the `tools.prompt_defs` table are excluded from the export. This is because importing tools with non-existent prompts into a target ObjectServer will cause the desktop to fail.

About the exclusions file

Some ObjectServer objects, such as tools, triggers, and procedures, can contain references to external files that are part of a standard Tivoli Netcool/OMNIbus or operating system installation. These files do not need to be exported, and can be excluded from configuration packages.

The exclusions file contains a listing of files and directories to exclude when exporting any configuration package. The exclusions file name is `$NCHOME/omnibus/etc/exclusions.xml`.

The default exclusions file contains entries to prevent some standard files and directories from being included in configuration packages; however, you can edit this file to add entries.

The `exclusions.xml` file contains one element for files and directories in `$NCHOME/omnibus` (OmniHome) and one element for each supported operating system (Platform).

A forward slash (/) character is used as a generic path separator. The forward slash is replaced by the operating system-specific separator during processing. For any entries in the OmniHome element, paths must be relative to `$NCHOME/omnibus`. For Platform elements, paths must be relative to the system root.

For Windows systems, you must include the drive letter; for example, C: or D:. The C: drive is the default.

Example: Exclusions file:

This example shows the content of an `exclusions.xml` file.

```
<exclusions>
<OmniHome>
  <File Name="/utils/nco_functions"/>
  <File Name="/utils/nco_mail"/>
  <File Name="/desktop/default.elv"/>
  <File Name="/desktop/default.elc"/>
  <File Name="/desktop/minimal.elc"/>
  <File Name="/ini/default.elc"/>
  <File Name="/ini/default.elv"/>
  <File Name="/ini/tool.elf"/>
  <File Name="/ini/minimal.elc"/>
  <File Name="/desktop/NC0elct.exe"/>
  <Dir Name="/bin"/>
  <Dir Name="/db"/>
  <Dir Name="/etc"/>
  <Dir Name="/install"/>
  <Dir Name="/log"/>
  <Dir Name="/platform"/>
  <Dir Name="/patches"/>
</OmniHome>

<Platform Name="SunOS" Type="UNIX">
  <Dir Name="/bin"/>
  <Dir Name="/etc"/>
  <Dir Name="/lib"/>
```

```

        <Dir Name="/sbin"/>
        <Dir Name="/usr"/>
    </Platform>

    <Platform Name="AIX" Type="UNIX">
        <Dir Name="/bin"/>
        <Dir Name="/etc"/>
        <Dir Name="/lib"/>
        <Dir Name="/lpp"/>
        <Dir Name="/sbin"/>
        <Dir Name="/usr"/>
    </Platform>

    <Platform Name="HP-UX" Type="UNIX">
        <Dir Name="/bin"/>
        <Dir Name="/etc"/>
        <Dir Name="/lib"/>
        <Dir Name="/sbin"/>
        <DIR Name="/usr"/>
    </Platform>

    <Platform Name="Linux" Type="UNIX">
        <Dir Name="/bin"/>
        <Dir Name="/etc"/>
        <Dir Name="/lib"/>
        <Dir Name="/sbin"/>
        <Dir Name="/usr"/>
    </Platform>

    <Platform Name="Windows 2000" Type="WIN">
        <Dir Name="C:/WINDOWS"/>
        <Dir Name="C:/WINNT"/>
    </Platform>

    <Platform Name="Windows 2003" Type="WIN">
        <Dir Name="C:/WINDOWS"/>
        <Dir Name="C:/WINNT"/>
    </Platform>

    <Platform Name="Windows XP" Type="WIN">
        <Dir Name="C:/WINDOWS"/>
        <Dir Name="C:/WINNT"/>
    </Platform>
</exclusions>

```

Creating a backup configuration

You can use **nco_confpack** to export a backup configuration package so that, should a problem occur with your Netcool/OMNIBus installation, you can import the configuration package to restore your ObjectServer configuration.

Note: This does not back up any ObjectServer alert data.

Related reference

“Importing configurations” on page 320

Example: Exporting configuration packages

This example shows various ways of using **nco_confpack** to export configuration packages from ObjectServers.

The following command exports all configuration objects from the ObjectServer NCOMS to the configuration package /tmp/NCOMS_package. It logs into the ObjectServer as the current system user with no password.

```
nco_confpack -export -server NCOMS -package /tmp/NCOMS_package
```

The following command logs into the ObjectServer specified in the list file as the current system user with no password. It then exports the objects indicated in the list file /tmp/listfile1.txt as a configuration package /tmp/NCOMS_package.

```
nco_confpack -export -file /tmp/listfile1.txt -package /tmp/NCOMS_package
```

The following command logs into the ObjectServers NCOMS and NCOMS2 as the current system user with no password. It exports all configuration objects from the ObjectServers into the configuration package /tmp/NCOMS_package.

```
nco_confpack -export -server NCOMS,NCOMS2 -package /tmp/NCOMS_package
```

The following command logs into NCOMS1 as user1 with the password pass1 and to NCOMS2 as user2 with the password pass2. It exports all configuration objects from the ObjectServers into the configuration package /tmp/NCOMS_package.

```
nco_confpack -export -server NCOMS,NCOMS2 -user NCOMS:user1,NCOMS2:user2 -password NCOMS:pass1,NCOMS2:pass2 -package /tmp/NCOMS_package
```

The following command logs into the ObjectServer NCOMS as the current system user with no password. It creates a configuration package in /tmp/NCOMS_package. In the configuration package, the name of the source ObjectServer (NCOMS) is replaced with MYSERVER.

```
nco_confpack -export -server NCOMS -rename NCOMS:MYSERVER -package /tmp/NCOMS_package
```

Example: Properties file for exporting a package file:

This example properties file exports a configuration package from the ObjectServer NCOMS_NY using the configuration list file NCOMS_NY_export_list.txt to identify which objects to export.

```
nc.home : '/opt/netcool'
omni.home : '/opt/netcool/omnibus'
license.file : '270000licenseA_NY&270000licenseB_NY'
objectserver.NCOMS_NY.user : 'joe_ny'
objectserver.NCOMS_NY.password : 'jOE_4_NY'
confpack.list.name : 'NCOMS_NY_export_list.txt'
confpack.package.name : 'NCOMS_NY_export.pak.jar'
confpack.omnibus.servers : 'NCOMS_NY'
confpack.export.rename : ''
```

Viewing configuration package contents

You can use **nco_confpack** to view the contents of an exported configuration package or to save the contents of the package to a text file. This is useful to check which objects you can import from the package into an ObjectServer.

To view the contents of a configuration package or to save the contents of the package to a text file, enter the following command:

```
$NCHOME/omnibus/bin/nco_confpack -contents -subcommand parameter, ...
```

In this command, *-subcommand parameter* can be any of the subcommands in the following table.

Table 75. Subcommands and corresponding properties for **nco_confpack -contents**

Subcommand	Property	Description
-file <i>string</i>	confpack.list.name <i>string</i>	Path and file name for the output text file. The default is stdout.
-package <i>string</i>	confpack.package.name <i>string</i>	Configuration package path and file name. The default is stdin. Note: If you do not use the -package subcommand to specify a file name, nco_confpack will wait indefinitely for information from stdin.

Related concepts

“Creating and editing configuration list files” on page 309

Example: Viewing configuration package contents

This example shows how to use **nco_confpack** to write the contents of a configuration package to stdout or a text file.

The following command writes the contents of the configuration package /tmp/NCOMS_package to stdout.

```
nco_confpack -contents -package /tmp/NCOMS_package
```

The following command writes the contents of the configuration package /tmp/NCOMS_package to the text file /jsmith/package1.txt.

```
nco_confpack -contents -package /tmp/NCOMS_package -file /jsmith/package1.txt
```

Importing configurations

To import a configuration, you must run the **nco_confpack** utility on the installation of Tivoli Netcool/OMNIBus that contains the ObjectServer into which you want to import the configuration. You can import a configuration package into any V7.3.1 ObjectServer. You can import information from a configuration package into only one ObjectServer at a time.

The source and target ObjectServers can be on different installations of Tivoli Netcool/OMNIBus. You can export the configuration from one installation, send it to an installation on another server, and import the configuration to an ObjectServer on that installation.

Note: If an object in the configuration package has the same name as an object of the same type in the target ObjectServer, the existing object is replaced with the object from the configuration package. For example, a tool named `sample` in the configuration package will overwrite an existing tool named `sample` in the target ObjectServer. To avoid data loss, make sure that you back up your ObjectServer before importing a configuration package.

To import a configuration package, run the following command:

```
$NCHOME/omnibus/bin/nco_confpack -import -subcommand parameter, ...
```

In this command, *-subcommand parameter* can be any of the subcommands in the following table.

Table 76. Subcommands and corresponding properties `nco_confpack -import`

Subcommand	Property	Description
<code>-force TRUE FALSE</code>	<code>confpack.import.force TRUE FALSE</code>	<p>By default, when importing a tool or stored procedure that references an external file (such as a script) the file is included during the import. If an identical file already exists, the existing file is <i>not</i> overwritten.</p> <p>You can use the <code>-force</code> option to force an overwrite even if the referenced file already exists.</p> <p>Attention: Use the <code>-force</code> subcommand with caution. Make sure your file system is backed up before importing the configuration package.</p> <p>The default is <code>FALSE</code>.</p>
<code>-from string</code>	<code>confpack.import.from string</code>	<p>The source ObjectServer name, as indicated in the configuration package from which you are importing configuration objects.</p> <p>Note: If the configuration package contains information for only one ObjectServer, you do not need to use this subcommand. If the configuration package contains information for multiple ObjectServers, this subcommand is required.</p> <p>The default is <code>' '</code>.</p>

Table 76. Subcommands and corresponding properties *nco_confpack -import* (continued)

Subcommand	Property	Description
<code>-memstoredatadirectory</code> <i>OSname:string,</i> <i>OSname2:string, ...</i>	objectserver.OSname .memstoredatadirectory <i>string</i>	<p>Specifies an alternative database directory for each ObjectServer, where:</p> <ul style="list-style-type: none"> <i>OSname</i> is the name of the ObjectServer. <i>string</i> is the path that contains the ObjectServer database files. The default is <code>\$NCHOME/omnibus/db</code>. <i>OSname</i> can be substituted with an asterisk (*) to indicate a general path for all ObjectServers for which an explicit path is not supplied. If the path is the same for all ObjectServers, <i>OSname</i> can be omitted. For example: <code>-memstoredatadirectory string</code> <p>You can have multiple entries in the same properties file for different ObjectServers. For example:</p> <pre>objectserver.NCOMSA. memstoredatadirectory : path1 objectserver.NCOMSB. memstoredatadirectory : path2</pre> <p>Note: On Windows, if you want to specify a path that includes a drive letter, <i>do not</i> omit the <i>OSname</i> value because the drive letter will be interpreted as an ObjectServer name. For example, specifying <code>-memstoredatadirectory C:\MyDir</code> causes C to be interpreted as the ObjectServer name.</p>
<code>-nowarn</code> TRUE FALSE	confpack.import.nowarn TRUE FALSE	<p>Suppresses warning messages.</p> <p>Note: If you use stdin to import the configuration package, you must enable this option.</p> <p>The default is FALSE.</p>
<code>-package</code> <i>string</i>	confpack.package.name <i>string</i>	<p>Name of the configuration package.</p> <p>The default is stdin.</p>
<code>-password</code> <i>string</i>	objectserver.OSname.password <i>string</i>	<p>Login password for the ObjectServer.</p> <p>The default password is ''.</p>
<code>-propsfile</code> <i>string</i>	N/A	<p>Specifies the nco_confpack properties file. You can use the properties file instead of entering individual subcommands on the command line.</p> <p>The default properties file is <code>\$NCHOME/omnibus/etc/nco_confpack.props</code>.</p>

Table 76. Subcommands and corresponding properties *nco_confpack -import* (continued)

Subcommand	Property	Description
<code>-select string</code>	confpack.import.select <i>string</i>	Specifies a file containing a subset of objects to be extracted from the package and imported into the ObjectServer. Create the file by using the <code>-list</code> command-line option. Edit the file to remove all entries except for the ones that you want to import into the ObjectServer. The default is <code>' '</code> .
<code>-server OSname</code>	confpack.omnibus.servers <i>OSname</i>	The ObjectServer into which you are importing configuration objects. You can only import data into an ObjectServer that is running on the local machine. The default ObjectServer is NCOMS.
<code>-timeoutOSname:string, OSname2:string,...</code>	objectserver.OSname.timeout <i>string</i>	Specifies the time, in milliseconds, that the utility waits for a response from the ObjectServer, where <i>OSname</i> is the name of the ObjectServer, and <i>string</i> is the amount of time. The default time is 6000 milliseconds (one minute).
<code>-user string</code>	objectserver.OSname.user <i>string</i>	Login user name for the ObjectServer. The default user name is the current operating system user.

Related concepts

“Creating and editing configuration list files” on page 309

Related reference

“Viewing configuration package contents” on page 320

Import considerations

When importing a configuration package, take note of a number of considerations and guidelines.

These considerations and guidelines are as follows:

- When using stdin to import the configuration package, you must use the `-nowarn` subcommand. This is because **nco_confpack** cannot read the configuration package and prompt the user at the same time.
- When importing user, group, and role information, **nco_confpack** attempts to use the user, group, or role ID from the source ObjectServer. If an ID is already in use on the target ObjectServer, the next available ID is used. ObjectServer IDs do not have to match operating system user IDs.
- If you attempt to import a user that belongs to a group that does not exist in the configuration package or on the target system, an error occurs and the user is not imported.
- If you import a user that already exists on the target ObjectServer, but in a different group, that user will belong to both groups and will assume the

permissions that are assigned to the group with the highest permission level. To ensure that the user has the correct permissions, you might need to delete the user from one of the groups.

- You become the owner of any object that you import into an ObjectServer. The ownership of the object in the source ObjectServer is not imported into the target ObjectServer.
- Permissions that are associated with an ObjectServer object are implicitly imported with the object; for example, the ability for a user to run SQL commands.
- You cannot import a class from the source ObjectServer if the target ObjectServer contains a class that is defined with the same ID, but a different name.
- You cannot import a user-defined signal if a signal with the same name, but different parameters, exists in the target ObjectServer.
- If you import a trigger that references an object, make sure that the object either exists on the target system or is included in the configuration package. If you do not, an error occurs during the import process.

Note: Back up your target Tivoli Netcool/OMNIbus installation before importing a configuration package.

Related tasks

“Creating a backup configuration” on page 318

Examples of importing configuration packages

These examples show how to use the **nco_confpack** utility to import configuration packages into ObjectServers.

Example: Importing full configuration packages:

This example shows various ways of importing a configuration package.

The following command imports the configuration package /tmp/NCOMS_package into the ObjectServer NCOMS. It logs into the ObjectServer as the current system user with no password.

```
nco_confpack -import -package /tmp/NCOMS_package -server NCOMS
```

The following command imports the configuration package /tmp/NCOMS_package into the ObjectServer NCOMS with the user name fred and the password secret.

```
nco_confpack -import -package /tmp/NCOMS_package -server NCOMS  
-user fred -password secret
```

The following command imports the configuration package /tmp/NCOMS_package into the ObjectServer MYSERVER. Only configuration objects from the ObjectServer NCOMS are imported.

```
nco_confpack -import -package /tmp/NCOMS_package -server MYSERVER -from NCOMS
```


Example: Importing part of a configuration package:

The following series of commands creates a configuration list file, and then imports part of the configuration package into a second ObjectServer. You edit the configuration list file in a text editor to specify the components to import into the second ObjectServer.

1. The following command writes the contents of the configuration package /tmp/NCOMS_package to the configuration list file /jsmith/package1.txt.

```
nco_confpack -contents -package /tmp/NCOMS_package -file /jsmith/package1.txt
```

2. Edit the file to leave only those lines corresponding to entries that are to be imported into the second ObjectServer. In this example, the following lines are left which correspond to custom tools menus:

```
ObjectServer  NCOMS1  Menu  AlertsMenu->&Custom Tools
ObjectServer  NCOMS1  Menu  AlertsMenu->&Custom Tools->&Far-End Events
ObjectServer  NCOMS1  Menu  AlertsMenu->&Custom Tools->&Near-End Events
```

3. The following command imports objects specified in the configuration list file /jsmith/package1.txt from the configuration package /tmp/NCOMS_package into the ObjectServer MYSERVER.

```
nco_confpack -import -package /tmp/NCOMS_package -server MYSERVER
-select /jsmith/package1.txt
```

Related tasks

“Editing configuration list files” on page 312

Example: Properties file for importing a package file:

This example properties file imports the configuration package NCOMS_NY_export.pak.jar into the ObjectServer NCOMS_LON.

```
nc.home          : '/opt/netcool'
omni.home        : '/opt/netcool/omnibus'
license.file     : '27000@licenseA_NY&27000@licenseB_NY'
objectserver.NCOMS_LON.user : 'joe_lon'
objectserver.NCOMS_LON.password : 'j0E789'
confpack.list.name : ''
confpack.package.name : 'NCOMS_NY_export.pak.jar'
confpack.omnibus.servers : 'NCOMS_LON'
confpack.import.nowarn : TRUE
confpack.import.force : FALSE
confpack.import.from : ''
confpack.import.select : ''
```

Chapter 12. Setting up desktop ObjectServers

You can configure a desktop ObjectServer architecture to reduce the load on ObjectServers that receive high numbers of events.

For example, this can occur when many regional ObjectServers send events to a central ObjectServer through unidirectional ObjectServer Gateways and many desktops connect directly to the central ObjectServer. If the ObjectServer becomes overloaded, unidirectional ObjectServer Gateways cannot insert all events into the ObjectServer. Desktops that are connected directly to the ObjectServer further increase the load, especially if a large number of desktops connect simultaneously.

Desktop ObjectServer architecture

You can use a desktop ObjectServer architecture to improve the performance of an ObjectServer that frequently experiences heavy loads.

The desktop ObjectServer architecture:

- Reduces the workload of the central ObjectServer by shifting the load to specialized desktop ObjectServers
- Improves desktop responsiveness; that is, the time between a desktop operator's action and its reflection in the desktop GUI
- Reduces the likelihood of the desktop GUI freezing
- Improves the end-to-end latency times in heavily-loaded, standard ObjectServer configurations
- Maintains high data integrity and consistency by simultaneously updating the master ObjectServer

The desktop ObjectServer architecture consists of a master ObjectServer and one or more desktop ObjectServers that share the duties normally performed by a single ObjectServer. Dual server desktops (DSDs) connect to a single, master ObjectServer when writing data, but read and display alert data from desktop ObjectServers. The main functionality of the DSD is the same as a standard desktop; for operators, the DSD behaves identically to a standard desktop.

The desktop ObjectServer architecture is shown in the following figure.

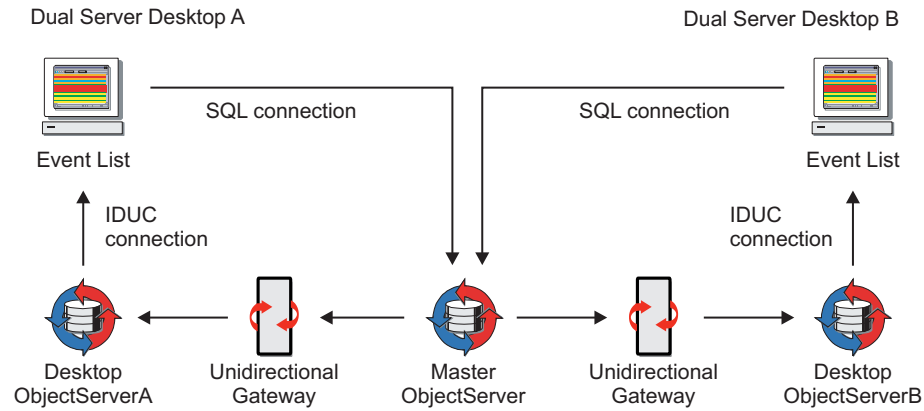


Figure 13. Example dual server desktop architecture

The DSD connects simultaneously to the desktop ObjectServer and the master ObjectServer. Any operator actions (for example, tools or journal actions) that are performed in the DSD are sent directly to the master ObjectServer through a one-way SQL connection.

Alerts in the master ObjectServer are sent to the DSD through the desktop ObjectServer. This is achieved by using a unidirectional ObjectServer Gateway from the master ObjectServer to the desktop ObjectServer, and an IDUC connection from the desktop ObjectServer to the DSD.

If dual-write mode is enabled, updates are also sent to the desktop ObjectServer through another one-way SQL connection, as shown in the following figure.

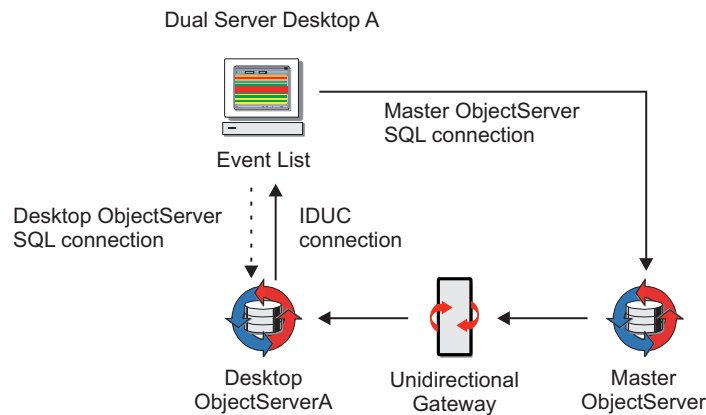


Figure 14. Example dual server desktop architecture with dual-write mode enabled

Related reference

“Viewing the results of tool actions using dual-write mode” on page 332

Considerations for setting up a desktop ObjectServer architecture

When configuring a desktop ObjectServer architecture, a set of guidelines is available for your consideration.

These guidelines are as follows:

- One ObjectServer must be designated as the master ObjectServer.
- One or more ObjectServers can be designated as desktop ObjectServers.
- A unidirectional ObjectServer Gateway must connect the master ObjectServer with each of the desktop ObjectServers.
- If you use the ObjectServer Gateway to replicate security data (including users, groups, roles, and restriction filters) between the master and desktop ObjectServers, you must maintain the security data in the master ObjectServer. Do not add ObjectServer objects directly to the desktop ObjectServers because when resynchronization occurs, any permissions in the desktop ObjectServers for objects maintained by the master ObjectServer are lost. Lost permissions can include permissions that are granted to roles, or roles that are granted to groups.

Attention: If you want to resynchronize security data when your ObjectServers are running in secure mode, run the gateway as the root user. If you fail to do this, when you attempt the resynchronization the gateway quits and the destination ObjectServer will have no security data. This is because the gateway deletes the destination permissions and so cannot insert rows copied from the source table. Running the gateway as the root user overcomes this problem because it does not require permissions to be set explicitly.

Configuring a desktop ObjectServer architecture

To set up a desktop ObjectServer architecture, you must create and configure a new desktop ObjectServer, and then configure a unidirectional ObjectServer Gateway to send the relevant data to the desktop ObjectServer.

Creating and configuring a desktop ObjectServer

You must run the database initialization utility with the relevant command-line options to create a desktop ObjectServer.

To create and configure a desktop ObjectServer:

1. To create the desktop ObjectServer, enter the appropriate command for your operating system:

Table 77. Creating a desktop ObjectServer

Option	Description
UNIX	<code>\$NCHOME/omnibus/bin/nco_dbinit -server <i>servername</i> -desktopserver -dsdprimary <i>masterservername</i> -dsddualwrite</code>
Windows	<code>%NCHOME%\omnibus\bin\nco_dbinit -server <i>servername</i> -desktopserver -dsdprimary <i>masterservername</i> -dsddualwrite</code>

In the above commands, *servername* is the name of the new desktop ObjectServer and *masterservername* is the name of the master ObjectServer.

The default SQL file used to create desktop ObjectServers is `$NCHOME/omnibus/etc/desktopserver.sql`. You can optionally use the `-desktopserverfile` command-line option to specify an alternative SQL file, in which case, the desktop ObjectServer is created using the commands in the SQL file you specify.

When you initialize the desktop ObjectServer, the master.national database table is created. This table identifies the master ObjectServer and the dual-write mode.

2. After creating the desktop ObjectServer, make sure you add it to the server definition file on any host running a dual server desktop.
3. Start the desktop ObjectServer.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Related tasks

“Starting an ObjectServer” on page 236

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Configuring the unidirectional ObjectServer Gateway

You must configure the unidirectional ObjectServer Gateway (**nco_g_objserv_uni**) to ensure that all relevant data is sent from the master ObjectServer to the desktop ObjectServer. To do this, you need to make some changes to the ObjectServer Gateway configuration files.

Note: For complete information about configuring the unidirectional ObjectServer Gateway, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*, SC14-7609.

To configure the unidirectional ObjectServer Gateway:

1. Create a directory for gateway files; for example, `$NCHOME/omnibus/gates/DSD_GATE`.

Tip: This task uses an example directory name of `$NCHOME/omnibus/gates/DSD_GATE`. Wherever referenced, replace this name with the actual directory name for your gateway files.

2. Copy all the files in `$NCHOME/omnibus/gates/objserv_uni` to `$NCHOME/omnibus/gates/DSD_GATE`.
3. Rename the `$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.props` file to `$NCHOME/omnibus/gates/DSD_GATE/DSD_GATE.props`.
4. In the `$NCHOME/omnibus/gates/DSD_GATE/DSD_GATE.props` file, modify the properties that are listed in the following table.

Table 78. Configuring unidirectional ObjectServer Gateway properties for the dual server desktop

Property	Description
Gate.MapFile	The location of the gateway map definition file. Set this to <code>\$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.map</code> .
Gate.Reader.Server	The name of the ObjectServer from which the gateway reads alerts; that is, the master ObjectServer.
Gate.Reader.Tbl.ReplicateDefFile	The path to the gateway table replication definition file. Set this to <code>\$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.reader.tblrep.def</code> .
Gate.StartupCmdFile	The path to the gateway startup command file. Set this to <code>\$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.startup.cmd</code> .
Gate.Writer.Server	The name of the gateway to which the ObjectServer writes alerts; that is, the desktop ObjectServer.

For example:

```
# Common Netcool/OMNIBus Properties.
MessageLog      : '$NCHOME/omnibus/log/DSD_GATE.log'

# Common Gateway Properties.
Gate.MapFile    : '$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.map'
Gate.StartupCmdFile : '$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.startup.cmd'

# Unidirectional ObjectServer Gateway Properties.
Gate.Reader.Server      : 'NETCOOLPRI'
Gate.Reader.Username    : 'root'
Gate.Reader.Password    : ''
Gate.Reader.Tbl.ReplicateDefFile:
                        $NCHOME/omnibus/gates/DSD_GATE/objserv_uni.reader.tblrep.def
Gate.Writer.Server      : 'DESKOS'
Gate.Writer.Username    : 'root'
Gate.Writer.Password    : ''
```

5. Copy the properties file `$NCHOME/omnibus/gates/DSD_GATE/DSD_GATE.props` to `$NCHOME/omnibus/etc`. The gateway looks for its properties file in this location.
6. In the `$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.map` file, add a `MasterSerial` entry to the end of the `StatusMap` mapping section. For example:

```
'URL'          = '@URL'          ON INSERT ONLY,
'ServerName'   = '@ServerName'   ON INSERT ONLY,
'ServerSerial' = '@ServerSerial' ON INSERT ONLY,
'MasterSerial' = '@Serial'       ON INSERT ONLY
);
```

This entry provides unique identification of the events in the master ObjectServer.

Note: The preceding steps provide you with a working unidirectional ObjectServer Gateway for use within the dual server desktop architecture. However, if you stop here, any configuration changes made to the master ObjectServer or any of the desktop ObjectServers must be made manually in the other ObjectServers in the system. For example, if you add a user to the master ObjectServer, you must manually add the user to the desktop ObjectServers to ensure that the user can log into the DSD. Also, any changes to desktop ObjectServer configurations, such as tools and conversions, will not be properly forwarded by the gateway. If you plan on making future changes to any of the ObjectServers in the DSD, you must complete the remaining steps.

7. Edit the file `$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.reader.tblrep.def` to specify which tables in the master ObjectServer are replicated in the desktop ObjectServer. The file contains commented table replication entries; for example:

```
# REPLICATE ALL FROM TABLE 'security.users'
#   USING MAP 'SecurityUsersMap'
#   INTO 'transfer.users';
```

Uncomment the appropriate table replication entries in the alerts, security, and tools sections of the file. To replicate all alerts, security, and tools table data, uncomment all entries.

You must also update the `objserv_uni.reader.tblrep.def` file to protect the permissions for the `master.national` table, which is present in the desktop ObjectServer, but not the master ObjectServer. Update the security section of the file as follows, to ensure that the permissions are not deleted during resynchronization:

```
REPLICATE ALL FROM TABLE 'security.role_grants'
  USING MAP 'SecurityRoleGrantsMap'
  INTO 'transfer.role_grants'
  RESYNC DELETES FILTER 'RoleID not in (3)';
```

```

REPLICATE ALL FROM TABLE 'security.permissions'
  USING MAP 'SecurityPermissionsMap'
  INTO 'transfer.permissions'
  RESYNC DELETES FILTER 'Object not in (\'master.national\')';

```

8. Edit the file \$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.map to define how to map the replicated data from the master ObjectServer to the desktop ObjectServer. The file contains commented mapping entries; for example:

```

# CREATE MAPPING SecurityUsersMap
# (
#   'UserID'      =      '@UserID'      ON INSERT ONLY,
#   'UserName'    =      '@UserName',
#   'SystemUser'  =      '@SystemUser',
#   'FullName'    =      '@FullName',
#   'Passwd'      =      '@Passwd',
#   'UsePAM'      =      '@UsePAM',
#   'Enabled'     =      '@Enabled'
# );

```

Uncomment all mapping entries in the file to correspond with the table data that you want to replicate.

9. Add an entry for the gateway to the Server Editor.

The unidirectional ObjectServer Gateway is now configured for use with the dual server architecture.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Viewing the results of tool actions using dual-write mode

Dual-write mode enables operators to quickly see the results of tool actions (for example, acknowledge and prioritize) from a DSD. When enabled, all tool actions are sent to both the desktop ObjectServer and the master ObjectServer.

When you initialize the desktop ObjectServer, you can enable or disable dual-write mode using the **nco_dbinit -dsddualwrite** command-line option.

You can enable or disable dual-write mode by changing the DualWrite column setting in the master.national table of the desktop ObjectServer. To enable dual-write mode, set the DualWrite column to 1. To disable dual-write mode, set the DualWrite column to 0. An example update command to enable dual-write mode is:

```
update master.national set DualWrite = 1;
```

You can also override the current dual-write mode setting using the event list **-dualwrite** command-line option.

Note: If you enable dual-write mode, there is a chance alert information may not be updated on all DSDs simultaneously. For example, this can be due to heavy network traffic. If you require all DSDs to always display identical information, disable dual-write mode.

Viewing operator journal entries from a dual server desktop

Alert journal entries made by an operator from a DSD alert are typically sent only to the master ObjectServer; however, if a selected alert is exclusive to the desktop ObjectServer (in which case it will have a MasterSerial value of 0), its manual journal entries are sent only to the desktop ObjectServer.

An alert is exclusive to the desktop ObjectServer if it is inserted into the desktop ObjectServer by any means other than the unidirectional ObjectServer Gateway (from the master ObjectServer to the desktop ObjectServer).

Desktop ObjectServer authentication

Tivoli Netcool/OMNIBus performs the following steps to authenticate a desktop ObjectServer:

1. Tivoli Netcool/OMNIBus checks for the existence of the MasterSerial column definition in the ObjectServer alerts.status table. If MasterSerial does not exist, the desktop enters standard mode and only connects to the desktop ObjectServer.
2. When an operator logs into the desktop, the desktop checks for the existence of the master.national table in the selected ObjectServer.
3. If the master.national table exists and has a valid entry in the MasterServer column, the desktop enters dual server desktop (DSD) mode. The DSD treats the selected ObjectServer as the desktop ObjectServer and the ObjectServer indicated in the MasterServer column as the master ObjectServer.

Note: The -masterserver command-line option for the event list overrides the MasterServer column.

If the desktop does not detect the master.national table in the selected ObjectServer, it enters standard ObjectServer mode.

4. The DSD attempts to authenticate with the master ObjectServer using the user name and password entered when the operator logged in (step 2).

Related reference

“Properties and command-line options for nco_dbinit” on page 232

Load balanced mode

In a configuration where there is a group of desktop ObjectServers, it is likely that the number of event list users logged into each desktop ObjectServer will not be even. In extreme cases, all users might be logged into one desktop ObjectServer, leaving the remaining desktop ObjectServers idle.

Load balanced mode automatically distributes event list user logins among a specified group of desktop ObjectServers according to a weighting that is specified by the administrator. This process is transparent to the event list user.

Configuring load balanced mode

Load balanced mode is configured as follows:

1. Determine a group of desktop ObjectServers among which event list connections should be distributed. These desktop ObjectServers must all be connected to the same master ObjectServer by a unidirectional ObjectServer Gateway.
2. Define each ObjectServer in the group as a Primary server either by using the Server Editor, or by editing the connections data file `$NCHOME/etc/omni.dat`.
3. Create entries in the `master.servergroups` table for all the desktop ObjectServers. The following table describes the column entries required for each desktop ObjectServer, within the `master.servergroups` table. You can use the Netcool/OMNIBus Administrator, as well as the INSERT command, to insert a row for each desktop ObjectServer.

Table 79. The `master.servergroups` table

Column	Data type	Description
ServerName	varchar(64)	The name of the desktop ObjectServer. This is the primary key.
GroupID	integer	The group to which each desktop ObjectServer belongs. Event list user logins are only distributed among desktop ObjectServers that have the same GroupID.
Weight	integer	The priority for each desktop ObjectServer. Higher values attract proportionally more connections. For example, an ObjectServer with a Weight of 2 attracts twice the number of connections as one with a Weight of 1. Load balanced connections are not redirected to ObjectServers with a Weight of 0.

Tip: You can configure the unidirectional ObjectServer Gateway so that the `master.servergroups` tables in the desktop ObjectServers are synchronized with the `master.servergroups` table in the master ObjectServer. For more information about configuring the unidirectional ObjectServer Gateway, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*, SC14-7609.

Results

Once this configuration is complete, event list connections to the desktop ObjectServers will be based on an algorithm that ensures an even distribution of connections to the desktop ObjectServers, in proportion to the weight specifications.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Related tasks

“Creating and configuring a desktop ObjectServer” on page 329

“Configuring the unidirectional ObjectServer Gateway” on page 330

“Manually editing the connections data file” on page 247

Example: Weighting

A system is set up with four desktop ObjectServers DISP_A, DISP_B, DISP_C, and DISP_D, where:

- DISP_A can support 1/6 of the connections.
- DISP_B can support 1/3 of the connections.
- DISP_C can support 1/2 of the connections.
- DISP_D is not available for load balanced connections.

DISP_A, supporting the least number of connections, is given the weight 1. DISP_B, supporting twice the number of connections as DISP_A, is given the weight 2. DISP_C, supporting three times the number of connections as DISP_A, is given the weight 3. DISP_D is not accepting load balanced connections so is given a weight of 0.

All of the ObjectServers are given the same GroupID so that connections can be redirected between them.

The insert commands to configure the master.servergroups table for this system are:

```
insert into master.servergroupsvalues ('DISP_A', 1, 1);
insert into master.servergroupsvalues ('DISP_B', 1, 2);
insert into master.servergroupsvalues ('DISP_C', 1, 3);
insert into master.servergroupsvalues ('DISP_D', 1, 0);go
```

Example: Load balanced groups

The system described in “Example: Weighting” is now extended to allow for two additional desktop ObjectServers: DISP_E and DISP_F.

These desktop ObjectServers can support the same number of connections between themselves, but do not share load balanced connections with the existing ObjectServers. DISP_E and DISP_F are assigned a GroupID of 2 and both have a weighting of 1.

The insert commands to configure the master.servergroups table for the additional desktop ObjectServers are:

```
insert into master.servergroupsvalues ('DISP_E', 2, 1);
insert into master.servergroupsvalues ('DISP_F', 2, 1);go
```

Related reference

“Example: Weighting”

Chapter 13. Tivoli Netcool/OMNIbus user access security

Tivoli Netcool/OMNIbus provides *user access security* mechanisms to protect your Tivoli Netcool/OMNIbus system from accidental or deliberate damage caused by users or potential users of your system.

Communication security techniques protect connections between different components of your Tivoli Netcool/OMNIbus system. Tivoli Netcool/OMNIbus uses the Secure Sockets Layer (SSL) security protocol to provide confidentiality, authenticity, and integrity of information between components.

Related concepts

Chapter 22, “Using SSL for client and server communications,” on page 377

User access security mechanisms

Tivoli Netcool/OMNIbus is a distributed system that can be used simultaneously by different types of users. In a typical deployment, operators use desktop tools to monitor the faults in a particular area of the network. At the same time, administrators use the SQL interactive interface, Netcool/OMNIbus Administrator, and process control to keep Tivoli Netcool/OMNIbus running smoothly and efficiently.

In a large deployment, it is typical to have different teams of users for different regions of deployment. For example, a Network Operation Center (NOC) in New York can be responsible for running and using the deployment of Tivoli Netcool/OMNIbus in the Americas, while another NOC in London can be responsible for the deployment in Europe.

With multiple users and types of users in multiple regions, it is important to control the rights that each user has to view and modify the Tivoli Netcool/OMNIbus system and the information it contains. To protect your system, you must consider the following user access areas:

- Authentication
- Authorization

Authentication

Authentication is the verification of the identity of a person.

For example, if someone tries to log in as administrator in your London NOC, it is important to first verify that they are who they claim to be and not an imposter.

All Tivoli Netcool/OMNIbus users have a user name with an associated password. Depending on the client application, you can specify the user name and password by using the properties file or command-line options, or by responding to a prompt from the application. The user name and password combination is authenticated before a connection is made to the ObjectServer. This access check validates that the user has a valid name and that the correct corresponding password is entered. Password encryption provides additional security.

Users can be authenticated in the ObjectServer, or externally authenticated. On UNIX and Linux systems, ObjectServer users can be externally authenticated by using Pluggable Authentication Modules (PAM) or a Lightweight Directory Access

Protocol (LDAP) system; the default mechanism is PAM. On Windows systems, users can be externally authenticated by using LDAP.

After logging in to an application, a user is allowed to proceed based on the groups to which the user belongs, and the roles that are assigned to these groups.

Related concepts

“PAM authentication (UNIX and Linux)” on page 345

“LDAP authentication” on page 340

“Secure mode authentication”

Authorization

Authorization is the verification of the rights to view and modify information.

For example, you might be authorized to create an automation in the ObjectServer for the London NOC, but not for the ObjectServer in the New York NOC.

Each ObjectServer object has actions associated with it. The ObjectServer stores information about the actions that each user is and is not authorized to perform on the system and for each object.

Administrators can allow and deny actions on the system and for individual objects by assigning permissions to roles, and granting and revoking roles for appropriate groups of users.

Administrators can also create restriction filters to limit the table data that a user or group can view.

Related concepts

“Implementing authorization by using groups and roles” on page 350

“Using restriction filters to filter table information” on page 355

Secure mode authentication

Tivoli Netcool/OMNIBus components perform authentication checks to provide a secure environment.

User authentication is enforced in the following ways:

- You can run the ObjectServer, proxy server, and process agent in secure mode. In this mode, probe and gateway connections to an ObjectServer or proxy server are authenticated with a user name and password. When the process agent is run in secure mode, connections are authenticated before external procedures are run. *Other client connection requests are always authenticated.*
- You can encrypt passwords that are stored in configuration and properties files. When in FIPS 140–2 mode, the password can either be specified in plain text, or can be encrypted with the **nco_aes_crypt** utility. If you are encrypting passwords by using **nco_aes_crypt** in FIPS 140–2 mode, you must specify AES_FIPS as the encryption algorithm.

When in non-FIPS 140–2 mode, the password can be encrypted with the **nco_g_crypt** or **nco_aes_crypt** utilities. If you are encrypting passwords by using **nco_aes_crypt** in non-FIPS 140–2 mode, you can specify either AES_FIPS or AES as the encryption algorithm. Use AES only if you need to maintain compatibility with passwords that were encrypted using the tools provided in versions earlier than Tivoli Netcool/OMNIBus V7.2.1.

Note: To prevent unauthorized users from gaining access, operating system security must be set appropriately for files such as configuration and properties files that might contain user names and passwords.

Related reference

“Property value encryption” on page 356

ObjectServer secure mode

You can run the ObjectServer in secure mode. When you specify the `-secure` command-line option, the ObjectServer authenticates probe, gateway, and proxy server connections by requiring a user name and password.

When a connection request is sent, the ObjectServer issues an authentication message. The probe, gateway, or proxy server must respond with the correct user name and password combination.

If the ObjectServer is not running in secure mode, probe, gateway, and proxy server connection requests are not authenticated.

Connections from other clients, such as the event list and the SQL interactive interface, are always authenticated.

Proxy server secure mode

You can run the proxy server in secure mode. When you specify the `-secure` command-line option, the proxy server authenticates probe connections by requiring a user name and password.

When a connection request is sent, the proxy server issues an authentication message. The probe must respond with the correct user name and password.

If the proxy server is not running in secure mode, probe connection requests are not authenticated.

Connecting securely from probes and gateways

When the ObjectServer or proxy server is running in secure mode, probe and gateway connections to the ObjectServer or proxy server are authenticated with a user name and password.

In addition, you can encrypt plain text login passwords that are stored in the gateway properties file. Passwords are decrypted by the target gateway and used to log in to the target system.

Process control security

You can run the process agent in secure mode. The `-secure` option controls the authentication of connection requests when running external procedures by verifying a user name and password on the local host.

The ObjectServer `-pa`, `-pausername`, and `-papassword` command-line options, and corresponding **PA.Name**, **PA.Username**, and **PA.Password** properties enable you to specify the process agent to connect to, and the user name and password to authenticate when running external procedures.

You can also specify that only certain hosts can connect to process agents by adding a security definition to the process agent configuration file. For each host

definition, you must also specify user name and password credentials for connecting to the process agent in secure mode, or you can optionally specify credentials for logging in to a remote process agent.

In addition, you can use the **nco_pa_crypt** or **nco_aes_crypt** utility to encrypt plain text login passwords that are stored in the process agent configuration file.

Related reference

“Property value encryption” on page 356

SQL interactive interface password protection in scripts

By default, the SQL interactive interface (**nco_sql**) encrypts login information when connecting to an ObjectServer in secure mode.

In addition, you can use the **nco_sql_crypt** utility (in non-FIPS 140–2 mode) to encrypt plain text login passwords so that they are not exposed in any scripts that run **nco_sql**.

LDAP authentication

Tivoli Netcool/OMNIBus supports external authentication of ObjectServer users whose passwords are stored in a Lightweight Directory Access Protocol (LDAP) compliant repository, such as Active Directory or Tivoli Directory Services.

You can configure the ObjectServer to act as an LDAP client, so that users connecting to the ObjectServer can have their passwords authenticated in the LDAP server. If configured for your environment, a single LDAP server can be maintained for all Tivoli Netcool/OMNIBus users, including users who access the desktop and Web GUI components, and the same user credentials can be used to access the system.

Restriction:

- Tivoli Netcool/OMNIBus is not intended to be used to manage user accounts in LDAP, and so does not provide the capability to change passwords in an LDAP server.
- The LDAP module used by the ObjectServer connects to a single LDAP server instance. The Web GUI component, which is deployed in the Tivoli Integrated Portal, can connect to multiple LDAP repositories.

Related tasks

“Adding an external LDAP repository” on page 481

Prerequisite information for LDAP configuration

Before attempting to configure Tivoli Netcool/OMNIBus, you need to obtain some LDAP configuration data from your LDAP administrator, and to verify that the required LDAP server configuration is in place.

Review the `$NCHOME/omnibus/etc/ldap.props` file, which holds the LDAP configuration settings, to determine the information required for communicating with the LDAP server. Obtain this configuration data from your LDAP administrator.

If setting up SSL connections between the ObjectServer and the LDAP server, take note that the ObjectServer acts as a client when it initiates an SSL connection with the LDAP server. The ObjectServer therefore needs to verify the signature on the

certificate presented by the LDAP server, and requires the public key of the issuing Certificate Authority (CA) to do this. Work with your LDAP administrator to obtain the self-signed root certificate issued by the CA; you will need to add this certificate to the ObjectServer key database that stores digital certificates.

If you have configured Tivoli Netcool/OMNIBus to operate in FIPS 140-2 mode, and with SSL, the LDAP interface must also be configured for FIPS 140-2 operation. Consult your LDAP administrator to verify that the required encryption support is in place for FIPS 140-2 operation.

Related tasks

“Configuring Tivoli Netcool/OMNIBus to use LDAP for external authentication”

Related reference

“LDAP properties” on page 343

Configuring Tivoli Netcool/OMNIBus to use LDAP for external authentication

You can configure ObjectServers to externally authenticate against user accounts that are managed in a centralized LDAP repository. A properties file named `$NCHOME/omnibus/etc/ldap.props` can be used to specify the configuration data that is required for authenticating against the external LDAP repository.

To configure LDAP:

1. Edit the `$NCHOME/omnibus/etc/ldap.props` file with the required LDAP properties settings.
2. Configure the ObjectServer to use LDAP authentication by setting the **Sec.ExternalAuthentication** ObjectServer property to LDAP. Authorization is managed in the ObjectServer.
3. To set up SSL connections between the ObjectServer and the LDAP server, complete the following actions:
 - On the ObjectServer, create a key database if one has not yet been created. The key database location is:
 - **UNIX** **Linux** `$NCHOME/etc/security/keys/omni.kdb`
 - **Windows** `%NCHOME%\ini\security\keys\omni.kdb`
 - Add the self-signed root certificate from the issuing CA of the LDAP server certificate, to the `omni.kdb` database.
 - In the `ldap.props` file, ensure that the SSL properties are configured:
 - Set the **SSLEnabled** property to TRUE.
 - Use the **SSLPort** property to specify a port number on which the LDAP server listens for SSL connections.
 - Use the **SSLKeyStoreLabel** property to specify the label of the certificate that the ObjectServer presents to the LDAP server.
4. Configure each Tivoli Netcool/OMNIBus user for external authentication by using any of the following methods:
 - From Netcool/OMNIBus Administrator, create or edit the user, ensuring that the following details are completed within the User Details pane:
 - **Username**: If creating a user, enter a user name that is identical to the name stored in the external authentication repository.
 - **Password** and **Verify**: Leave these fields blank. (Passwords are stored in the external repository.)
 - **External Authentication**: Select this check box.

- From the SQL interactive interface, use the CREATE USER command to create the user, or the ALTER USER command to edit the user. Ensure that:
 - The user name entered is identical to the name stored in the external authentication repository.
 - No password is specified.
 - The PAM keyword is set to TRUE.

What to do next

Optional: You can encrypt the LDAP password by using the **nco_keygen** utility and **nco_aes_crypt** utility. You can then reedit the \$NCHOME/omnibus/etc/ldap.props by setting the following properties: **ConfigCryptoAlg** (set this property to AES), **Hostname**, and **ConfigKeyFile**, and also **LDAPBindPassword** and **LDAPBindDN**.

Related tasks

“Creating a key database” on page 383

“Adding certificates from CAs” on page 387

Related reference

“LDAP properties” on page 343

“Property value encryption” on page 356

Additional configuration for Web GUI users

User accounts are required in the ObjectServer for all connecting clients. When running the Web GUI component, users can be externally authenticated in an LDAP server, but these users must have corresponding user accounts in the ObjectServer.

The Web GUI can be configured to synchronize the list of user accounts in the LDAP server with the user accounts held in the ObjectServer, and to create corresponding user accounts for any LDAP users who cannot be found in the ObjectServer.

Web GUI user accounts that are created in the ObjectServer by the synchronization process require additional configuration if the users need to use desktop tools such as the Conductor and event list:

- You must enable the users in the ObjectServer.
- You must add the users to the Normal group to ensure that they have been granted sufficient permissions to display and manipulate alerts in the event list, create filters and views, and run standard tools on alerts.

You can edit the user details either in Netcool/OMNIBus Administrator, or by using the ALTER USER and ALTER GROUP SQL commands. From Netcool/OMNIBus Administrator, access the User Details window, select the **User Enabled** check box on the **Settings** tab, and then use the **Groups** tab to assign the user to the Normal group. Alternatively, from the SQL interactive interface, run the ALTER USER command with ENABLED set to TRUE, and then run the ALTER GROUP command with the ASSIGN MEMBERS setting.

Related tasks

“Enabling user synchronization between a federated repository and the ObjectServer” on page 480

LDAP properties

Use the `$NCHOME/omnibus/etc/ldap.props` properties file to define configuration settings for connecting to an LDAP repository.

The LDAP properties are described in the following table. You must verify the value of all these properties with the LDAP administrator, with the exception of the **ConfigCryptoAlg**, **ConfigKeyFile**, and **SSLKeyStoreLabel** properties.

Tip: You can encrypt string values in a properties file by using property value encryption.

Table 80. LDAP properties

Property	Description
ConfigCryptoAlg <i>string</i>	<p>Specifies the cryptographic algorithm to use for decrypting string values (including passwords) that were encrypted with the nco_aes_crypt utility and then stored in the properties file. Set the <i>string</i> value as follows:</p> <ul style="list-style-type: none">• When in FIPS 140–2 mode, use AES_FIPS.• When in non-FIPS 140–2 mode, you can use either AES_FIPS or AES. Use AES only if you need to maintain compatibility with passwords that were encrypted by using the tools provided in versions earlier than Tivoli Netcool/OMNIBUS V7.2.1. <p>The value that you specify must be identical to that used when you ran nco_aes_crypt with the -c setting, to encrypt the string values.</p> <p>Use this property in conjunction with the ConfigKeyFile property.</p>
ConfigKeyFile <i>string</i>	<p>Specifies the path and name of the key file that contains the key used to decrypt encrypted string values (including passwords) in the properties file.</p> <p>The key is used at run time to decrypt string values that were encrypted with the nco_aes_crypt utility. The key file that you specify must be identical to the file used to encrypt the string values when you ran nco_aes_crypt with the -k setting.</p> <p>Use this property in conjunction with the ConfigCryptoAlg property.</p>

Table 80. LDAP properties (continued)

Property	Description
DistinguishedName <i>string</i>	<p>Specifies the distinguished name (DN) that identifies the user being authenticated in the target LDAP server. A sample format showing some of the attribute type-value pairs in the DN is:</p> <pre>cn=%s, o=string1, ou=string2, dc=string3, l=string4, st=string5, c=string6</pre> <p>Where:</p> <ul style="list-style-type: none"> • %s must be entered as shown, for the common name value. The %s will be replaced by the name of the user that is supplied to the ObjectServer for authentication. • string1 specifies your organization or company name. • string2 specifies the organizational unit or department name. • string3 specifies the domain component. • string4 specifies the locality or city of your organization. • string5 specifies your state or province. • string6 specifies the two-letter ISO code for your country. <p>Examples are:</p> <pre>cn=%s, ou=Development, o=ABCcorp</pre> <pre>cn=%s, ou=NOC, dc=ABCcorp, dc=com</pre> <pre>cn=%s, ou=Operators, ou=NOC, l=london, o=ABCcorp</pre> <p>The default is:</p> <pre>cn=%s</pre> <p>The attributes can be in uppercase or lowercase; for example, CN or cn. At a minimum, you must specify the common name setting (in the form cn=%s).</p>
Hostname <i>string</i>	<p>Identifies the name of the host on which the LDAP server is running, and to which the ObjectServer connects. Acceptable values are:</p> <ul style="list-style-type: none"> • A single host name. • A blank-separated list of host names, and optionally, port numbers, in the following format: host1[:port1] host2[:port2] ... <p>You might find this format useful for specifying a failover configuration. Connections are attempted in the order given for the host names and port numbers. When the ObjectServer successfully establishes a connection to an LDAP server, the ObjectServer remains connected to that server until the connection is no longer required, or fails. If a port number is not specified, the port number defined for the Port property is used.</p> <p>Example entries are:</p> <pre>Hostname: 'server1'</pre> <pre>Hostname: 'server2:1200'</pre> <pre>Hostname: 'server1:800 server2:2000 server3'</pre> <p>The default is localhost.</p>

Table 80. LDAP properties (continued)

Property	Description
LDAPVersion <i>integer</i>	Indicates the LDAP version that the server is running. Valid values are 2 and 3. The default is 3.
LDAPBindDn <i>string</i>	Specifies the distinguished name of the LDAP user account that is used for bind authentication. This value is used to establish a persistent connection to the LDAP server, and is used for subsequent authentication operations. The default is ''. <p>Use this property in conjunction with the LDAPBindPassword property.</p>
LDAPBindPassword <i>string</i>	Specifies the password for LDAP bind authentication. The default is ''. <p>Use this property in conjunction with the LDAPBindDn property.</p>
Port <i>integer</i>	Specifies the port on which the LDAP server is listening. The default is 389.
SSLEnabled TRUE FALSE	Determines whether SSL can be used for connections to the LDAP server. The default is FALSE. <p>On Windows only, if SSL is enabled for connections to the LDAP server, the following environment variable must be set for the ObjectServer to start successfully:</p> <p>GSKIT_LOCAL_INSTALL_MODE=true</p>
SSLKeyStoreLabel <i>string</i>	Specifies the label of the server certificate for the ObjectServer. This certificate is held in the Tivoli Netcool/OMNIBus key database, and can be presented to the LDAP server when client authentication is required. If this property is not set and SSL is enabled, server authentication is used. This property is applicable only when the SSLEnabled property is set to TRUE. <p>The default is ''.</p>
SSLPort <i>integer</i>	Specifies the port on which the LDAP server is listening for SSL connections. This property is applicable only when the SSLEnabled property is set to TRUE. <p>The default is 636.</p>

Related tasks

“Managing digital certificates for SSL communication” on page 380

Related reference

“Property value encryption” on page 356

PAM authentication (UNIX and Linux)

Pluggable Authentication Modules (PAM) is an integrated UNIX login framework. PAM is used by system entry components, such as the **dtlogin** display manager of the Common Desktop Environment, to authenticate users logging into a UNIX system.

PAM can also be used by PAM-aware applications for authentication. These applications include the ObjectServer, the process agent, and gateways.

You can set up PAM authentication either by using external authentication sources, such as NIS and LDAP, or by using a single, central ObjectServer.

Restriction: The PAM module that is provided in the Tivoli Netcool/OMNIBus installation *does not* support external authentication to a third-party authentication system. This PAM module is intended for use only when setting up a single, central ObjectServer as an authentication source.

Configuring Tivoli Netcool/OMNIBus to use PAM for external authentication

On UNIX and Linux, you can configure ObjectServers, process agents, and gateways to use a PAM framework with external authentication modules such as NIS and LDAP.

Before you begin

You must have PAM installed, as described in the documentation for the module being used.

Important: Different versions of UNIX and Linux use different methods of configuring PAM, so it is important that you refer to the documentation for your operating system when installing and configuring PAM externally.

In your operating system, you must define configuration values for the Tivoli Netcool/OMNIBus components (or services) that require authentication. On UNIX, a single file (/etc/pam.conf) is used for PAM configuration. On Linux, each PAM policy is typically held in a separate configuration file, which bears the service name of the associated component, and is stored in the /etc/pam.d/ directory. You must create a configuration file for each of the Tivoli Netcool/OMNIBus services, within this directory. If the /etc/pam.d/ directory does not exist on your Linux system, you can use the /etc/pam.conf file instead.

To enable external authentication between Tivoli Netcool/OMNIBus and PAM:

- Linux** Assuming the pam.d directory exists on your Linux system, create the following configuration files for the Tivoli Netcool/OMNIBus services. You can create each configuration file by copying /etc/pam.d/system-auth.
 - /etc/pam.d/nco_objserv: Required for the ObjectServer.
 - /etc/pam.d/netcool: Required for the process agent.
 - /etc/pam.d/gateway_name: Required for the gateway, where *gateway_name* represents the binary name of the gateway; for example, nco_g_objserv_uni or nco_g_objserv_bi.
- Update the PAM configuration file (/etc/pam.conf or /etc/pam.d/*service_name*) with the following entries for Tivoli Netcool/OMNIBus. If you are using separate configuration files on Linux, the service name is omitted in the PAM configuration file.

	Service name	Module type
Required for the ObjectServer	nco_objserv	auth, account, password
Required for the process agent	netcool	auth
Required for the gateway	gateway_name (where gateway_name is the binary name)	auth, account

Consult the documentation for the PAM module for additional configuration information required, such as the control flags, module paths, and options. A sample configuration is:

service	module_type	control_flag	module_path	options
nco_objserv	auth	required	pam_filename	

3. Configure the ObjectServer to use PAM authentication by setting the **Sec.ExternalAuthentication** ObjectServer property to PAM. On UNIX and Linux, the default is PAM.

When **Sec.ExternalAuthentication** is set to PAM, the ObjectServer can use the external authentication method specified in the system PAM configuration file for authentication and password management. However, it does not use PAM for authorization. Authorization is managed in the ObjectServer.

4. Configure the process agent to use PAM authentication. When you run the `$NCHOME/omnibus/bin/nco_pad` command, set the `-authenticate` command-line option to PAM.

Note: When you run the process control agent daemon (`nco_pad`) with PAM authentication on SUSE Linux, the default `nco_pad` stack size must be increased. To increase the `nco_pad` stack size to accommodate PAM authentication, run the `$NCHOME/omnibus/bin/nco_pad` command, specifying one of the following command-line options:

- `-stacksize 139248` (for SUSE Linux version 9.0)
 - `-stacksize 278496` (for SUSE Linux version 10.0).
5. Configure the gateway to use PAM authentication. For ObjectServer gateways or other gateways that support PAM authentication, set the **Gate.UsePamAuth** property to TRUE.

See the publications for the individual gateways to establish which gateways support PAM authentication. To view the publications, go to the IBM Tivoli Network Management Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>. Then expand the *IBM Tivoli Netcool/OMNIBus* node in the navigation pane on the left and go to the *Tivoli Netcool/OMNIBus gateways* node.

6. Configure each Tivoli Netcool/OMNIBus user for external authentication by using any of the following methods:
 - From Netcool/OMNIBus Administrator, create or edit the user, ensuring that the following details are completed within the User Details pane:
 - **Username:** Enter a user name that is identical to the name stored in the external authentication repository.
 - **Password** and **Verify:** Leave these fields blank. (Passwords are stored in the external repository.)
 - **External Authentication:** Select this check box.
 - From the SQL interactive interface, use the CREATE USER command to create the user, or the ALTER USER command to edit the user. Ensure that:
 - The user name entered is identical to the name stored in the external authentication repository.
 - No password is specified.
 - The PAM keyword is set to TRUE.

Related concepts

“Implementing authorization by using groups and roles” on page 350

Related reference

“User authentication failure with Pluggable Authentication Modules (PAM)” on page 624

Configuring an ObjectServer as a PAM authentication source

To configure an ObjectServer to control PAM authentication, you must set up your ObjectServers to defer authentication and password management to a single, central ObjectServer, and then enable external authentication for each ObjectServer and user on your system.

PAM modules implement authentication using, for example, LDAP, NIS or kerberos. In the rare situation where this is not available or not required, it is possible to use an ObjectServer itself as a method of authenticating users.

In this case, process agents and the ObjectServer will use another ObjectServer as a method of authenticating users. Process agents or the first ObjectServer use PAM, which is configured to use the ObjectServer PAM module, which in its turn connects to the other ObjectServer.

Once these tasks are completed, the ObjectServer can be used for performing PAM authentication.

Setting up an ObjectServer as the central authentication source

In order to recognize a central ObjectServer as an authentication source, each machine on which a client ObjectServer is running must have the ObjectServer PAM module installed. The central ObjectServer must also be defined within the system PAM configuration file, or within an ObjectServer PAM configuration file.

To install the ObjectServer PAM module on each client ObjectServer and configure a central ObjectServer for authentication:

1. Log in to your system as the root user.
2. Run the `$NCHOME/omnibus/install/nco_install_ospam` script.
3. When prompted, specify the name of the ObjectServer to be used as an authentication source. The `nco_install_ospam` script:
 - a. Installs the ObjectServer PAM module.
 - b. Updates your system PAM configuration file with auth, account, and password entries for the `nco_objserv` application. These entries configure the ObjectServer to use the ObjectServer PAM module for authentication and password management. On most UNIX platforms, the name of the system PAM configuration file is `/etc/pam.conf`.

Tip: On Linux platforms, each PAM policy is held in a separate configuration file that bears the service name of the associated application. As a result, the `nco_install_ospam` script creates a new file, `/etc/pam.d/nco_objserv`, with auth, account, and password entries.

You can optionally amend the ObjectServer entries in the system PAM configuration file.

- c. Creates an ObjectServer PAM configuration file called `$NCHOME/omnibus/etc/pam_omnibus_os.conf`. Your specified ObjectServer is defined within this file.

You can change which ObjectServer to use as an authentication source using this file, or by amending the ObjectServer entries in the system PAM configuration file.

Modifying your ObjectServer settings in the system PAM configuration file:

You can add arguments to the ObjectServer PAM module entries in the system PAM configuration file. This includes specifying which central ObjectServer is to be used for authentication and password management.

Note: The system PAM configuration file must be changed by the root user.

The following table lists the arguments that can be added to the ObjectServer PAM module entries in this file.

Table 81. ObjectServer PAM module arguments

Argument	Description
debug	If specified, debugging information is written to syslog.
log_to_stderr	If specified, information is logged to stderr instead of syslog.
no_warn	If specified, warning level information is not sent to the application requesting authentication or password management.
server= <i>string</i>	Specifies the central ObjectServer to use for authentication and password management. If not specified, the ObjectServer specified in the ObjectServer PAM configuration file is used.
try_first_pass	If specified, the ObjectServer PAM module does not prompt for a password. Instead it obtains a previously entered password when PAM modules are stacked. This argument can only be specified in auth and password entries.
use_first_pass	If specified, the ObjectServer PAM module does not prompt for a password. Instead it obtains a previously entered password when PAM modules are stacked. This argument can only be specified in auth entries.

Note: Arguments that are specified for the ObjectServer PAM module in the system PAM configuration file override the settings in the ObjectServer PAM configuration file.

Related reference

“Modifying your ObjectServer settings in the ObjectServer PAM configuration file” on page 350

Modifying your ObjectServer settings in the ObjectServer PAM configuration file:

The ObjectServer PAM configuration file is created when you install the ObjectServer PAM module by using the **nco_install_ospam** script.

The ObjectServer PAM configuration file is called \$NCHOME/omnibus/etc/pam_omnibus_os.conf. This file defines the central ObjectServer that is configured to handle PAM authentication.

The following table describes the settings that you can change in the pam_omnibus_os.conf file.

Table 82. ObjectServer PAM configuration file properties

Property	Description
Debug TRUE FALSE	If TRUE, debugging information is written to syslog. The default is FALSE.
LogToStderr TRUE FALSE	If TRUE, information is logged to stderr instead of syslog. The default is FALSE.
Server <i>string</i>	Specifies the ObjectServer to use for authentication. This is initially set to the ObjectServer that you specified when you ran the nco_install_ospam script to install the ObjectServer PAM module.

Note: Arguments that are specified for the ObjectServer PAM module in the system PAM configuration file override the settings in the pam_omnibus_os.conf file.

Related reference

“Modifying your ObjectServer settings in the system PAM configuration file” on page 349

Implementing authorization by using groups and roles

Permissions control access to objects and data in the ObjectServer. By combining one or more permissions into *roles*, you can manage access quickly and efficiently.

Each user is assigned to one or more *groups*. You can assign permissions to each group to perform actions on database objects by granting one or more roles to the group. You can create logical groupings such as super users or system administrators, physical groupings such as London or New York NOCs, or any other groupings to simplify your security setup.

For example, creating automations requires knowledge of Tivoli Netcool/OMNIBus operations and the way that a particular ObjectServer is configured. You do not typically want all of your users to create or modify automations. One solution is to create a role named AutoAdmin, with permissions to create and modify trigger groups, files, SQL and external procedures, and signals. You can then grant that role to a group of administrators who will be creating and updating automations.

The security.sql script contains default groups and roles for different classes of users, such as operators and administrators. You can also use this script as a template to create your own groups and roles.

Users, groups, and roles can be configured by using Netcool/OMNIBus Administrator or ObjectServer SQL. See the *IBM Tivoli Netcool/OMNIBus Administration Guide* for further information.

System and object permissions

Permissions determine what types of actions users can perform in the ObjectServer.

You assign permissions to roles by using the GRANT command. There are two types of permissions:

- *System permissions*, which control the commands that can be run in the ObjectServer
- *Object permissions*, which control access to individual objects, such as tables

System permissions include the ability to use the SQL interactive interface, create a database, and shut down the ObjectServer. For example:

- ISQL permission is required to connect to the ObjectServer by using the SQL interactive interface.
- ISQLWrite permission is required to modify ObjectServer data by using the SQL interactive interface.

Object permissions specify the actions that each role is authorized to perform on a particular object. Each object has a set of associated actions. For example, the actions that you can perform on an ObjectServer database are:

- DROP
- CREATE TABLE
- CREATE VIEW

Default Tivoli Netcool/OMNIBus roles

When you run the database initialization utility (**nco_dbinit**) to create an ObjectServer, a set of default roles is created.

The following table describes the default roles, which are defined in the `security.sql` script.

Table 83. Default roles

Role name	Description
CatalogUser	<p>This role includes permissions to view information about system, tools, security, and desktop database tables.</p> <p>This role provides a basis for Tivoli Netcool/OMNIBus permissions. This role does not provide sufficient permissions to use any Tivoli Netcool/OMNIBus applications.</p> <p>All groups should be assigned this role.</p>
AlertsUser	<p>This role includes the following permissions:</p> <ul style="list-style-type: none">• View, update, and delete entries in the alerts.status table• View, insert, and delete entries in the alerts.journal table• View and delete entries in the alerts.details table <p>This role, in combination with the CatalogUser role, enables you to display and manipulate alerts, create filters and views, and run standard tools in the event list.</p>

Table 83. Default roles (continued)

Role name	Description
AlertsProbe	<p>This role includes permissions to insert and update entries in the alerts.status table, and insert entries in the alerts.details table.</p> <p>This role, in combination with the CatalogUser role, provides the permissions that a probe needs to generate alerts in the ObjectServer. These permissions should be granted to any user that runs a probe application.</p>
AlertsGateway	<p>This role includes permissions to insert, update, and delete entries in the alerts.status table, alerts.details table, alerts.journal table, alerts.conversions table, alerts.col_visuals table, alerts.colors table, the desktop tools tables, and the tables in the transfer database. The transfer database is used internally by the bidirectional ObjectServer Gateway to synchronize security information between ObjectServers.</p> <p>This role also includes permissions to select, insert, update, and delete entries in the master.servergroups table, and permissions to raise the following signals: gw_counterpart_down, gw_counterpart_up, gw_resync_start, and gw_resync_finish.</p> <p>This role, in combination with the CatalogUser role, provides the permissions that a gateway needs to generate alerts in the ObjectServer. These permissions should be granted to any user that runs a gateway application.</p>
DatabaseAdmin	<p>This role includes permissions to create databases and files, and to create tables in the alerts, tools, and service databases. This role also includes permissions to modify or drop the alerts.status, alerts.details, and alerts.journal tables, and permissions to create and drop indexes in the alerts.status, alerts.details, and alerts.journal tables.</p> <p>This role, in combination with the CatalogUser role, provides permissions to create relational data structures in the ObjectServer.</p>
AutoAdmin	<p>This role includes permissions to create trigger groups, files, SQL procedures, external procedures, and user signals. This role also includes permissions to create, modify, and drop triggers in the default trigger groups, and to modify or drop default trigger groups.</p> <p>This role, in combination with the CatalogUser role, provides permissions to create automations in the ObjectServer.</p>
ToolsAdmin	<p>This role includes permissions to delete, insert, and update all tools tables.</p> <p>This role, in combination with the CatalogUser role, provides permissions to create and modify tools that can be run from the desktop and Netcool/OMNIBus Administrator .</p>

Table 83. Default roles (continued)

Role name	Description
DesktopAdmin	This role includes permissions to update all desktop catalogs to insert, update, and delete colors, visuals, menus, classes, resolutions, and conversions. This role, in combination with the CatalogUser role, provides permissions to customize the desktop.
SecurityAdmin	This role, in combination with the CatalogUser role, includes permissions to manipulate users, groups, and roles by using Netcool/OMNIbus Administrator or the SQL interactive interface. This role also includes permissions to set properties and drop user connections.
ISQL	This role, in combination with the CatalogUser role, includes permission to view ObjectServer data by using the SQL interactive interface.
ISQLWrite	This role, in combination with the CatalogUser role, includes permissions to view and modify ObjectServer data by using the SQL interactive interface.
SuperUser	This role has all available permissions. You cannot modify the SuperUser role.
Public	All users are assigned this role. By default, the Public role is not assigned any permissions. You can modify, but not drop, the Public role.
ChannelAdmin	This role includes permissions to set up channels for accelerated event notification.
ChannelUser	This role includes permissions to receive and act on notifications for accelerated events that are broadcast over channels.

Default Tivoli Netcool/OMNIbus groups

When you run the database initialization utility (**nco_dbinit**) to create an ObjectServer, a set of default groups is created.

The following table describes the default groups for Network Management operators and administrators, which are defined in the `security.sql` script.

Table 84. Default groups

Group name	Description
Probe	This group is assigned the CatalogUser, AlertsUser, and AlertsProbe roles.
Gateway	This group is assigned the CatalogUser, AlertsUser, and AlertsGateway roles.
ISQL	This group is assigned the ISQL role.
ISQLWrite	This group is assigned the ISQLWrite role.
Public	This group is assigned the Public role. All users are members of this group.
Normal	This group is assigned the CatalogUser, AlertsUser, and ChannelUser roles. This group cannot be deleted or renamed.

Table 84. Default groups (continued)

Group name	Description
Administrator	This group is assigned the CatalogUser, AlertsUser, ToolsAdmin, DesktopAdmin, ChannelUser, and ChannelAdmin roles. This group cannot be deleted or renamed.
System	This group is assigned the CatalogUser, AlertsUser, ToolsAdmin, DesktopAdmin, AlertsProbe, AlertsGateway, DatabaseAdmin, AutoAdmin, SecurityAdmin, ISQL, ISQLWrite, SuperUser, ChannelUser, and ChannelAdmin roles. This group cannot be deleted or renamed.

Using groups for row level security in the event list

The Normal, Administrator, and System groups provide group row level security in the event list. These groups cannot be deleted or renamed.

The **AlertSecurityModel** ObjectServer property determines whether group row level security is enforced in the event list. By default, the **AlertSecurityModel** is disabled. In this case:

- Members of the Normal group can modify a row that is assigned to themselves or the nobody user.
- Members of the Administrator group can modify a row that is assigned to themselves, the nobody user, or a member of the Normal group.

If the **AlertSecurityModel** property is enabled, only users in the group that owns the row can modify the row. In this case, a member of the Normal or Administrator group can modify a row that is assigned to a group of which they are a member.

A member of the System group can always modify any row.

Default Tivoli Netcool/OMNIBus users

When you run the database initialization utility (**nco_dbinit**) to create an ObjectServer, a set of default users is created.

The following table describes the default users, which are defined in the security.sql script.

Table 85. Default users

User name	Description
root	This user is created with an empty string as a password by default. You can reset the password by using Netcool/OMNIBus Administrator , or the ALTER USER ObjectServer SQL command.
nobody	This user is disabled and cannot be used to access the ObjectServer. Ownership of each alert in the alerts.status table is assigned to a user when the row is inserted. By default, probes assign rows to the nobody user.

The permissions that are assigned to the default users are shown in the following table.

Table 86. Default user permissions

Permission	User root	User nobody
Set enable or disable	No	No
Set full name	Yes	No
Set password	Yes	No
Set PAM	Yes	No
Assign or remove restriction filter	Yes	Yes Note, however, that you cannot log on as a nobody user.
Drop user	No	No
Grant or revoke permission	No	No
Be a member of a group	Yes	Yes

Using restriction filters to filter table information

A restriction filter provides a way to restrict the rows that are displayed when a user views table data. When the filter is assigned to a user or group, the filter controls the data that is displayed and modified from client applications, and modified in SQL commands.

Only rows that satisfy the criteria specified in the filter condition are returned.

Defining and following an audit trail

When you secure applications, it is important to monitor, or audit, the effectiveness of your security. One way to do this is to configure the system to log certain types of messages. Then you can monitor the logs to see if anything of interest or concern has occurred.

To configure user audit logs, use the ObjectServer properties or command-line options described in the following table.

Table 87. Auditing ObjectServer properties and command-line options

Property	Command-line option	Description
Sec.AuditLog <i>string</i>	<code>-secauditlog <i>string</i></code>	Specifies the file to which audit information is written. The default is <code>\$NCHOME/omnibus/log/servername/audit.log</code> .
Sec.AuditLevel <i>string</i>	<code>-secauditlevel <i>string</i></code>	Specifies the level of security auditing performed. Possible values are debug, info, warn, and error. The default is warn. The debug and info levels generate messages for authentication successes and failures, while warn and error generate messages for authentication failures only.

As part of your security process, check your logs frequently.

Property value encryption

You can use property value encryption to encrypt string values in a properties file or configuration file so that the strings cannot be read without a key. When the process that uses the properties file or configuration file starts up, the strings are decrypted.

You can use this encryption mechanism in the ObjectServer, proxy server, **nco_postmsg**, LDAP, probe, and gateway properties files. You can also use this mechanism to encrypt passwords that are stored in process agent configuration files.

The property value encryption mechanism uses the Advanced Encryption Standard (AES), which supports keys of 128, 192, and 256 bits, a command-line key generator (**nco_keygen**), and an encryption utility (**nco_aes_crypt**). Cryptographic algorithms are also available for use in FIPS 140-2 and non-FIPS 140-2 mode. The procedure is as follows:

1. Generate a key and store it in a key file by running the **nco_keygen** utility.
2. Set the value of the **ConfigKeyFile** property to the file path and file name of the key file that the **nco_keygen** utility generates. This step is not applicable if you are encrypting passwords in the process agent configuration file.
3. Encrypt a string value with the key by running the **nco_aes_crypt** utility.
4. Add the encrypted sting value to a properties file.

Related reference

“Switching your installation to FIPS 140-2 mode” on page 299

Generating a key in a key file

Run the **nco_keygen** utility to generate a key and store it in a key file. Command-line options are available for you to either specify a hexadecimal value for the key, or to specify a length in bits for automatic key generation.

You can create a single key that is used by all the properties files, or create a key for each properties file.

To generate a key within a key file:

Run **nco_keygen** as follows. Optional entries are shown in square brackets.

```
$NCHOME/omnibus/bin/nco_keygen -o key_file [-l length | -k key]
```

In this command:

- *key_file* represents the output file path and file name to which the key is saved.
- *length* represents the length in bits of the key, as specified by you. This number must be divisible by 8 to make a whole number of bytes. The default is 128. Only 128, 192, and 256 are valid key lengths for AES encryption.
- *key* represents the value of the key in hexadecimal digits, as specified by you. You can use either the -l or -k command-line option, but not both.

If you use the -o command-line option to specify an output file name, and omit both the -l and -k options, a randomly-generated 128-bit key is written to the file.

Results

The **nco_keygen** utility writes the key to the file, using the format *length:key*, where *length* is the number of bits in the key, represented as ASCII decimal numerals, and *key* is the key data.

The key can be used to both encrypt and decrypt data. For decryption, the key file must be accessible to the process decrypting the data. Access to the key file could be controlled by UNIX or Windows file permissions or other methods, though this is not covered by any Tivoli Netcool/OMNIBus schema or tools.

Specifying the key file as a property

In the properties file in which you want to specify an encrypted string value, set the value of the **ConfigKeyFile** property to the file path and file name of the key file that was generated by the **nco_keygen** utility.

You can set the **ConfigKeyFile** property in the following files:

- ObjectServer properties files: `$NCHOME/omnibus/etc/servername.props`
- Proxy server properties files: `$NCHOME/omnibus/etc/proxyserver.props`
- **nco_postmsg** properties file: `$NCHOME/omnibus/etc/nco_postmsg.props`
- LDAP properties file: `$NCHOME/omnibus/etc/ldap.props`
- Probe properties files: `$OMNIHOME/probes/arch/probename.props`
Where *arch* represents the operating system directory
- Gateway properties files: `$OMNIHOME/etc/gatewayname.props`

Tip: This property setting is required so that encrypted strings in the properties file can be decrypted by the key at run time.

When running the process agent daemon **nco_pad**, you can use the **-keyfile** command-line option to specify the file path and file name of the key file.

Encrypting a string value with the key

Use the **nco_aes_crypt** utility to encrypt a string value with the key that was generated by the **nco_keygen** utility.

To encrypt a string value:

Run **nco_aes_crypt** as follows:

```
$NCHOME/omnibus/bin/nco_aes_crypt -c cipher -k key_file string_value
```

In this command:

- *cipher* is the algorithm that is used to encrypt the string value. Specify one of the following values for *cipher*, based on your mode of operation:
 - FIPS 140–2 mode: Specify `AES_FIPS`.
 - Non-FIPS 140–2 mode: Specify either `AES_FIPS` or `AES`. Use `AES` (the default) only if you need to maintain compatibility with passwords that were encrypted using the tools provided in versions earlier than Tivoli Netcool/OMNIBus V7.2.1.
- *key_file* is the file path and name of the key file. This value must match that specified for the **ConfigKeyFile** property in the properties file.
- *string_value* is the string value that you want to encrypt.

Restriction: Due to startup order, the **MessageLevel** property cannot currently be encrypted.

Results

The output is displayed in the console window in encrypted form, and is delimited with @ symbols. You can now copy the output text, including the @ symbols, for use within the relevant properties file.

Tip: The **nco_aes_crypt** utility has additional command-line options that you can use to:

- Encrypt the contents of a file instead of a string value.
- Send encrypted output to a file instead of the console window.
- Manually decrypt encrypted values.

Related reference

“nco_aes_crypt command-line options”

Adding an encrypted value to a properties file

After encrypting a string value, add it to the properties file within which you want to hide the actual value.

To add an encrypted value to a properties file:

1. Open the properties file for editing.
2. Specify (or paste) the encrypted string value, including the @ delimiter symbols, as the setting for the relevant property. For example:
`Gate.ObjectServerA.Password : '@44:Kris2m3QLsy+dZYnt3/jptl8cd7c6Fmboaj+E6XrNw8=@'`
3. Set the value of the **ConfigCryptoAlg** property to the cryptographic algorithm to use when decrypting the string; for example, AES_FIPS. This value must match that specified when you ran **nco_aes_crypt** with the -c setting, to encrypt the string value.

The **ConfigCryptoAlg** property is used in conjunction with the **ConfigKeyFile** property to decrypt the string value when required.

nco_aes_crypt command-line options

You can use the **nco_aes_crypt** utility to encrypt and decrypt string values, or data held in a file.

The **nco_aes_crypt** utility is located in \$NCHOME/omnibus/bin, and requires a key file that can be generated by using the **nco_keygen** utility.

The syntax for the **nco_aes_crypt** utility is as follows, with optional values shown in square brackets:

```
nco_aes_crypt [-d] [-o string] [-c string] -k string -f filename
nco_aes_crypt [-d] [-o string] [-c string] -k string data
```

The command-line options are described in the following table.

Table 88. *nco_aes_crypt* command-line options

Command-line option (or value)	Description
-d	Sets the mode in which the utility runs to decrypt mode. The default is encrypt mode. This command-line option is not supported on Windows.
-o <i>string</i>	Specifies the output file to which the encrypted data is written.
-c <i>string</i>	Specifies the cryptographic algorithm to use for encrypting or decrypting. The values are: <ul style="list-style-type: none"> • AES_FIPS: Use this value in FIPS 140–2 mode. • AES: Use this value only to maintain compatibility with the AES property encryption that was available in Tivoli Netcool/OMNibus V7.2. This is applicable only in non-FIPS 140–2 mode.
-k <i>string</i>	Specifies the path and name of the file that stores the key that encrypts or decrypts data.
-f <i>string</i>	Specifies the path and name of a file that contains data to be encrypted or decrypted.
<i>data</i>	Specifies a string value to be encrypted or decrypted.

Related tasks

“Generating a key in a key file” on page 356

Related reference

“Switching your installation to FIPS 140-2 mode” on page 299

Chapter 14. Quick reference to setting up SSL

Use this information as a quick reference to setting up ObjectServer failover with SSL in non-FIPS mode.

The following table shows the actions to perform when setting up SSL.

Table 89. Quick reference for setting up SSL

Action	Instructions	More information
Set up the omni.dat file	Chapter 15, "Setting up the omni.dat file for SSL," on page 363	Chapter 22, "Using SSL for client and server communications," on page 377 "Generating the interfaces file on UNIX" on page 379: Manually editing the connections data file for SSL.
Generate the interfaces file	Chapter 16, "Generating the interfaces file," on page 365	"Starting iKeyman" on page 382
Generate the certificates	Chapter 17, "Generating the certificates," on page 367	"About the key database files" on page 381 "Managing certificates from the command line" on page 395 "Guidelines for certificate management" on page 382
Add the signer certificate to the client from the server	Chapter 18, "Adding the signer certificate to the client from the server," on page 369	
Configure the clients with the available CommonNames property	Chapter 19, "Configuring the clients with the available CommonNames property," on page 371	"Distributed installations and SSL" on page 379

Chapter 15. Setting up the omni.dat file for SSL

Make the following changes to the omni.dat file:

```
[PSERV]
{
    Primary:    myhost_name.ibm.com    ssl 7100
}
[BSERV]
{
    Primary:    myhost_name.ibm.com    ssl 8100
}
[NCOMS]
{
    Primary:    myhost_name.ibm.com    ssl 7100
    Backup:     myhost_name.ibm.com    ssl 8100
}
```

Chapter 16. Generating the interfaces file

Generate the interfaces file by running the following command:
`$NCHOME/bin/nco_igen`

Chapter 17. Generating the certificates

Certificates generated by the **nc_ikeyman** utility do not include the **Basic Constraints** field which is required for Java based clients.

1. Create the CMS key database by running the following command:

```
$NCHOME/bin/nc_gskcmd -keydb -create -db "$NCHOME/etc/security/keys/omni.kdb"
-pw password -stash -expire 366
```

2. Create a self signed CA certificate by running the following command:

```
$NCHOME/bin/nc_gskcmd -cert -create -db "$NCHOME/etc/security/keys/omni.kdb"
-pw password -label "CA" -size 1024 -ca true -dn "CN=CA ,
O=IBM, OU=Support, L=SouthBank,
ST=London, C=GB" -expire 366 -x509version 3
```

Where:

- *CA* specifies the common name of the certificate owner. This is the name of the server to which your clients are connecting; for example, *NCOMS*.
- *IBM* specifies your company name.
- *Support* specifies the organizational unit or department name within which the certificate will be used.
- *SouthBank* specifies the locality or city of your organization.
- *London* specifies your state or province.
- *GB* specifies the two-letter ISO code for your country.

For more information about generating a self-signed certificate, see “Creating a self-signed certificate” on page 385.

3. Create a certificate request for the primary ObjectServer by running the following command:

```
$NCHOME/bin/nc_gskcmd -certreq -create -db "$NCHOME/etc/security/keys/omni.kdb"
-pw password -label "PSERV" -size 1024 -dn "CN=<NCOMS> ,
O=<IBM>, OU=<Support>, L=<SouthBank>,
ST=<London>, C=<GB>" -file "$NCHOME/etc/security/keys/pservreq.arm"
```

Note: The label is identical to the servername in the *omni.dat* file but the common name (CN) is set to the virtual ObjectServer for both certificate requests.

For more information about generating a self-signed certificate, see “Requesting a server certificate from a CA” on page 389.

4. Create a certificate request for the backup ObjectServer by running the following command:

```
$NCHOME/bin/nc_gskcmd -certreq -create -db "$NCHOME/etc/security/keys/omni.kdb"
-pw password -label "BSERV" -size 1024 -dn "CN=<NCOMS> ,
O=<IBM>, OU=<Support>, L=<SouthBank>,
ST=<London>, C=<GB>" -file "$NCHOME/etc/security/keys/bservreq.arm"
```

Note: The label is identical to the servername in the *omni.dat* file but the common name (CN) is set to the virtual ObjectServer for both certificate requests.

5. Sign the certificate requests using the signer certificate label CA created in step 2, by running the following command:

For the primary ObjectServer:

```
$NCHOME/bin/nc_gskcmd -cert -sign -db "$NCHOME/etc/security/keys/omni.kdb"
-pw password -label "CA" -target "$NCHOME/etc/security/keys/pservcert.arm"
-expire 10 -file "$NCHOME/etc/security/keys/pservreq.arm"
```

For the backup ObjectServer:

```
$NCHOME/bin/nc_gskcmd -cert -sign -db "$NCHOME/etc/security/keys/omni.kdb"  
-pw password -label "CA" -target "$NCHOME/etc/security/keys/bservcert.arm"  
-expire 10 -file "$NCHOME/etc/security/keys/bservreq.arm"
```

For more information about generating a self-signed certificate, see “Signing a certificate request file with a signer certificate” on page 390.

6. Receive the signed certificate in the omni.kdb file for the ObjectServer, by running the following command:

For the primary ObjectServer:

```
$NCHOME/bin/nc_gskcmd -cert -receive -db "$NCHOME/etc/security/keys/omni.kdb"  
-pw password -file "$NCHOME/etc/security/keys/pservcert.arm"
```

For the backup ObjectServer:

```
$NCHOME/bin/nc_gskcmd -cert -receive -db "$NCHOME/etc/security/keys/omni.kdb"  
-pw password -file "$NCHOME/etc/security/keys/bservcert.arm"
```

For more information about generating a self-signed certificate, see “Receiving server certificates from CAs” on page 391.

7. Extract the signer certificate by running the following command:

```
$NCHOME/bin/nc_gskcmd -cert -extract -db "$NCHOME/etc/security/keys/omni.kdb"  
-pw password -label "CA" -target "$NCHOME/etc/security/keys/cacert.arm"
```

8. Distribute this file to the clients that are connecting to the ObjectServer using SSL.

Chapter 18. Adding the signer certificate to the client from the server

Add the signer certificate to the client from the server by running the following command:

```
$NCHOME/bin/nc_gskcmd -cert -add -db "$NCHOME/etc/security/keys/omni.kdb"  
-pw password -label "CA" -file "cacert.arm"
```

Chapter 19. Configuring the clients with the available CommonNames property

Use the following information to specify acceptable common names for the unidirectional and bidirectional gateways:

Chapter 20. Gateways

In a failover pair, clients identify both ObjectServers by using the same server name. This name must be the common name of the server when using the SSL port to connect.

For the unidirectional gateway, you can use the **Gate.Reader.CommonNames** and **Gate.Writer.CommonNames** properties to specify acceptable common names for the primary and backup ObjectServers.

For the bidirectional gateway, you can use the **Gate.ObjectServerA.CommonNames** and **Gate.ObjectServerB.CommonNames** properties.

The following is an example configuration for a unidirectional gateway:

```
Gate.Reader.Server:          'PSERV'  
Gate.Reader.CommonNames:    'NCOMS'  
Gate.Writer.Server:         'BSERV'  
Gate.Writer.CommonNames:    'NCOMS'
```

Note: You cannot connect directly to PSERV or BSERV using the example configuration shown above. This can only be done by using NCOMS, that is by using a virtual ObjectServer.

Chapter 21. Probes

If the probe is connecting to an ObjectServer using SSL, and the Common Name field of the received certificate does not match the name specified by the server property, use the following property to specify a comma-separated list of acceptable SSL common names (the default is to use the server property).

SSLServerCommonName: 'NCOMS'

Chapter 22. Using SSL for client and server communications

Tivoli Netcool/OMNIBus supports the use of the Secure Sockets Layer (SSL) protocol for communication between its servers and clients.

SSL uses digital certificates for key exchange and authentication. When a client initiates an SSL connection, the server presents the client with a certificate that is signed by a Certificate Authority (CA); that is, a trusted party that guarantees the identity of the certificate and its creator. The server certificate contains the identity of the server, the public key, and the digital signature of the certificate issuer.

By reading the server certificate, the client can determine if the server is a trusted source, and then accept or reject the connection. To verify the signature on the server certificate, the client requires the public key of the issuing CA. Because public keys are distributed in certificates, this means that the client must have a certificate for the issuing CA. This certificate must be signed by the CA.

Server certificates can be generated for ObjectServers, process agents, and proxy servers.

Certificates serve two specific purposes:

- They provide authenticated proof to a client that the server they connect to is owned by the company or individual that has installed the certificate.
- They contain the public key that the client uses to establish an encrypted connection to the server.

When in FIPS 140-2 mode, all encryption and key generation functions that are required for the secured SSL connections are provided by FIPS 140-2 approved cryptographic providers.

To configure SSL communication, you must perform the following tasks:

1. Create entries for SSL connections in the Server Editor.
2. If you are running in FIPS 140-2 mode, configure cryptographic properties for encryption and key generation.
3. Create and distribute certificates and keys to servers and clients by using the IBM Key Management (iKeyman) utility, which is used for certificate management.

Related concepts

Chapter 13, “Tivoli Netcool/OMNIBus user access security,” on page 337

Configuring the Server Editor for SSL connections

A set of guidelines is provided for configuring the Server Editor to use encrypted (SSL) connections, unencrypted (non-SSL) connections, or both.

Using the Server Editor to configure SSL on UNIX

In the UNIX Server Editor, you can enable encrypted connections, unencrypted connections, or both.

On the server host computer:

- You can set the unencrypted port and leave the SSL port unset (or set it to 0); in this case, only unencrypted connections are allowed.
- You can set the SSL port and unset the unencrypted port (or set it to 0); in this case, only encrypted connections are allowed.
- You can set both an unencrypted port and an SSL port; in this case both encrypted and unencrypted connections are allowed. Firewalls can be configured to allow access to the appropriate ports from other systems.

Note: If the server allows both encrypted and unencrypted connections, clients that use the same interfaces file as the server (including local clients) connect using the unencrypted port. If you want to use SSL to connect on these computers, do not specify an unencrypted port for the server.

On each client computer:

- If you want the client to connect to the server from this computer without using encryption, create an entry that specifies the server host, server name, and unencrypted port.
- If you want the client to connect to the server from this computer by using encryption, create an entry that specifies the server host, server name, and SSL port. For this entry, the server name that you specify *must* be identical to the common name that is specified for the server when creating and authorizing a certificate request.

Note: If you create entries for both an SSL connection and an unencrypted connection on the same client computer for the same server, you must use the common name for the SSL entry (as specified when creating a certificate request), and an alternative name for the unencrypted entry.

Related tasks

“Configuring server communication information” on page 242

Using the Server Editor to configure SSL on Windows

In the Windows Server Editor, you can enable encrypted connections, unencrypted connections, or both.

On the server host computer:

- If you want the server to allow encrypted connections from clients, create a Listener entry by selecting the **Listener** check box, and also select the **SSL** check box.
- If you want the server to allow unencrypted connections from clients, create a Listener entry by selecting the **Listener** check box, and clear the **SSL** check box.

On each client computer:

- If you want the client to connect to the server from this computer without using encryption, clear the **SSL** check box.
- If you want the client to connect to the server from this computer by using encryption, select the **SSL** check box.

Note: If you create entries for both an SSL connection and an unencrypted connection on the same client computer for the same server, you must use the common name for the SSL entry (as specified when creating a certificate request), and an alternative name for the unencrypted entry.

Related tasks

“Configuring server communication information” on page 242

Generating the interfaces file on UNIX

You can edit the connections data file `omni.dat` directly, and run **nco_igen** to generate an interfaces file that can then be distributed to other systems. However, if you are using SSL connections, the procedures for editing the `omni.dat` file and running **nco_igen** must be modified.

Manually editing the connections data file for SSL

The following example shows an `omni.dat` file entry with both an unencrypted port and an SSL port defined for the NCOMS ObjectServer:

```
[NCOMS]
{
  Primary: nocturama 3000 ssl 3500
}
```

When you run **nco_igen** for this entry, it generates two server (master) entries, one with an unencrypted port of 3000 and one with an SSL port of 3500. Two client (query) entries are also created. However, for a server that allows both encrypted and unencrypted connections, clients that use the same interfaces file as the server (local clients) connect using the unencrypted port. If you want to use SSL to connect locally, do not specify an unencrypted port for the server.

Related reference

“Distributed installations and SSL”

Distributed installations and SSL

If you are using SSL and unencrypted ports on your server host computer, you must create an interfaces file, which uses SSL ports, for your remote client computers. Distribute this interfaces file to remote client computers, rather than using the interfaces file that is generated on the server host computer.

If using an SSL port, the server name that you specify *must* be identical to the common name that is specified for the server when creating a certificate request.

Note: In a failover pair, clients identify both ObjectServers by using the same server name. This name must be the common name of the server when using the SSL port to connect. For the unidirectional gateway, you can use the **Gate.Reader.CommonNames** and **Gate.Writer.CommonNames** properties to specify acceptable common names for the primary and backup ObjectServers. For the bidirectional gateway, you can use the **Gate.ObjectServerA.CommonNames** and **Gate.ObjectServerB.CommonNames** properties. For further information about using these ObjectServer Gateway properties, see the *IBM Tivoli Netcool/OMNIBus ObjectServer Gateway Reference Guide*, SC14-7609.

Related reference

“Generating the interfaces file on UNIX”

Managing digital certificates for SSL communication

The IBM Key Management (iKeyman) utility is a graphical tool that you can use to manage keys and digital certificates that are required for SSL communication. You can also manage keys and digital certificates from a command-line interface.

iKeyman uses a Certificate Management System (CMS) key database to store digital certificates and keys. The key database requires a password in order to protect private keys, which are used to sign documents and to decrypt messages that are encrypted with public keys.

In a key database, digital certificates from CAs are stored as *signer* certificates. Any self-signed certificates that are created, or any server certificates that are received from issuing CAs in response to a certificate request, are stored as *personal* certificates.

Notes for SSL connections in FIPS 140–2 mode

To use the **nc_gskcmd** command-line utility for setting up SSL communications, set up your operating system correctly.

Before you begin

The utility is provided by IBM Global Security Kit (GSKit). The path to the GSKit libraries must appear in the environment variable applicable to your operating system, as shown in the following table:

Table 90. Location of GSKit libraries and corresponding environment variables

Operating system	Location of libraries	Environment variable
UNIX	\$NCHOME/platform/arch/lib where <i>arch</i> is your operating system directory	LD_LIBRARY_PATH
Windows	%NCHOME%\platform\win32\lib	PATH

The **gsk8capicmd** utility is located at :

- **UNIX** **Linux** \$NCHOME/platform/arch/bin where *arch* is your operating system directory
- **Windows** %NCHOME%\platform\win32\bin

To set up your operating system for the use of the **gsk8capicmd** utility:

Change to the \$NCHOME/bin location and create a symbolic link to the **nc_gskcmd** utility by running the following command:

```
ln -s nco_run nc_gskcmd
```

Results

You can now use the **nc_gskcmd** command-line utility to perform the tasks required to set up SSL communication.

Related concepts

Chapter 10, “Configuring FIPS 140–2 support for the server components,” on page 295

Related tasks

“Checking the shared library paths” on page 99

Related reference

“Setting Tivoli Netcool/OMNIbus environment variables (UNIX and Linux)” on page 95

About the key database files

A CMS key database consists of a number of files that are named with the same file name stem, but with differing extensions.

These files are described in the following table.

Table 91. Key database files

File extension	Description	Tivoli Netcool/OMNIbus file name
.kdb	A key database file is used to store key pairs and digital certificates.	omni.kdb
.rdb	When a certificate request is created, a .rdb file is created to store the requested key pair and the certificate request data. When a signed certificate is obtained from a CA and received into the key database, the signed certificate is matched up with the private key in the .rdb file and together they are added to the .kdb file as a certificate and its private key information. The request entry is then deleted from the request key database.	omni.rdb
.crl	A .crl file is created for legacy reasons and is no longer used. This file was used to store a certificate revocation list (CRL) detailing revoked or suspended certificates.	omni.crl
.sth	<p>You can save the password for a key database to a stash file if you require automatic login to the key database in order to gain access to the digital certificates. The password is stored in encrypted format in the stash file.</p> <p>Whenever the key database is accessed, the system checks whether a stash file exists. If found, the file contents are decrypted and used as input for the password.</p> <p>Note: Tivoli Netcool/OMNIbus requires a stash file.</p>	omni.sth

The key database files are stored in the following location:

- UNIX: \$NCHOME/etc/security/keys
- Windows: %NCHOME%\ini\security\keys

Guidelines for certificate management

To set up SSL connections between your clients and servers, you require a trusted signer certificate and a server certificate that has been signed by the trusted signer.

Use the following guidelines to set up a trusted network with SSL certificates:

1. On each computer where a server component (ObjectServer, process agent, or proxy server) is installed, create a key database for storing digital certificates. Also create a key database on each computer from which clients connect to the server by using an SSL port. Use a dedicated key database file (`omni.kdb`) for each Tivoli Netcool/OMNIbus installation on a server or client computer.
2. On each server computer, establish whether all the required root certificates are included in the set of default signer certificates that are available in the key database. If not, obtain any required self-signed root certificate (and any other intermediate certificates in the chain of trust) from a trusted CA. On receipt of these signer certificates, distribute each root (and intermediate) certificate to the server and client computers that require SSL connections.
3. If you want to set up a private trust network within which you are acting as the issuing CA for your server certificates, create a self-signed certificate on each server computer.
4. To use the self-signed certificate as a signer certificate, distribute the self-signed certificate to all clients by *extracting* the certificate from the server key database, and then *adding* the extracted certificate to the key database on each client computer.
5. From each server computer, create a request for a digital certificate for the server, and send the certificate request to a trusted CA for authorization. The CA authorizes the certificate request and uses its self-signed root certificate to generate a server certificate. The CA then returns the signed server certificate. If you are acting as the issuing CA within a private trust network, and you want to use a self-signed certificate to generate a server certificate, you must sign the certificate request and then return it as a signed server certificate.
6. On receipt of the server certificate, *receive* the certificate into the key database for the server. The server certificate is used to authenticate the server side of Tivoli Netcool/OMNIbus communications when a client requests a secure connection.
7. On each server computer, configure your system to use the new server certificate as the default certificate. This task is necessary only if you have more than one personal certificate in your key database.

Starting iKeyman

You can perform most of your certificate management tasks from the iKeyman GUI.

To start iKeyman:

Perform the relevant action for your operating system, as shown:

Operating system	Action
UNIX	From the command line, enter the following command: <code>\$NCHOME/bin/nc_ikeyman</code>

Operating system	Action
Windows	<p>Perform either of the following actions:</p> <ul style="list-style-type: none"> From the command line, enter the following command: %NCHOME%\bin\nc_ikeyman.bat Tip: Use this as your preferred option for starting iKeyman to ensure that the default key database location is always set to %NCHOME%\ini\security\keys\ within the iKeyman GUI. From Windows Explorer, navigate to the location %NCHOME%\bin and double-click the file nc_ikeyman.vbs.

The IBM Key Management window opens.

Creating a key database

A key database stores keys and digital certificates, and enables secure network connections between clients and servers.

When you create a key database, it is automatically populated with a number of signer certificates from common public CAs.

Important: To create a key database in FIPS 140-2 mode, use the **nc_gskcmd** utility. Do not use the iKeyman GUI.

To create a key database:

- To create a key database in FIPS 140-2 mode, use the **nc_gskcmd** utility. For more information, see “Managing certificates from the command line” on page 395, and also see the *IBM Global Security Kit Secure Sockets Layer Introduction and iKeyman User’s Guide*, SC32-1700. The following criteria apply to key database passwords in FIPS 140-2 mode:
 - The minimum password length is 14 characters.
 - A password must have at least one lower case character, one upper case character, and one digit or special character.
 - Each character must not occur more than three times in a password.
 - No more than two consecutive characters of the password can be identical
- To create a key database by using iKeyman:
 - Start iKeyman.
 - From the IBM Key Management window, click **Key Database File > New**.
 - Complete this window as follows:

Key database type

Select CMS from this list.

File Name

Type `omni.kdb` as the key database file name. The default is `key.kdb`.

Location

This location specifies the directory where the key database is stored, and typically defaults to either of the following directories:

- UNIX: `$NCHOME/etc/security/keys/`
- Windows: `%NCHOME%\ini\security\keys\`

Note: On Windows, if you started iKeyman from the command line by entering `%NCHOME%\bin\nc_ikeyman.bat`, the location shown above

is the default. If you started iKeyman from Windows Explorer by double-clicking the file %NCHOME%\bin\nc_ikeyman.vbs, the default location is given as %NCHOME%\bin\.

If UTF-8 encoding is enabled on Windows, the key database path must contain only characters that are supported by the default system code page.

Accept the default directory on UNIX. On Windows, ensure that the location is %NCHOME%\ini\security\keys\.

OK Click this button to accept the settings for the key database file.

The Password Prompt window opens, so that you can specify a password for controlling access to the key database.

4. Complete this window as follows:

Password

Type a password. As you type the characters, an indication of the password strength is given.

Note: Passwords are case sensitive, so whenever you are required to specify this password to open the key database, you *must* use the correct case to avoid errors.

Confirm Password

Retype the password.

Set expiration time

Select this check box to specify a period after which the password will expire. Enter the period in days. The default is 60 days. If this check box is clear, the password never expires.

Stash the password to a file?

Select this check box to save the password in an encrypted format to a stash file. This is a mandatory requirement for Tivoli Netcool/OMNIBus.

OK Click this button to close the window and create the key database.

The key database file is created in the specified directory, with the name omni.kdb, and additional files named omni.crl and omni.rdb. The stash file is also saved in the same location, with the name omni.sth.

Tip: Consider setting appropriate user permissions on the stash file to prevent unauthorized access.

Results

The IBM Key Management window now shows the file location and name of the key database, and the default signer certificates.

Important: As a security measure to prevent misuse of the default signer certificates, delete all of the default signer certificates from the key database.

What to do next

If you require additional signer certificates, you can request and add them to the key database. You can also view the contents of certificates and delete certificates.

Related concepts

"Setting your locale" on page 403

Related tasks

"Starting iKeyman" on page 382

"Requesting a server certificate from a CA" on page 389

"Changing the key database password" on page 394

"Viewing certificate details" on page 393

"Deleting certificates" on page 393

Creating a self-signed certificate

If you require a self-signed certificate, you can create one in the key database of a server computer.

When you create a self-signed certificate, you must specify information about your organization, which is used to generate the associated public-private key pair. The public key is incorporated into the certificate, and is used to check the validity of other certificates that are issued by the CA. The private key is used to sign the certificate, and is stored locally and securely within the key database.

To create a self-signed certificate:

From the command line, enter the following command:

```
$NCHOME/bin/nc_gskcmd -fips [true|false] -cert -create -db filename -pw  
password -label key_label -size key_size -ca Boolean -dn distinguished_name  
-expire integer1 -x509version integer2
```

In this command:

- *-fips* is an optional command-line option to sign the certificate with FIPS 140-2 certified cryptography. Use the argument *true* for FIPS 140-2 mode.
- *filename* specifies the name and path of the key database in which you want to store the certificate. Specify this value as a quoted string; for example, "*\$NCHOME/etc/security/keys/omni.kdb*" (on UNIX) or "*%NCHOME%\ini\security\keys\omni.kdb*" (on Windows).
- *password* specifies the password for accessing the key database.
- *key_label* specifies a meaningful short description that can be used to identify the self-signed certificate in the key database. Specify this value as a quoted string.

Tip: To help identify the certificate as self signed within the iKeyman GUI, you can append the words Certification Authority or CA to the label text.

- *key_size* specifies the length of the key (in bits). The values are 512, 1024, and 2048. The longer the key, the more secure the encryption. A longer key can, however, result in slower performance.
- *Boolean* is either *true* or *false*. Specify *true* to add the Basic Constraints extension to the self-signed certificate. The extension is added with a *CA:true* and *PathLen:max int*. This enables you to use the self-signed certificate to sign other certificates.

Note: Do *not* create the self-signed certificate with *-ca* set to *false*.

- *distinguished_name* specifies the distinguished name of the certificate owner as a quoted string in the following format:

```
"CN=string1, O=string2, OU=string3, L=string4, ST=string5, C=string6"
```

Where:

- *string1* specifies the common name of the certificate owner. This is the name of the server to which your clients are connecting; for example, NCOMS.

Note: The common name for the server must be the same as the server name in the connections data file (\$NCHOME/etc/omni.dat or %NCHOME%\ini\sql.ini) on the client machines.

- *string2* specifies your company name.
- *string3* specifies the organizational unit or department name within which the certificate will be used.
- *string4* specifies the locality or city of your organization.
- *string5* specifies your state or province.
- *string6* specifies the two-letter ISO code for your country.

Note: The common name (CN) setting of the -dn command-line option is mandatory, but all the other settings are optional for self-signed certificates.

- *integer1* specifies an expiry period for the certificate in days. Specify any value that ranges from 366 days to 7300 days (that is, 20 years).
- *integer2* specifies the version of X.509 certificate to create. The values are 1, 2, and 3. The default is 3.

Results

If you start the iKeyman GUI and open the omni.kdb database file, the newly-created certificate can be seen in the IBM Key Management window, as one of your entries in the **Personal Certificates** list. The key label is used to identify the certificate.

What to do next

You must now distribute this certificate as a signer certificate to all the clients that need to connect to the server by using SSL. To distribute the self-signed certificate, you must extract the certificate as a file to a specified network location, and then add the extracted file to the key database on each client computer.

Related tasks

“Extracting certificates from a key database”

“Adding certificates from CAs” on page 387

Related reference

“Managing certificates from the command line” on page 395

Extracting certificates from a key database

You can extract a copy of a signer or personal certificate from one key database and then add it to another key database as a signer certificate. When you extract a certificate, the public key is also extracted.

You can use this task to copy a self-signed certificate from a server computer to a network location.

To extract the certificate :

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database (omni.kdb) from which you want to extract the certificate. Then click **OK**.

4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, perform the relevant action as follows:
 - To extract a personal certificate (such as a self-signed certificate), select Personal Certificates from the drop-down list. Select the certificate that you want to extract, and then click **Extract Certificate**.
 - To extract a signer certificate, select Signer Certificates from the drop-down list. Select the certificate that you want to extract, and then click **Extract**.

The "Extract Certificate to a File" window opens.

6. Complete the window as follows:

Data type

Select a data type that matches that of the certificate.

Certificate file name

Specify the file name to which you want to extract the certificate. You can save the file as a .arm file.

Location

Specify the location where you want to save the extracted certificate file.

OK

Click this button to save the certificate to the specified file, and return to the IBM Key Management window.

What to do next

You must now open each key database into which you want to add the extracted certificate, and then add the certificate as a signer certificate.

Related tasks

"Starting iKeyman" on page 382

"Adding certificates from CAs"

Adding certificates from CAs

When you receive a root or associated intermediate certificate from an issuing CA, you must add the certificate to the key database on all client and server computers that require an SSL connection. Similarly, if you want to distribute a self-signed certificate that you have extracted from a server key database, you must add the extracted certificate file to all client computers.

Tip: If you have obtained a required certificate (or certificate details) from a CA, you must first save the information as a .arm text file or another acceptable format such as .cer, to a temporary location. (Your extracted self-signed certificate is already in .arm format.)

To add a certificate to a key database:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database (omni.kdb) to which you want to add the certificate. Then click **OK**.

4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, select Signer Certificates from the drop-down list, and then click **Add**.
6. Specify the following information:

Certificate file name

Specify the file name of the self-signed (or other) certificate file that you want to add to this database.

Location

Specify the location where you saved the file.

Tip: You can also use the **Browse** button to select the file and its location.

OK Click this button to accept these details.

The "Enter a Label" window opens.

7. Type a meaningful label for the certificate and click **OK** to save the file to the key database.

Results

The certificate is listed in the IBM Key Management window, as one of your entries in the **Signer Certificates** list. The label that you entered is used to identify the certificate.

What to do next

After adding the certificate to the key database, verify that the Basic Constraints extension has been set for the certificate. A Basic Constraints extension is required for all CA certificates that are used to sign server certificates. To check for the Basic Constraints extension:

1. From the iKeyman GUI, select the relevant certificate from the list of signer certificates and click **View/Edit**.
2. From the Key Information window, click **View Details**.
3. From the resulting window, look in the **Field** list for a node titled **Basic Constraints**, and then click the **Value** item under this node. In the **Value** area below the **Field** list, the following entry should be shown: **CA:true**.

Related tasks

"Starting iKeyman" on page 382

"Extracting certificates from a key database" on page 386

Requesting a server certificate from a CA

To obtain a certificate for a server, you must generate a certificate request in the key database and then submit the request to the relevant CA.

When you create a certificate request, you are prompted for information about your organization in order to generate a public-private key pair. The public key is incorporated into the certificate request to the CA, and the private key for the server is stored locally within the key database.

To create a certificate request from a server computer:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database (`omni.kdb`) in which you want to create the request. Then click **OK**.
4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, select **Personal Certificate Requests** from the drop-down list, and then click **New**. The "Create New Key and Certificate Request" window opens.
6. Complete this window as follows. Optional entries are indicated in the window.

Key Label

Specify the server name as the label. This is the name of the server to which your clients are connecting, and must be the same as the server name in the connections data file (`$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini`) on the server machine; for example, `NCOMS`.

Key Size

Select the length of the key (in bits) from this list. The default is 1024.

The longer the key, the more secure the encryption. A longer key can, however, result in slower performance.

Common Name

Specify the common name of the certificate owner. This name must match the server name that is specified in the connections data file (`$NCHOME/etc/omni.dat` or `%NCHOME%\ini\sql.ini`), or the properties file on the client computers.

Tip: Some clients in a virtual setup provide a property that enables you to specify a list of acceptable SSL common names; for example, the `SSLServerCommonName` probe property.

Organization

Specify your company name.

Organization Unit

Specify the organizational unit or department name within which the certificate will be used.

Locality

Specify the locality or city of your organization.

State/Province

Specify your state or province.

Zipcode

Specify your postal code.

Country or region

Select the two-letter ISO code for your country.

Enter the name of a file in which to store the certificate request

Specify a file name and location to which the request details should be saved. The default is:

- UNIX: \$NCHOME/etc/security/keys/certreq.arm
- Windows: %NCHOME%\ini\security\keys\certreq.arm

OK Click this button to create the request and close the window.

Results

The newly-created certificate request is listed in the IBM Key Management window, as an entry in the **Personal Certificate Requests** list. The key label is used to identify the request.

What to do next

You must now send the .arm file to the CA to request a digital certificate for the server. (This CA can be a public trusted CA, or the issuing CA within your private trust network.) After verifying your identity, the CA will send you a signed certificate, which is encrypted with their private key. You must then receive the signed certificate into the key database on the server.

Related tasks

"Starting iKeyman" on page 382

"Signing a certificate request file with a signer certificate"

"Receiving server certificates from CAs" on page 391

Signing a certificate request file with a signer certificate

This task is applicable only if you are operating within a private trust network, and are acting as the issuing CA for your server certificates. After a certificate request file is created, you must use your self-signed certificate to sign the request.

To sign a certificate request file with a self-signed certificate:

From the command line, enter the following command:

```
$NCHOME/bin/nc_gskcmd -fips [true|false] -cert -sign -db filename -pw
password -label key_label -target server_filename -expire integer -file
request_filename
```

In this command:

- *-fips* is an optional command-line option to sign the certificate with FIPS 140-2 certified cryptography. Use the argument *true* for FIPS 140-2 mode.
- *filename* specifies the name and path of the key database in which the self-signed certificate is stored. Specify this value as a quoted string; for example, "\$NCHOME/etc/security/keys/omni.kdb" (on UNIX) or "%NCHOME%\ini\security\keys\omni.kdb" (on Windows).
- *password* specifies the password for accessing the key database.
- *key_label* specifies the label of the self-signed certificate in the key database. Specify this value as a quoted string.

- *server_filename* specifies the name and path of the server certificate that you want to generate. Specify this value as a quoted string. You can specify the name as a .arm file.
- *integer* specifies an expiry period for the server certificate in days. Specify a value that ranges from 366 days to 7300 days (that is, 20 years). Note, however, that the expiry period for the server certificate must be less than the expiry period of the self-signed certificate.
- *request_filename* specifies the name and path of the certificate request file. Specify this value as a quoted string; for example, "\$NCHOME/etc/security/keys/certreq.arm" (on UNIX) or "%NCHOME%\ini\security\keys\certreq.arm" (on Windows).

Results

The server certificate file is created in the specified location.

What to do next

You must now start the iKeyman GUI, and receive the server certificate into the key database.

Related tasks

"Receiving server certificates from CAs"

Related reference

"Managing certificates from the command line" on page 395

Receiving server certificates from CAs

When the issuing CA sends you a server certificate in response to your certificate request, you must *receive* the server certificate into the key database in which you created the request.

Tip: If the CA sends additional signing certificates or intermediate CA certificates, you must first *add* these additional certificates to the key database *before* receiving the server certificate.

To receive a server certificate into the key database:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database to which you want to add the server certificate. Then click **OK**.
4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, select Personal Certificates from the drop-down list, and then click **Receive**.
6. Specify the following information:

Certificate file name

Specify the name of the certificate file, typically in .arm format, or another acceptable format such as .cer.

Location

Specify the location where you saved the file.

Tip: You can also use the **Browse** button to select the file and its location.

OK Click this button to accept these details and save the file to the key database.

Results

The signed certificate from the CA is merged with its request, and is added to the key database as a server certificate with its private key information. The request entry is then deleted from the key database. In the IBM Key Management window, the server certificate is shown in the **Personal Certificates** list, with the label that was assigned to the request.

What to do next

If not already set as the default certificate (as indicated by an asterisk to the left of the label), you must now set this certificate as the default in the key database of the server.

Related tasks

“Starting iKeyman” on page 382

Specifying the default certificate

You can specify a default certificate if more than one personal certificate is stored in the key database. For example, after receiving a server certificate from a CA, you can begin to use the certificate by making it the default.

To specify the default certificate:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database that contains the certificate you want to set as the default. Then click **OK**.
4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, select Personal Certificates from the drop-down list.
6. Select the certificate that you want to set as the default, and then click **View/Edit**. The Key Information window opens. This window provides a summary of the certificate details.
7. Select the **Set the certificate as the default** check box and click **OK**.

Results

In the IBM Key Management window, the label is annotated with an asterisk (*) .

Clients connecting to the Tivoli Netcool/OMNIbus server will be presented with this certificate and can use the public key in the certificate to encrypt the data that they send to the server.

Related tasks

“Starting iKeyman” on page 382

Viewing certificate details

You can examine the contents of any signer or personal certificate that is stored in the key database. While doing this, you can choose to set the certificate as a trusted root certificate, or as the default certificate.

To view the details of a certificate:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database (`omni.kdb`) that contains the certificate to be viewed. Then click **OK**.
4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. From the **Key database content** area, perform the relevant action as follows:
 - To view a signer certificate, select Signer Certificates from the drop-down list.
 - To view a personal certificate, select Personal Certificates from the drop-down list.
6. Select the certificate that you want to view, and then click **View/Edit**. The Key Information window opens. This window provides a summary of the certificate details.
7. If you are viewing a signer certificate, you can make the certificate a trusted root certificate by selecting the **Set the certificate as a trusted root** check box. If you are viewing a personal certificate that is not the default certificate, you can make the certificate the default by selecting the **Set the certificate as the default** check box.
8. To view the full details about the certificate, click **View Details**.

Related tasks

“Starting iKeyman” on page 382

Deleting certificates

You can delete signer or personal certificates that you no longer require from the key database.

To delete one or more certificates from the key database:

1. Optional: Create a backup of the certificate by extracting it to a different location, in case you require it again at a later date. You can do this for one or more of the certificates to be deleted.
2. Start iKeyman.
3. From the IBM Key Management window, click **Key Database File > Open**.
4. From the Open window, specify the file name and location of the key database (`omni.kdb`) that contains the certificates to be deleted. Then click **OK**.
5. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
6. From the **Key database content** area, perform the relevant action as follows:
 - To delete signer certificates, select Signer Certificates from the drop-down list.
 - To delete personal digital certificates, select Personal Certificates from the drop-down list.

7. Select the certificates to be deleted. You can select multiple certificates by using the Ctrl or Shift key.
8. Click **Delete** and then confirm the deletion.

Results

Each deleted certificate is removed from the IBM Key Management window, and from the key database.

Related tasks

“Starting iKeyman” on page 382

“Extracting certificates from a key database” on page 386

Changing the key database password

It is good practice to change the password for the key database regularly. From the iKeyman GUI, you are also prompted to change the password if you try to open the key database with an expired password.

To change the key database password as part of your standard procedure:

1. Start iKeyman.
2. From the IBM Key Management window, click **Key Database File > Open**.
3. From the Open window, specify the file name and location of the key database (`omni.kdb`) that requires a password change. Then click **OK**.
4. Type the current key database password in the Password Prompt window and click **OK**. The key database contents are shown in the IBM Key Management window.
5. Click **Key Database File > Change Password**. The Change Password window opens.
6. Complete this window as follows:

New Password

Type a password. As you type the characters, an indication of the password strength is given.

Note: Passwords are case sensitive, so whenever you are required to specify this password to open the key database, you *must* use the correct case to avoid errors.

Confirm New Password

Retype the password.

Set expiration time

Select this check box to specify a period after which the password will expire. Enter the period in days. The default is 60 days. If this check box is clear, the password never expires.

Stash the password to a file?

Select this check box to save the password in an encrypted format to a stash file. This is a mandatory requirement for Tivoli Netcool/OMNIBus.

OK Click this button to save the password and close the window.

Results

The password is encrypted and saved to the `omni.sth` stash file in the same location as the key database.

Related tasks

“Starting iKeyman” on page 382

Managing certificates from the command line

You can use the **nc_gskcmd** command-line utility for performing tasks that are available from the iKeyman GUI, and additional tasks that are not available in the GUI.

To manage certificates from the command line, run the following command:

```
$NCHOME/bin/nc_gskcmd modifier object action options
```

In this command:

- *modifier* is a command-line option that you can use to enable FIPS 140-2 mode, to set language preferences, and enable trace logging
- *object* is a command-line option indicating that an action is required on an object, typically a key database, certificate, or certificate request. This must be the first command-line option specified.
- *action* is a command-line option that defines a specific action to be taken on the object. This must be the second command-line option specified.
- *options* are mandatory and optional command-line options that are valid for the specified *object* and *action* pair. These command-line options can be in any order.

Note: Not all actions and their associated options are applicable for use in Tivoli Netcool/OMNIbus.

The following table lists each *modifier* for the **nc_gskcmd** command.

Table 92. Modifiers for *nc_gskcmd*

Modifier	Description
-fips [true false]	Enables or disables FIPS 140-2 mode.
-local <i>language</i>	Sets the language preference as specified by <i>language</i> .
-keydb <i>path</i>	Enables trace logging to the file specified by <i>path</i>

The following table lists each *object* and its associated set of *actions* for the **nc_gskcmd** command.

Table 93. Objects and corresponding actions for *nc_gskcmd*

Object	Action	Description
-keydb	-change pw	Changes the password for a key database.
	-convert	Converts the key database from one format to another.
	-create	Creates a key database.
	-delete	Deletes the key database.
	-expiry	Displays the expiry date of the password for a key database.
	-list	Displays the supported types of key database.

Table 93. Objects and corresponding actions for `nc_gskcmd` (continued)

Object	Action	Description
	-stashpw	Stashes the password of a key database into a file.
-cert	-add	Adds a CA certificate from a file into a key database.
	-create	Creates a self-signed certificate.
	-delete	Deletes a certificate.
	-details	Shows the details of a specific certificate.
	-export	Exports a personal certificate and its associated private key from a key database into a PKCS#12 file or another key database.
	-extract	Extracts a certificate from a key database.
	-getdefault	Shows the default personal certificate.
	-import	Imports a certificate from a key database or a PKCS#12 file.
	-list <i>string</i>	Lists the certificates in the key database. The values are all, personal, CA, and site. The default is to list all certificates. Specifying -list on its own also lists all certificates.
	-modify	Modifies a certificate. Note: Currently, the only field that can be modified is the Certificate Trust field.
	-receive	Receives a certificate from a file into a key database.
	-setdefault	Sets a personal certificate as the default certificate.
	-sign	Signs a certificate that is stored in a file with a certificate that is stored in a key database, and then stores the resulting signed certificate in a file.
-certreq	-create	Creates a certificate request.
	-delete	Deletes a certificate request from a certificate request database.
	-details	Shows the details of a specific certificate request.
	-extract	Extracts a certificate request from a certificate request database into a file.
	-list	Lists all certificate requests in the certificate request database.
	-recreate	Re-creates a certificate request.
-help		Displays help information for the <code>nc_gskcmd</code> command.
-version		Displays version information about the <code>nc_gskcmd</code> command and exits.

The following table lists the *options* that are valid for the specified *object* and *action* pair.

Table 94. Options for object and action pairs

Option	Description
-ca TRUE FALSE	Adds the Basic Constraints extension to the self-signed certificate. The extension is added with a CA:true and PathLen:max int if the value passed is true, or not added at all if the value passed is false.
-crypto <i>string</i>	Indicates a PKCS#11 cryptographic device operation.
-db <i>string</i>	Specifies the fully-qualified path name of a key database.
-default_cert YES NO	Sets a certificate as the default certificate to be used for client authentication. The default is no.

Table 94. Options for object and action pairs (continued)

Option	Description
-dn <i>string</i>	Specifies the X.500 distinguished name. Enter the value as a quoted string in the following format: "CN= <i>common_name</i> , O= <i>organization</i> , OU= <i>organization_unit</i> , L= <i>location</i> , ST= <i>state_province</i> , ZIP= <i>postal_code</i> , C= <i>country</i> " For example: "CN=Jane Doe, O=IBM, OU=Java Development, L=Endicott, ST=NY, ZIP=13760, C=country" Only CN is mandatory.
-encryption <i>string</i>	Specifies the strength of encryption that is used in the certificate export command. The value can be strong or weak. The default is strong.
-expire <i>integer</i>	Specifies the expiration time of either a certificate or a key database password (in days). The duration is 0 to 7300 (that is 20 years). The default is 60 days for a key database password. An expiry of 0 means that the password associated with the key database does not expire. For a self-signed certificate, specify a range from 366 to 7300.
-file <i>string</i>	Specifies the file name of a certificate or a certificate request (depending on specified <i>object</i>).
-format <i>string</i>	Specifies the format of a certificate. The value can be either <code>ascii</code> for Base64-encoded ASCII, or <code>binary</code> for Binary DER data. The default is <code>ascii</code> .
-label <i>string</i>	Specifies the descriptive text that is used to identify a certificate or a certificate request in the key database.
-new_format <i>string</i>	Specifies a new format for the key database.
-new_label <i>string</i>	Specifies a new certificate label or alias to replace an existing label.
-new_pw <i>string</i>	Specifies a new key database password.
-old_format <i>string</i>	Specifies the old format of the key database.
-pfx	Interprets a PKCS#12 file as a Microsoft pfx variant.
-pw <i>string</i>	Specifies the password for the key database or PKCS#12 file. In FIPS 140-2 mode, the following restrictions apply to passwords: <ul style="list-style-type: none"> • The minimum password length is 14 characters. • A password must have at least one lower case character, one upper case character, and one digit or special character. • Each character must not occur more than three times in a password. • No more than two consecutive characters of the password can be identical
-size <i>integer</i>	Specifies the key size. The values are 512, 1024, and 2048. The default is 1024.
-stash	Stashes the key database password to a <i>key_database_name.sth</i> file in the same location as the key database file.
-san_dnsname	Adds one or more DNS names to the Subject Alternate Name attribute. Must be in "preferred name syntax" according to RFC 1034.
-san_emailaddr	Adds one or more e-mail addresses to the Subject Alternate Name attribute. Must be an "addr-spec" as defined in RFC 822.

Table 94. Options for object and action pairs (continued)

Option	Description
-san_ipaddr	Adds one or more IP addresses to the Subject Alternate Name attribute. Must be a string according to RFC 1338 and RFC 1519.
-secondaryDB	Specifies secondary key database support for PKCS#11 device operations.
-secondaryDBpw	Specifies the password for the secondary key database for PKCS#11 device operations.
-showOID	Displays the entire certificate or certificate request.
-target <i>string</i>	Specifies the destination file or key database into which a certificate is being exported or imported.
-target_pw <i>string</i>	Specifies the password for the key database if -target specifies a key database.
-target_type <i>string</i>	Specifies a type for the database that is specified by the -target command-line option. The allowable value for Tivoli Netcool/OMNIBus is <i>cms</i> , which specifies a CMS key database.
-tokenlabel <i>string</i>	Specifies a label for a PKCS#11 cryptographic device.
-trust <i>string</i>	Specifies the trust status of a CA certificate. The value can be <i>enable</i> or <i>disable</i> . The default is <i>enable</i> .
-type <i>string</i>	Specifies the type of database. The allowable value for Tivoli Netcool/OMNIBus is <i>cms</i> , which indicates a CMS key database.
-usereasoncode	Returns a multi-valued error code if the nc_gskcmd command fails, or 0 if it is successful.
-x509version <i>integer</i>	Specifies the version of X.509 certificate to create. The values are 1, 2 and 3. The default is 3.

For further information on the usage of these command-line options, see the *IBM Global Security Kit Secure Sockets Layer Introduction and iKeyman User's Guide*, SC32-1700.

Chapter 23. IPv6 configuration

Tivoli Netcool/OMNIBus provides support for both IPv4 and IPv6. The components can operate and coexist on a network supporting IPv4 only, IPv6 only, or a dual IPv4 and IPv6 configuration.

The Tivoli Netcool/OMNIBus server components operate in IPv6 and IPv4 environments as follows:

- The ObjectServer can process events that originate from both IPv4 & IPv6 networks. When the ObjectServer is running on a dual-stacked host, the host name returned to the client in response to a command is the host name of the server that corresponds to the IP version that the client is running. For example, a client running IPv4 receives the IPv4 host name of the ObjectServer, and a client running IPv6 receives the IPv6 host name of the ObjectServer.
- In dual IPv4 and IPv6 environments, the unidirectional and bidirectional ObjectServer gateways can listen on both interfaces on the communications socket.
- The proxy server can support connections between probes and ObjectServers that are running on IPv4 and IPv6 hosts.
- In dual IPv4 and IPv6 environments, the process agent can listen on both interfaces on the communications socket.

The following IPv6 address formats are supported:

- Eight groups of four hexadecimal characters, separated by colons; for example, ABCD:EF01:2345:6789:ABCD:EF01:2345:6789
- Where all 16 bits are zero, the segment can be replaced with a colon (:). For example, the address 1010:0000:0000:0000:ABCD:EF01:2345:6789 can be written as 1010::ABCD:EF01:2345:6789.
- IPv4 addresses can be represented as IPv6 addresses. For example:
 - 0:0:0:0:0:192.101.50.5
 - 0:0:0:0:FFFF:103.27.35.8These addresses can also be represented as:
 - ::192.101.50.5
 - ::FFFF:103.27.35.8

Configuring IPv6 support

When you install or change a server component on any host in your Tivoli Netcool/OMNIBus system, you must set up the component to communicate with other components. Server communications information is configured by using the Server Editor.

When setting up server communications information in a dual IPv6 and IPv4 environment, you can use IPv6 addresses, IPv4 addresses, or host names to specify the name of the computer on which the server component is installed. If you want to use a host name instead of an IPv6 address, you must configure host lookup on your operating system.

UNIX configuration

After using the Server Editor (or `nco_xigen`) to set up component communications with IPv6 or IPv4 addresses, the server communications information is saved in an interfaces file `$NCHOME/etc/interfaces.arch`, where *arch* represents your operating system directory. If you have a distributed installation, with different Tivoli Netcool/OMNIbus components running on multiple systems in your network, you must distribute the interfaces file that contains the communications information, to each Tivoli Netcool/OMNIbus system.

If you are running Tivoli Netcool/OMNIbus components on more than one UNIX operating system, you must generate compatible interfaces files for each operating system and then distribute the files to the relevant hosts. You can configure component communications on *one* Tivoli Netcool/OMNIbus computer, and then from that computer, generate interfaces files for all the available operating systems. You can generate interfaces files for each operating system from the command line, or you can use the Server Editor to generate interfaces files, as follows:

- From the command line, enter the following command:
`$NCHOME/bin/nco_igen -all`
This generates an interfaces file named `$NCHOME/etc/interfaces.arch` for each operating system, where *arch* represents the UNIX operating system name; for example, `interfaces.hpux11` and `interfaces.solaris2`. Copy the relevant operating system interfaces file to the `$NCHOME/etc` directory on each of the host computers.
- From the Server Editor, specify your communications settings and then select the **Generate All** check box. Click the **Apply** button to generate interfaces files named `$NCHOME/etc/interfaces.arch`, where *arch* represents individual UNIX operating system names. Copy the relevant operating system interfaces file to the `$NCHOME/etc` directory on each of the host computers.

Example IPv4 and IPv6 configurations in the omni.dat file

On UNIX, the connections data file `$NCHOME/etc/omni.dat` is used to create the interfaces file for Tivoli Netcool/OMNIbus communications. Example IPv4 and IPv6 settings within this file are shown here.

Example: Configuring the omni.dat file with a host name and an IPv6 address

Example entries in the `omni.dat` file with a host name and an IPv6 address are as follows:

```
[NCOMS]
{
    Primary:      presley 9000
}

[BARROW]
{
    Primary:      fec0:0000:0000:7777:0218:fcef:fe8c:4f3b 8002
}
```

Example: Configuring a dual stack IPv4 and IPv6 ObjectServer to listen on both IPv4 and IPv6 ports

To enable probes on an IPv6 computer to connect to a dual stack IPv4 and IPv6 ObjectServer computer, you must configure a backup ObjectServer using the IPv6 address of the ObjectServer. In the example `omni.dat` file, 192.168.0.1 is the IPv4

address of the dual stack IPv4 and IPv6 ObjectServer, and 2094:82a:2a6e:123:503:badd:fe43:f552 is its IPv6 address.

```
[MAINOBJ]
{
    Primary:      192.168.0.1 4100
    Backup:       2094:82a:2a6e:123:503:badd:fe43:f552 4100
}
```

If IPv4 and IPv6 domain names are configured on your network, you can also use the fully-qualified domain name of the ObjectServer computer as the Primary entry in `omni.dat`; for example `sf0.ipv4.domain.com` or `sf0.ipv6.domain.com`.

Windows configuration

On each Windows computer, use the Server Editor to set up component communications with IPv6 or IPv4 addresses, as required.

On Windows XP and Windows 2003 computers, you must additionally install the IPv6 protocol driver and configure an external IPv6 address. You can do this from the Control Panel by using the **Network Connections** utility. Open the Properties window for the Local Area Connection, and from the **General** tab, install the IPv6 protocol driver and configure the external IPv6 address. See your operating system documentation for complete information on IPv6 setup.

Loading remote event list configurations by using HTTP or FTP on Windows

If you want to load an event list configuration (`.elc`) from a remote server by using HTTP or FTP, you can specify an IPv4 or IPv6 address for the server name.

If you are running on Windows and intend to access a `.elc` file on a remote server by using the IPv6 address of the server, be aware that the Windows event list requires version 7 of the Windows system file `wininet.dll`. This version of the file provides support for IPv6 literal addresses in host names, and is available from Internet Explorer 7 onwards. Therefore, you must ensure that either of the following conditions is met:

- Version 7 of `wininet.dll` is installed on the computer from which you run the `NCOEvent.exe` command. This file is typically stored in `C:\WINDOWS\system32`.
- Internet Explorer 7 is installed.

Tip: You might also find it useful to verify whether the event list can load the `.elc` file. To do this, enter the IPv6 format of the URL to the file within the **Address** field in a Web browser, to see whether you can access the `.elc` file.

For further information about opening event list configurations from remote servers, see the *IBM Tivoli Netcool/OMNIBus User's Guide*.

Probe rules file configuration

You can include a number of secondary rules files in your main rules file by using the `include` statement.

If you want to include a remote probe rules file that is stored on an IPv6 Web server, you must use brackets `[]` to delimit the IPv6 address in the Web address. For example:

```
include "http://[fed0::7887:234:5edf:fe65:348]:8080/probewatch.rules"
```

For further information on embedding multiple rules files in a rules file, see the *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*.

Related concepts

“Configuring server communication details in the Server Editor” on page 240

Related tasks

“Setting up distributed installations” on page 248

Chapter 24. Multicultural support

Tivoli Netcool/OMNIBus uses the International Components for Unicode (ICU) library for character set conversion, and supports the character encodings that ICU supports.

ICU is a cross-platform Unicode-based globalization library that includes support for locale-sensitive string comparison, date formatting, time formatting, number formatting, currency formatting, message formatting, text boundary detection, and character set conversion. For a list of the supported character encodings, see the ICU Web site at <http://www.icu-project.org/>.

Text data is automatically converted between character encodings if a client and an ObjectServer are using different encodings. Run the ObjectServer in an encoding that includes all of the characters that are used in all locations in your Tivoli Netcool/OMNIBus deployment. If your deployment uses data from different languages, run the ObjectServer in 8-bit Unicode Transition Format (UTF-8) encoding to ensure that it can handle all text data.

Note: If you are using external authentication sources to verify user credentials, you must establish whether these authentication sources also support multi-byte characters. If multi-byte characters are not supported, you must specify user names and passwords by using ASCII characters.

Setting your locale

The language, character set, sort order, and data format settings that are used at run time are determined by your locale settings. You can use the localization environment variables on UNIX and Linux, or the Control Panel on Windows, to set your locale.

On UNIX and Linux

On UNIX and Linux, set one or more of the following localization environment variables to help define the locale settings for your environment. For example, on Solaris you can set the variables in `/etc/default/init`, and on AIX, you can set the variables in `/etc/environment`.

Table 95. Localization environment variables for UNIX and Linux

Environment variables	Description
LC_ALL	The LC_ALL value takes precedence over the values of all the other environment variables, and if set, determines the language, character set, sort order, and data formats.
LC_COLLATE	This environment variable defines the collating sequence (or sort order).
LC_CTYPE	This environment variable defines the character classification and case conversion.
LC_MESSAGES	This environment variable defines the language and character set for messages.

Table 95. Localization environment variables for UNIX and Linux (continued)

Environment variables	Description
LC_MONETARY	This environment variable defines the format for monetary numeric information.
LC_NUMERIC	This environment variable defines numeric, non-monetary formatting.
LC_TIME	This environment variable defines the date and time formats.
LANG	If LC_ALL is not set, the LANG value determines the language, character set, and sort order. Different elements of the LANG value can be overridden by setting the LC_COLLATE, LC_CTYPE, LC_MESSAGE, and LC_TIME environment variables.

On Windows

To set your locale on Windows, use the **Regional Settings** or **Regional and Language Options** item in the Control Panel. Configure your settings as follows in the window that is displayed:

1. From the **Formats** tab, select the language to be used for displaying dates, times, currencies, and numbers.
2. From the **Language for non-Unicode programs** area on the **Advanced** or **Administrative** tab, select the language in which you intend to run Tivoli Netcool/OMNIBus.

You will be asked to reboot your computer for the new settings to take effect.

The languages set in these steps must be identical.

You can choose to run the ObjectServer, ObjectServer Gateway, **nco_dbinit** utility, **nco_postmsg** utility, and individual probes and gateways in UTF-8 encoding by using a Windows-specific command-line option **-utf8enabled**. This command-line option controls the encoding of data that is passed into, or generated by, these applications, and must be set to TRUE to run in UTF-8. When **-utf8enabled** is set to FALSE (the default), the default system code page is used.

The following table describes the encodings that can be used for data that is passed into these applications, and data that is generated by these applications.

Table 96. Acceptable encodings for input and output

Application	Command-line input	File input	File output
ObjectServer	String-based command-line options that are entered at the command line are encoded in the system default code page only.	Affected file: Properties file (.props) If -utf8enabled is set to TRUE, your property settings are encoded in UTF-8. If -utf8enabled is set to FALSE, your property settings are encoded in the default system code page.	Affected files: Properties file and log file (.props and .log) If -utf8enabled is set to TRUE, the output written to these files is encoded in UTF-8. If -utf8enabled is set to FALSE, the file outputs are encoded in the default system code page.

Table 96. Acceptable encodings for input and output (continued)

Application	Command-line input	File input	File output
ObjectServer Gateway	String-based command-line options that are entered at the command line are encoded in the system default code page only.	Affected files: Map file and properties file (.map and .props) If -utf8enabled is set to TRUE, your property and map file settings are encoded in UTF-8. If -utf8enabled is set to FALSE, your property and map file settings are encoded in the default system code page.	Affected file: Log file (.log) If -utf8enabled is set to TRUE, the output written to this file is encoded in UTF-8. If -utf8enabled is set to FALSE, the file output is encoded in the default system code page.
nco_dbinit	String-based command-line options that are entered at the command line are encoded in the system default code page only.	Affected files: SQL import files and properties file (.sql and .props) If -utf8enabled is set to TRUE, your SQL and property settings are encoded in UTF-8. If -utf8enabled is set to FALSE, your SQL and property settings are encoded in the default system code page.	Not applicable
nco_postmsg	String-based command-line options that are entered at the command line are encoded in the system default code page only.	Affected file: Properties file (.props) If -utf8enabled is set to TRUE, your property settings are encoded in UTF-8. If -utf8enabled is set to FALSE, your property settings are encoded in the default system code page.	Affected file: Log file (.log) If -utf8enabled is set to TRUE, the output written to this file is encoded in UTF-8. If -utf8enabled is set to FALSE, the file output is encoded in the default system code page.

To use UTF-8 encoding, create and run the ObjectServer in this encoding, and determine whether to also run the supported probes and gateways, and **nco_postmsg** in UTF-8, or whether to run these client applications in the default system locale. For information about the probes and gateways that can run in UTF-8 encoding on Windows, see the individual probe and gateway publications. If using SSL, take note that the key database path (%NCHOME%\ini\security\keys) must contain only characters that are supported by the default system code page.

Also note that process agents and proxy servers do not support UTF-8 encoding on Windows, and run in the default system encoding only.

Additional information

On Windows, running in a UTF-8 encoding ensures compliance with the GB18030 standard for Chinese characters. On UNIX and Linux, you can use the localization variables to specify a GB18030 locale. For the Web GUI component, additional steps are required for GB18030 compliance.

If you want to add a new locale, you will be required to install the appropriate locale module or language pack on your computer. See your operating system documentation for further information.

Note: When using Netcool/OMNIBus Administrator, you must ensure that the character set encoding of each ObjectServer being managed has a corresponding entry in the file `$NCHOME/omnibus/java/jars/csemap.dat`. This file provides a mapping between Sybase and JRE character set encoding naming conventions. If the character set encoding of an ObjectServer is missing from `csemap.dat`, you must add a mapping to this file by using the format:

Sybase_encoding Java_encoding

For example:

`ascii_7 ASCII`

Related tasks

“Configuring the Web GUI for GB18030 characters” on page 213

Related reference

“Properties and command-line options for `nco_dbinit`” on page 232

Identifying which locales are supported on your computer

You can run the **locale** command on UNIX, or use the Windows Control Panel to list all of the locales that are supported on your computer.

To verify which locales are supported on your computer:

- On UNIX, run the following command:

```
locale -a
```

Tip: You can also use the **locale** command without any command-line options to list the current locale, and use the **locale charmap** command to display the encoding.

- On Windows, use the **Regional Settings** or **Regional and Language Options** item in the Control Panel.

What to do next

You can assign any of the listed locales to the `LANG` or `LC_*` environment variables. The listed locales are case-sensitive so ensure that you use the correct case when assigning them to environment variables. You can also view the locales supported for the UNIX desktop components and desktop configuration.

Related tasks

“Identifying which locales are supported for the UNIX desktop” on page 407

Enabling or disabling localized sorting

Use the ObjectServer property **Store.LocalizedSort** to enable or disable localized sorting. Localized sorting is disabled by default for optimal system performance.

The **Store.LocalizedSort** ObjectServer property either enables you to perform standard C library string comparisons (the default), or enable localized sorting. When localized sorting is enabled, you can additionally use the **Store.LocalizedSortCaseSensitive** property to control the case sensitivity of the sort order.

Example

Example localized sorting

When localization is disabled, Å is treated as a variant of A and the two characters will sort near each other.

When localization is enabled in a Danish locale, Å is treated as a separate letter that sorts just after Z.

Identifying which locales are supported for the UNIX desktop

All the locales that are supported for use in the UNIX desktop are installed in the location `$NCHOME/omnibus/desktop/locale/arch`, where *arch* represents the operating system directory.

Within this location, a directory or symbolic link exists for each of the locales for which desktop configuration is supported.

Configuring fonts for the UNIX desktop

If you want to view the UNIX desktop in your locale, you might find it necessary to configure the fonts that are needed to display the text in the encoding of your locale.

The Tivoli Netcool/OMNIBus installation includes resource files that contain definitions for the user interface elements of the UNIX desktop applications; for example, definitions for window dimensions, font selections, colors, string values for window titles, menus, buttons, icons, field labels, and message strings.

Resource file translations are available for the following locales: English, French, German, Japanese, Korean, Russian, Spanish, Simplified Chinese, and Traditional Chinese. Additionally, locales that use the ISO-8859-1 character set are expected to display fonts correctly, with the English setting on. Other character sets might require some font configuration.

The resource files are stored in the following location:

`$NCHOME/omnibus/desktop/locale/arch/locale_name/app-defaults`

Where *arch* is the operating system directory and *locale_name* is the full locale name; for example `en_GB.ISO8859-1`. Note that some locale names might be symbolic links with abbreviated names.

The resource files include:

- NCO: Definitions for the Conductor, and its associated Filter Builder, and View Builder
- NCOBanner: Definitions for the Conductor splash screen
- NCOELCT: Definitions for the transient event list
- NCOEvent: Definitions for the Event List monitor box window, the event list, and associated windows such as the Login window, Filter Builder, and View Builder
- NCOHelp: Definitions related to the online help; this file might not have any definitions
- NCOMessage: Definitions for the messaging dialog box that can be used with tools
- NCOXigen: Definitions for the Server Editor
- NCOXprops: Definitions for the Properties Editor

If your locale is not included in the Tivoli Netcool/OMNIBus installation package, the resource files for the en_US.ISO8859-1 locale are used by default. You can configure your installation to use another locale that is not provided in the installation package. If your locale uses a character set encoding other than ISO-8859-1, you must additionally ensure that you define a font that can accurately render the resource file characters into the characters for your locale.

To configure another locale and font set:

1. Run the following command to list all supported locales:

```
locale -a
```

2. Set the LC_ALL environment variable to one of these locales.

3. Run the following command to display your character encoding:

```
locale charmap
```

Make a note of the encoding because it will be required later.

4. To create a set of localized resource files in a font that renders correctly, go to the directory \$NCHOME/omnibus/desktop/locale/*arch*, where *arch* represents your operating system directory. You must copy a set of resource files from a locale that contains suitable fonts for your encoding and then customize the copied files. For example, to create files for the Arabic locale (ar), create a directory with the locale name, and copy the resource files for the en_US.ISO8859-1 locale:

```
cd $NCHOME/omnibus/desktop/locale/arch
```

```
mkdir ar
```

```
cd ar
```

```
cp -r ../en_US.ISO8859-1/* .
```

The resources files (prefixed NCO), images subdirectory, and default event list configuration files are copied to the ar directory. You must now look for a suitable set of fonts on your system, which matches the application font in the resource file.

5. From the command line, enter the appropriate command for your operating system:

Operating System	Command
AIX	/usr/X11R6/bin/xlsfonts -fn " <i>font_name</i> "
HP-UX	/usr/bin/X11/xlsfonts -fn " <i>font_name</i> "
Linux (Red Hat)	/usr/X11R6/bin/xlsfonts -fn " <i>font_name</i> "
Solaris	/usr/openwin/bin/xlsfonts -fn " <i>font_name</i> "

In this command, *font_name* is the character encoding that was output in step 3 on page 408. Specify this value as a wildcard by using asterisks (*). Note that you must enclose the value in quotation marks to prevent the shell from interpreting the asterisks in the text. For example:

```
/usr/openwin/bin/xlsfonts -fn "*-iso8859-6" The list of matching fonts is shown.
```

6. Preview each of these fonts to determine whether they are suitable. For each font, enter the following command:

```
xfd -fn font_name
```

Where *font_name* is one of the matching font names returned in the previous step. A window opens, showing the full name of the font and a grid containing one character per cell. You might need to use the **Next Page** and **Previous Page** buttons to view all the characters. When you have identified suitable fonts, you can add the font set to your resource files.

7. Open each of the resource files named NCO in turn, to change the font. For example, for the event list resources, you must set NCOEvent*fontList, NCOEvent*sub_matrix.labelFont, *view_builder*display_matrix.labelFont, and NCOEvent*info_matrix.labelFont to font sets that contain all fonts required for the locale.

UNIX font names are of the form:

```
-foundry-font family-weight-slant-set width-serif-pixels-points-hres-vres-spacing-average  
width-character set-encoding
```

You can specify font names with wildcards. For example, the default font for the event list is

```
-adobe-helvetica-bold-r-normal--12-*75-75-*--iso*--*
```

For Arabic, you can replace this with:

```
-dt-interface user-bold-r-normal-m serif-14-140-75-75-p-188-iso8859-6
```

When using EUC character sets, several fonts are required at one time; for example, EUCJIS (Japanese) requires iso8859-1, jisx0201.1976-0, jisx0208.1983-0, and jisx0212.1990-0 fonts. You can specify such a font set with one or more font names containing wild cards. (Fonts within a font set are separated with a semicolon and font sets are ended with a colon).

8. If required, change other settings in the resources as follows:
 - Specify default widths (in pixels) of the windows. You might need to adjust these values to accommodate your font and ensure that text labels on the windows are displayed appropriately.
 - Replace string values for window titles (*.title), button labels (*.labelString), messages (*.messageString), and other textual elements with your translated text. Make sure that the translated text uses the character encoding of your locale.
9. Save your changes to the files. You can now run Tivoli Netcool/OMNIbus with the correct locale and fonts.

Setting up the ObjectServer to use translated user interface text in the desktop

The ObjectServer database contains some configuration data that is displayed in the UNIX and Windows desktop (that is, the event list and Conductor). When you initialize the ObjectServer database, this configuration data is read from the desktop SQL definition file, which inserts default values into the desktop tables, including default colors, column visuals, conversions, tools, and menus.

The default desktop SQL definition file is `$NCHOME/omnibus/etc/desktop.sql`.

The ObjectServer uses a single `desktop.sql` file, and all event lists and Conductors that are connected to a given ObjectServer display the strings in the same language.

Translations of the default configuration data are available in the following languages: Japanese, Korean, Simplified Chinese, and Traditional Chinese.

Translated strings are provided in separate `desktop.sql` files for:

- Column visuals that are used as the default names for columns in the event list
- Conversions that cause numeric event data to be displayed as strings for fields such as Severity, Acknowledged, Type, and NmosManagedStatus
- The names of items on the **Tool** menus

The translated files for each supported language are stored in `$NCHOME/omnibus/etc/locale/locale_name/desktop.sql`, where *locale_name* is the full locale name. To use any of these files, you must specify which file you want to use, when initializing the ObjectServer database (by using the **nco_dbinit** command).

Important: You must run **nco_dbinit** in the locale in which you are going to start and run the ObjectServer. If you want to run in UTF-8 encoding, you must convert the `.sql` files encoded in natural languages such as Chinese or Japanese, into UTF-8. Then specify the `-utf8enabled` command-line option with a setting of `TRUE` when you run **nco_dbinit**.

To initialize the database in your required locale, run the **nco_dbinit** command with the `-desktopfile` command-line option, as follows:

```
$NCHOME/omnibus/bin/nco_dbinit -server servername -desktopfile string
```

In this command, *servername* is the name of the new ObjectServer, and *string* is the path and file name of the `desktop.sql` file for your required locale. For example:

```
$NCHOME/omnibus/bin/nco_dbinit -server DENCO -desktopfile  
$NCHOME/omnibus/etc/locale/zh_TW.EUC/desktop.sql
```

You must then start the ObjectServer in the same locale used when running the **nco_dbinit** command. For example:

```
$NCHOME/bin/nco_objserv -name DENCO
```

You can modify the configuration data after the ObjectServer is created by using Netcool/OMNIBus Administrator, and you can translate the strings into other languages.

Related tasks

“Creating an ObjectServer” on page 231

Chapter 25. Extending the functionality of Tivoli Netcool/OMNIBus

Tivoli Netcool/OMNIBus includes a set of resources, which you can use to extend the functionality of the product. Integration with other Tivoli products is required for some of the customizations provided.

The resources are installed in the `$NCHOME/omnibus/extensions` directory, and include customizations to:

- Set up a multitiered environment to increase performance and event handling capacity.
- Configure high availability.
- Perform file transfer operations between computers.
- Enhance probe rules for the detection of event floods and unusually low or high event rates.
- Configure self monitoring of probes to collect metrics about the amount of memory used for various processing operations, and the number of events received, discarded, and generated.
- Support predictive analytics in an integrated Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring environment.
- Enable events from IBM Tivoli Application Dependency Discovery Manager (TADDMM) to be monitored in Tivoli Netcool/OMNIBus.
- Enable event management of a virtual environment by using the joint capabilities of Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring.

These resources are always installed regardless of the installation feature chosen.

Overview of the `$NCHOME/omnibus/extensions` directory

The `$NCHOME/omnibus/extensions` directory extends the capability of Tivoli Netcool/OMNIBus through a set of sample or template files that can be used as a starting point for your required configuration.

The `$NCHOME/omnibus/extensions` directory contains the following subdirectories:

- `control_shutdown`
- `eventflood`
- `itmdeploy`
- `itmpredictive`
- `itmvirtualization`
- `multitier`
- `roi`
- `taddm`

Each of these subdirectories contains read-only files that provide a sample configuration. Treat the original files as templates that are available for reference purposes. To extend the capability of Tivoli Netcool/OMNIBus by using the contents of these subdirectories, you are generally required to make copies of the

files and then remove the read-only permissions before configuring as directed in the relevant procedures. In some cases, you can run commands that reference some of the files.

Note: The read-only permissions are not enforced on Windows.

Contents of the control_shutdown directory

The control_shutdown directory stores an SQL script that can be used to update the ObjectServer schema with the customizations required for configuring a controlled shutdown of an ObjectServer.

Further information about configuring a controlled shutdown is available within this installation guide.

Contents of the eventflood directory

The eventflood directory stores sample secondary rules files that can be used to detect when a probe is subject to an event flood or an anomalous rate of receipt of events. A flood rules file is provided with all the rules for determining the current event rates and the action to take during an event flood, or an unusually high or low rate of receipt of events. A flood configuration rules file contains variables that are used to configure the flood rules file.

For further information about configuring and detecting event floods or anomalous event rates, see the *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*.

Contents of the itmdeploy directory

The itmdeploy directory stores sample files that can be used to:

- Retrieve files that you want to review or update, for Tivoli Netcool/OMNIBus and probe installations that have been deployed to remote computers.
- Transfer updated files back to the remote computers.

Files that are external to your Tivoli Netcool/OMNIBus installation directory can also be retrieved and replaced on remote computers.

This customization requires integration with IBM Tivoli Monitoring. A file transfer utility with its corresponding properties file, and a .jar file are provided for performing file transfer operations.

Further information about the remote deployment of probes and file transfer operations is available within this chapter of the installation guide.

Contents of the itmpredictive directory

The itmpredictive directory stores sample files that are required to configure Tivoli Netcool/OMNIBus so that predictive events generated in IBM Tivoli Monitoring can be viewed in the Active Event List or the desktop event list. The ability to view predictive events requires integration with IBM Tivoli Monitoring, and the Probe for Tivoli EIF.

Sample files are provided for:

- Adding triggers, fields, a class ID and its conversion, and tools to the ObjectServer

- Processing and mapping predictive event data to alert data that can be inserted into the alerts.status table in the ObjectServer
- Creating the filter, view, and event list configuration that can be used in the event list
- Creating the filter, view, tools, prompts, and menu that can be used in the Active Event List

Further information about configuring predictive events is available within this chapter of the installation guide and the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*. For further information about monitoring predictive events, see the *IBM Tivoli Netcool/OMNIBus User's Guide* and the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Contents of the itmvirtualization directory

The itmvirtualization directory stores sample files that are required to configure Tivoli Netcool/OMNIBus to support event management for a virtual environment. This customization requires integration with IBM Tivoli Monitoring, and the Probe for Tivoli EIF.

Sample files are provided for:

- Adding triggers and a database table to the ObjectServer
- Processing and mapping situation event data to alert data that can be inserted into the alerts.status table and a custom table in the ObjectServer
- Reversing the changes made to the ObjectServer schema

Further information about configuring event management of a virtual environment is available within this chapter of the installation guide.

Contents of the multitier directory

The multitier directory stores sample files that can be used to configure a multitiered architecture of ObjectServers and ObjectServer Gateways that are installed in collection, aggregation, and display layers. Map definition files, properties files, and table replication definition files are provided for configuring the ObjectServer Gateways. SQL scripts are also available for updating the ObjectServer schema, and for reversing the applied changes.

Further information about configuring a multitiered architecture is available within this installation guide.

Contents of the roi directory

The roi directory stores an SQL script that can be used to update the ObjectServer schema, and a sample secondary rules file that can be used to configure a probe for self monitoring.

A set of sample reports are also provided that require user customization, and integration with Tivoli Data Warehouse and Tivoli Common Reporting. Working knowledge of these components is required to support this configuration.

For further information about configuring probes for self monitoring, see the *IBM Tivoli Netcool/OMNIBus Probe and Gateway Guide*.

Contents of the taddm directory

The taddm directory stores sample files that are required to configure Tivoli Netcool/OMNIbus so that events that are generated in TADDM can be viewed in the Active Event List or the event list. The ability to view these events requires integration with TADDM, and the Probe for Tivoli EIF.

Sample files are provided for:

- Adding the required class ID, conversion, menu, and tools to the ObjectServer
- Processing and mapping TADDM event data to alert data that can be inserted into the alerts.status table in the ObjectServer
- Adding the required menu and tools to the Web GUI component

Further information about configuring TADDM events is available within this chapter of the installation guide and the *IBM Tivoli Netcool/OMNIbus Web GUI Administration and User's Guide*. For further information about monitoring TADDM events, see the *IBM Tivoli Netcool/OMNIbus User's Guide* and the *IBM Tivoli Netcool/OMNIbus Web GUI Administration and User's Guide*.

Related concepts

Chapter 9, "Configuring high availability," on page 285

"Deploying probes remotely" on page 451

"Enabling predictive eventing and predictive analytics"

"Managing virtualized environments" on page 443

Chapter 8, "Configuring and deploying a multitiered architecture," on page 251

"Enabling support for TADDM events" on page 435

Enabling predictive eventing and predictive analytics

In an integrated Tivoli Netcool/OMNIbus and IBM Tivoli Monitoring environment, you can use the built-in predictive analytics capability to identify potential performance and capacity problems that could result in performance degradation or service outages. Within this environment, you can generate events that represent predictions for systems that are in danger of an impending threshold violation, and which require attention.

Note: This section assumes that you have a working knowledge of IBM Tivoli Monitoring.

Predictive event

IBM Tivoli Monitoring and the Probe for Tivoli EIF can be configured to forward predictive events to Tivoli Netcool/OMNIbus, and the resulting alerts can then be monitored in the Active Event List or the desktop event list.

Depending on which predictive eventing and analytics functions you configure, the following types of predictive event can be generated:

- Predictive events based on the event rate: If the linear trending function is configured, these predictive events display predictions of problems with the event rate of a specific device, for example, if a monitored device is showing an increased error event rate, and will exceed a defined threshold within seven days.

- Predictive events based on deviations from the baseline: If the baselining function is configured, these predictive events display deviations from a defined average event rate, which is calculated from archived data.
- Predictive events based on linear trending using historical data that is collected from monitoring agents in the IBM Tivoli Monitoring environment.

The following usage scenario outlines possible actions to take when predictive events are forwarded to Tivoli Netcool/OMNIBus for display in the event list and Active Event List:

- When alert data for a predictive event is displayed in the event list or Active Event List, begin to gather initial information on the predicted problem, such as the location of the problem and the number of days before the threshold is violated.
- Investigate the reasons for the prediction by looking through the actual events and other predictive events that are generated for the managed entity or node.
- When sufficiently satisfied with the validity of the prediction, begin to take corrective actions on the managed entity before the problem actually occurs.
- Alternatively, temporarily ignore the predictive event while monitoring for follow-up predictions and actual events that occur on the managed entity in the period between the generation of the predictive event and the threshold violation.
- For multiple predictive events, prioritize your response based on the order in which the events are displayed in the event list or Active Event List.

Linear trending for device event rates

Linear trending for device event rates uses IBM Tivoli Monitoring predictive analytics functionality to determine if the event rates received by the ObjectServer are likely to exceed upper thresholds within a defined period of time. Tivoli Performance Analyzer uses the event rates received from the ObjectServer to produce trends. If a threshold is violated within a defined time frame, a predictive event is sent to the ObjectServer. Support is provided for calculations over seven, 30, or 90 days. You can define critical and warning thresholds, when these thresholds are exceeded, the resulting predictive event has the corresponding severity.

The linear trending calculations use the Least Squares Regression method. This method approximates a linear pattern of use, over time, for selected attributes based on their values in the past.

Note: The Least Squares Regression method provides approximate data. You should apply a margin of approximately plus or minus 12 hours to the predictive events that are generated.

Baselining for device event rates

Baselining of the event rate per client device calculates the average event rate, per device, over a defined period of time. Event rate data, archived in Tivoli Data Warehouse, is used to build a corridor of normality. Current event rates are measured against the baseline average over a defined period of weeks for the current period of time. You define the upper and lower deviations from the average rate, that is the thresholds that the event rate must exceed. If a threshold is exceeded, IBM Tivoli Monitoring generates a situation, which is received by the Probe for Tivoli EIF. The Probe for Tivoli EIF in turn generates an event in Tivoli

Netcool/OMNIBus. The minimum period that data needs to build up is seven days, ideally you should allow for 14 days of data.

Related concepts

“Integration with other Tivoli products” on page 31

Configuration setup for predictive events

To configure and monitor predictive events, you require Tivoli Netcool/OMNIBus, the Probe for Tivoli EIF, and IBM Tivoli Monitoring to be installed within an integrated environment.

The following figure shows the required configuration setup for the product components in the integrated environment.

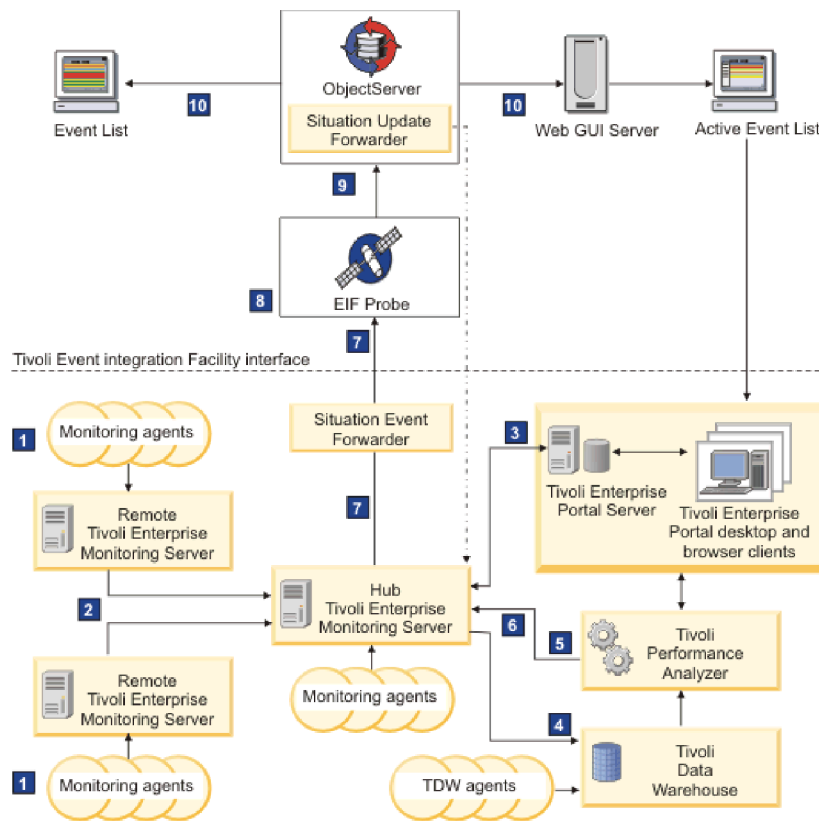


Figure 15. Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring configuration for predictive events

The configuration flow is as follows:

- 1** Tivoli Enterprise Monitoring Agents are installed on the systems or subsystems that you want to monitor. These agents collect data from the monitored or managed systems and send the data to one or more Tivoli Enterprise Monitoring Servers.
- 2** Tivoli Enterprise Monitoring Servers collect alerts received from the monitoring agents, and collect performance and availability data. The monitoring servers also manage the connection status of the agents. One monitoring server in each environment must be designated as the hub. The hub monitoring server controls the remote monitoring servers, and any monitoring agents that might be directly connected to the hub monitoring server.

- 3** A Tivoli Enterprise Portal Server provides the presentation layer for the data collected. The portal server retrieves data from the hub monitoring server in response to user actions from Tivoli Enterprise Portal clients, and presents the data to the portal clients in a Java-based user interface.
- 4** Tivoli Data Warehouse stores historical data that is collected from monitoring agents in your environment. You must configure IBM Tivoli Monitoring to retain data samples in history files and save these files to the Tivoli Data Warehouse on a regular basis. Two specialized agents (the Warehouse Proxy agent, and the Summarization and Pruning agent) interact with Tivoli Data Warehouse to receive, aggregate, and prune data.
- 5** The Tivoli Performance Analyzer component provides predictive capability that enables you to monitor resource consumption trends, anticipate future performance issues, and avoid or resolve problems more quickly. Tivoli Performance Analyzer performs linear trending over the historical data that exists in the Tivoli Data Warehouse, and enables you to simulate situations and events based on predicted behavior. By default, trending is done for disk space, CPU usage, memory usage, and network traffic. You can set up an analytical linear trend task that specifies the data to be analyzed, warning and critical thresholds, and the forecast time periods.
- Tivoli Performance Analyzer consists of a configuration tool, and predefined tasks, situations, and workspaces, which are all accessible from the Tivoli Enterprise Portal. Additionally, a Performance Analyzer warehouse agent interacts with:
- Tivoli Data Warehouse to retrieve the stored historical data collected by other agents
 - The portal server to receive the instruction to run the analytical task, and to perform analytical calculations on the data
 - The hub monitoring server to pass on the results of the trending
- 6** Whenever the analytical task runs at its specified frequency and schedule, prediction values are calculated for a set of predefined output attributes. These attributes are retrieved by the hub monitoring server and are:
- Stored in Tivoli Data Warehouse as new Tivoli Performance Analyzer attributes
 - Available for display in the Tivoli Enterprise Portal
 - Evaluated when predefined (or custom) situations run, in order to generate predictive events that provide advanced warning of potential problems
- 7** The hub monitoring server can be configured to forward these predictive events to Tivoli Netcool/OMNIBus ObjectServers for display. The hub monitoring server uses a situation event forwarder to map predictive events to Event Integration Facility (EIF) events, and uses the Tivoli EIF interface to forward the EIF events to an EIF receiver, which, in this case, is the Probe for Tivoli EIF.
- 8** The Probe for Tivoli EIF receives the events, processes the predictive event data, maps the data to ObjectServer fields, and then sends alerts to the ObjectServer. Modifications are required to the probe rules file to map the predictive event data to ObjectServer fields.
- 9** The ObjectServer requires some configuration to interpret and store the alerts. Dedicated fields in the alerts.status table are also used to store data that is unique to the predictive events received from IBM Tivoli Monitoring. The IBM Tivoli Monitoring event synchronization component

must also be installed on the ObjectServer host. This component provides customization resources that enable the ObjectServer and the Probe for Tivoli EIF to handle generic situation events and predictive events. The event synchronization component also includes a Situation Update Forwarder process, which enables updates to alerts to be sent back to the originating hub monitoring server.

Restriction: There is no capability to update predictive events within Tivoli Netcool/OMNIBus, and for those updates to be forwarded back to the originating hub monitoring server. This capability exists only for other types of situation events that are received from IBM Tivoli Monitoring. As such, any actions on predictive events must be performed in IBM Tivoli Monitoring.

- 10** Health and performance data is gathered from the ObjectServer by the Tivoli Netcool/OMNIBus IBM Tivoli Monitoring Health and Performance Agent and sent to the Tivoli Enterprise Monitoring Server.
- 11** Predictive events that are inserted into the alerts.status table can be viewed, filtered, and sorted in the Active Event List within the Web GUI, or in the event list. Launch-in-context functionality is also enabled from predictive events in the Active Event List, to the Tivoli Enterprise Portal. This feature enables you to view details about a predictive event in the relevant Tivoli Enterprise Portal workspace. To use the launch-in-context functionality, single sign-on must be configured.

Related tasks

“Configuring predictive eventing in your integrated environment” on page 428

Related reference

“Tivoli Netcool/OMNIBus configuration resources for predictive events” on page 425

Configuration setup for linear trending

To configure and monitor predictive analytics for linear trending, you require Tivoli Netcool/OMNIBus, the Probe for Tivoli EIF, and IBM Tivoli Monitoring to be installed within an integrated environment.

The following figure shows the required configuration setup for the product components in the integrated environment.

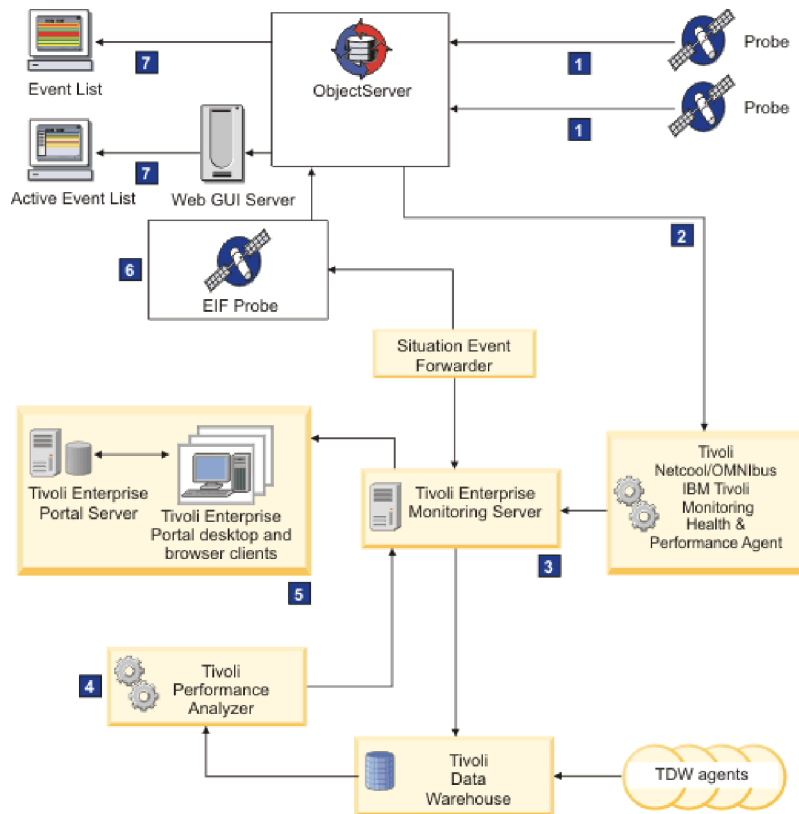


Figure 16. Tivoli Netcool/OMNibus and IBM Tivoli Monitoring configuration for predictive events

The configuration flow is as follows:

- 1** The probes are installed on the devices or systems that you want to monitor, and send the events to the ObjectServer. The automations in the ObjectServer detect the event rates for each monitored device.
- 2** The ObjectServer writes the event rate data to the event log files, and the IBM Tivoli Monitoring Health Performance Agent reads these files. To transform the events into IBM Tivoli Monitoring situations, the SQL and automations for predictive analytics must be run on the ObjectServer, as well as the appropriate situation classes. The Health Performance Analyzer feeds the situations to the Tivoli Enterprise Monitoring Server.
- 3** Tivoli Data Warehouse archives the historical event rate data. You must configure IBM Tivoli Monitoring to retain data samples in history files and save these files to the Tivoli Data Warehouse on a regular basis. Two specialized agents (the Archiving agent, and the Summarization and Pruning agent) interact with Tivoli Data Warehouse to receive, aggregate, and prune data.
- 4** Tivoli Data Warehouse feeds the archived event rate data to the Tivoli Performance Analyzer, where a trend uses the data to calculate the likely event rate in the future. Thresholds can be configured, which, when violated, generate a situation.

Tivoli Performance Analyzer consists of a configuration tool, and predefined tasks, situations, and workspaces, which are all accessible from the Tivoli Enterprise Portal. Additionally, a Performance Analyzer warehouse agent interacts with:

- Tivoli Data Warehouse to retrieve the stored historical data collected by other agents
- The portal server to receive the instruction to run the analytical task, and to perform analytical calculations on the data
- The hub monitoring server to pass on the results of the trending

- 5** All situations are parsed to Tivoli Enterprise Portal. The trend can be viewed in two default workspaces.
- 6** The Tivoli Enterprise Monitoring Server forwards the situations created by Tivoli Performance Agent to the Probe for Tivoli EIF. The probe receives the situations, processes the situation attribute data, maps the data to ObjectServer fields, and then sends alerts to the ObjectServer. The probe uses the rules file configuration to convert the situation data to event data.
- 7** Events that are inserted into the alerts.status table can be viewed, filtered, and sorted in the Active Event List within the Web GUI, or in the event list. Launch-in-context functionality is also enabled from predictive events in the Active Event List, to Tivoli Enterprise Portal. This feature enables you to view details about an event in the relevant Tivoli Enterprise Portal workspace. In Tivoli Enterprise Portal, you can identify the trend to which the predictive event corresponds.

Prerequisites for predictive eventing and predictive analytics

Before you can configure the integrated IBM Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring environment, the following prerequisites must be met.

At a minimum, the following products and versions must be installed, with the required configurations:

- “IBM DB2 V9.1”
- “IBM Tivoli Monitoring V6.2.2 Fix Pack 2”
- “IBM Tivoli Netcool/OMNIBus V7.3.1” on page 423

IBM DB2 V9.1

The IBM DB2 database must be installed and set up with all default users and groups. For more information about installing IBM DB2 V9.1, see the *IBM DB2* information center at: <http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp>

IBM Tivoli Monitoring V6.2.2 Fix Pack 2

IBM Tivoli Monitoring must be set up with one or more remote and hub monitoring servers, Tivoli Enterprise Portal (server and clients), Tivoli Data Warehouse, and Tivoli Performance Analyzer V6.2.2 Fix Pack 2.

For the IBM Tivoli Monitoring setup, use the default encryption key, `IBMTivoliMonitoringEncryptionKey`. You must specify the following features for installation: TEMA, TEPS, TEMS, TEPD, and ECLIPSE. Select the agent `TIVOLI ENTERPRISE USER EXTENSION`. After installation, the summarization and pruning agent must be configured.

For Tivoli Enterprise Portal, you must specify the host name of the server on which Tivoli Enterprise Portal is to be installed, and specify DB2 as the database type.

For Tivoli Data Warehouse, all the databases must reside in the DB2 database.

For Tivoli Performance Analyzer, all features must be installed. Tivoli Performance Analyzer must use all the DB2 databases that were set up during the installation of IBM Tivoli Monitoring. Use the following JDBC driver for the connection to Tivoli Data Warehouse:

- **Linux** **UNIX** /opt/IBM/sqllib/java/db2jcc.jar; /opt/IBM/sqllib/java/db2jcc_license_cu.jar
- **Windows** C:\Program Files\IBM\sqliib\java\db2jcc.jar; c:\Program Files\IBM\sqliib\java\db2jcc_license_cu.jar

Windows On 64 bit computers, you must use the 32 bit Program Files directories. The data sources created by the default ODBC Data Source Administrator applet available from the Control Panel are not available for 32-bit applications. Therefore, you must use the 32-bit version of the ODBC Data Source Administrator applet from `WINDOWS\SysWOW64\odbcad32.exe`.

After installation of IBM Tivoli Monitoring, verify that the Tivoli Enterprise Portal database and tables were created, and log in to the Tivoli Enterprise Portal desktop. Also, if Eclipse does not start after installation, change the port number. If the Warehouse Proxy is not running after installation, you can start it in the Manage Tivoli Enterprise Monitoring Services GUI. If an error occurs during configuration of the Warehouse Proxy, you can manually re-create the Warehouse Proxy by using the Create Database Wizard.

For further installation and configuration information about IBM Tivoli Monitoring, see the *IBM Tivoli Monitoring* Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.2fp2/welcome.htm.

For further installation and configuration information about Tivoli Performance Analyzer V6.2.2, see the *Tivoli Performance Analyzer V6.2.2* Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?topic=/com.ibm.kpa.ovr.doc/c_ovr_product_overview.html

From your IBM Tivoli Monitoring installation, you must enable event forwarding to the Tivoli Netcool/OMNIBus ObjectServers. For each hub monitoring server from which predictive events should be forwarded, enable the Tivoli Event Integration Facility, and then specify the host name of the computer where the Probe for Tivoli EIF is installed, and the port number on which the probe is listening. For further information, locate the IBM Tivoli Monitoring product version node in the information center. Then expand the subnodes as follows: *Installation and Configuration Guides > Installation and Setup Guide > Integrating event management systems > Setting up event forwarding to Netcool/OMNIBus > Configuring the monitoring server to forward events*.

IBM Tivoli Netcool/OMNIBus V7.3.1

The designated ObjectServer hosts to which predictive events should be forwarded must be set up. Install the Conpack feature on each host.

Set up the client workstations with access to the Tivoli Netcool/OMNIBus desktop tools and the Web GUI.

The Web GUI must be installed and configured on a host computer. The Tivoli server should, by default, contain an installation of the Web GUI Administration Application Program Interface (WAAP) client, which is required for loading the required customizations to support predictive events. For more information about configuring the Web GUI for predictive eventing, see “Enabling predictive eventing in the Web GUI” on page 532.

If you want to enable launch-in-context functionality from a predictive event in the Active Event List, to the Tivoli Enterprise Portal without having to log in, single sign-on must be configured between the Tivoli server and the Tivoli Enterprise Portal Server.

One or more Probes for Tivoli EIF must be installed in your Tivoli Netcool/OMNIbus environment, as described in the README.txt and description.txt files in the probe download package. Each ObjectServer to which you want to forward predictive events must have a Probe for Tivoli EIF associated with it, with an edited probe properties file. Customized rules files for processing predictive events are provided for use with Probe for Tivoli EIF. The `tivoli_eif.rules` file is extended to map generic situation event attributes to ObjectServer fields in the `alerts.status` table. This rules file also contains a commented-out include statement for embedding the `predictive_event.rules` file that is provided with Tivoli Netcool/OMNIbus. Uncomment this include statement. See “Further information” on page 425 for more information about schema update required for situation events. You must also specify the server name and port on which the Probe for Tivoli EIF is running in the Tivoli Enterprise Monitoring Server.

The IBM Tivoli Monitoring event synchronization component must be installed on the host computer of each ObjectServer to which you want to forward IBM Tivoli Monitoring situation events and predictive events. The event synchronization component enables changes in the status of situation events to be sent from Tivoli Netcool/OMNIbus back to the originating monitoring server. During the installation of the component, you are required to enter information about each hub monitoring server with which situation events will be synchronized. When you install the event synchronization component, the Situation Update Forwarder process is installed along with its supporting binary and configuration files. Files that can be used to configure the ObjectServer and Probe for Tivoli EIF are also installed. See “Further information” on page 425 for more information about the event synchronization component.

The database schema for each ObjectServer requires updating to enable its database schema to enable alert data from situation events to be stored. (Note that additional ObjectServer schema changes are required specifically for predictive events.) For each ObjectServer, the `alerts.status` table must be updated with the mapping for Tivoli Enterprise Console to ObjectServer fields. This is done by importing the `tec_db_update.sql` file into the table. See “Further information” on page 425 for more information about updating the mapping.

The IBM Tivoli Monitoring for Tivoli Netcool/OMNIbus Agent must be installed and configured. After installation, you must load the Agent triggers and log files into the ObjectServer from the `itm_os.sql` file. You must also specify the host name of the Tivoli Enterprise Monitoring Server, the location of the ObjectServer log file (typically `$NCHOME\omnibus\log`), and the name of the ObjectServer. You can then start the Agent from the Tivoli Enterprise Monitoring Server. You should then

verify that OMNIBus Agent items are displayed on the Tivoli Enterprise Portal Server desktop workspace and match those in the Tivoli Netcool/OMNIBus event lists.

Further information about setting up an integrated IBM Tivoli Monitoring and IBM Tivoli Monitoring environment can be found at the *IBM Tivoli Monitoring* Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Further information

For more information, see the following websites:

- For further information about installing the event synchronization component, locate the **IBM Tivoli Monitoring** in the left navigation pane of the *IBM Tivoli Monitoring* Information Center. Then expand the subnodes as follows:
Installation and Configuration Guides > Installation and Setup Guide > Integrating event management systems > Setting up event forwarding to Netcool/OMNIBus > Installing the event synchronization component
- For further information about setting up the mapping between Tivoli Enterprise Console and ObjectServer fields, see http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIBus.doc/probes/tivoli_eif/tivoli_eif/wip/concept/tveif_configuringtheobjectserver.html
- For further information about the schema update required for situation events, and how to use the customized `tivoli_eif.rules` file, locate the **IBM Tivoli Monitoring** product version node in the *IBM Tivoli Monitoring* information center. Then expand the subnodes as follows:
Installation and Configuration Guide > Installation and Setup Guides > Integrating event management systems > Setting up event forwarding to Netcool/OMNIBus > Configuring the Netcool/OMNIBus Object Server.
In this referenced location in the information center, only the subtasks titled **Updating the OMNIBus database schema** and **Configuring the EIF probe** are mandatory for predictive events and analytics. The other subtasks are redundant, and are relevant only if you are working with other types of situation events.

Tivoli Netcool/OMNIBus configuration resources for predictive events

Tivoli Netcool/OMNIBus provides a number of resources to enable predictive events. These resources are available as sample files that are located in the `$NCHOME/omnibus/extensions/itmpredictive` directory.

ObjectServer resources

The following Objectserver resources support predictive events:

- A class ID of 89300 is reserved for predictive events.
- In the `alerts.status` table, the following columns are added to store data that is specific to predictive events:

Column name	Data type	Mandatory	Description
TrendDirection	integer	No	Applicable to predictive events that are received from IBM Tivoli Monitoring. Indicates the trend for the prediction. The values are: -1: Falling 0: Constant 1: Rising
PredictionTime	type	Yes	Applicable to predictive events that are received from IBM Tivoli Monitoring. Specifies the time, in days, in which Tivoli Performance Analyzer predicts that the defined thresholds will be violated. You should apply a margin of approximately plus or minus 12 hours to the predictive events that are generated.

- The following conversions map integer values to string values:
 - Conversion for class ID 89300: Predictive Events
 - Conversions for the TrendDirection column: 1 = Rising, 0 = Constant, and -1 = Falling
 - Conversion for the DaysToCriticalThreshold and DaysToWarningThreshold columns: 9999 is converted to an empty string because 9999 represents the absence of a DaysToCriticalThreshold or DaysToWarningThreshold value.
- The triggers new_row_predictive and deduplicate_predictive are supplied. These triggers are assigned to the default_triggers trigger group. The new_row_predictive trigger ensures that when a new predictive event is inserted into the ObjectServer, the correct fields are set and the expiry time for the event is set. The deduplicate_predictive trigger ensures that the correct fields are copied on deduplication and that the expiry time for the event is set.

These resources are added to the ObjectServer when you import the predictive_events_menutools_native_gui.jar package file, which is one of the sample files in the \$NCHOME/omnibus/extensions/itmpredictive directory.

Event visualization resources

The following resources support the display of predictive events in the event list and Active Event List:

- A filter file predictive_event.elf, view file predictive_event.elv, and event list configuration file predictive_event.elc, are provided for filtering and sorting predictive events in the event list. The filter and view are defined as follows:
 - The **Predictions** filter is defined with the following WHERE clause:
where Class = 89300

- The **Predictions** view contains the following default columns, which are displayed as follows, from left to right: Node, TrendDirection, Summary, FirstOccurrence, LastOccurrence, Count, PredictionTime.

The sort priority and sort order of the columns is as follows:

1. Severity in descending order
 2. LastOccurrence in ascending order
 3. PredictionTime in ascending order
- Configuration resources for the Active Event List are in the form of a WAAPI command file called `predictive_events_web_gui.xml`, which creates a filter, view, tools, prompts, and menu options for predictive events, and adds web resources (a `.jsp` file, images, and a stylesheet) in the Web GUI server. Information about the Web GUI Administration Application Program Interface (WAAPI) is available in the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Rules file resources

A customized rules file, which can process predictive events, is provided for the Probe for Tivoli EIF. This rules file is called `predictive_event.rules`.

The `predictive_event.rules` file specifically maps predictive event attributes (from Tivoli Performance Analyzer) to ObjectServer fields. The attribute-to-column mappings between the Tivoli Performance Analyzer attributes that are calculated from the trending analysis, and the `alerts.status` columns are as follows:

Table 97. Rules file mappings for predictive events

Tivoli Performance Analyzer attribute	alerts.status column	Notes on mapping
Direction	TrendDirection	Indicates an upward or a downward prediction trend, or a flat line. New column in <code>alerts.status</code> .
Confidence	ExtendedAttr	A percentage value between 0 and 100 that denotes the level of confidence in the predictive trend. In this range, 0 depicts no confidence and 100 depicts a perfectly-approximated function. Not promoted to column.
Strength	ExtendedAttr	The strength of the trend, based on a correlation between the confidence and the number of samples analyzed. Not promoted to column.
TimeStamp	LastOccurrence	The timestamp indicating when the prediction was calculated.

Table 97. Rules file mappings for predictive events (continued)

Tivoli Performance Analyzer attribute	alerts.status column	Notes on mapping
Num_Of_Samples	ExtendedAttr	The number of samples (or data points) used for establishing the trend. The higher the number of samples, the more accurate the estimated prediction. Not promoted to column.
Node	Agent	The host name of the device which Tivoli Performance Analyzer is running. This is the device that is producing the fault events on which the trend is based.
system_name	Node	The host name on which the predictive metrics originated.
89300	Class	The reserved class ID of 89300, which is allocated to predictive events.
<i>Literal string values</i>	Summary	Tivoli Performance Analyzer provides a set of literal string values that are inserted into the Summary column of the alerts.status table.
<i>All other attributes</i>	ExtendedAttr	Additional extended attributes.
3 (WARNING)	Severity	The severity level, as specified in IBM Tivoli Monitoring. The value can be either 3 or 5.
5 (CRITICAL)	Severity	The severity level, as specified in IBM Tivoli Monitoring. The value can be either 3 or 5.

Related tasks

“Configuring predictive eventing in your integrated environment”

Configuring predictive eventing in your integrated environment

A predictive event is an alert that warns operators that a failure might potentially occur at some point in the future. Predictive events are generated within IBM Tivoli Monitoring and can be forwarded to Tivoli Netcool/OMNIBus for display within the event list or the Active Event List.

Before you begin

At a minimum, you require the following product versions: Tivoli Netcool/OMNIBus V7.3.1 and IBM Tivoli Monitoring V6.2.2. It is assumed that you have installed and configured both of these products, so that they are in an operational state.

To configure predictive eventing in an integrated Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring environment:

1. If your integrated environment contains IBM Tivoli Monitoring V6.2.2, copy the `itm_event.rules` file from the following location into the `$NCHOME/omnibus/probes/arch` directory on each computer where the Probe for Tivoli EIF is installed.

<http://www.ibm.com/support/docview.wss?uid=swg21459938>

2. From Tivoli Netcool/OMNIBus, go to the `$NCHOME/omnibus/extensions/itmpredictive` directory.
3. Use the files in this directory to apply the predictive eventing configuration as follows:

- On each ObjectServer host, import the predictive eventing configuration into the ObjectServer schema by changing to the `$NCHOME/omnibus/bin` directory and then running the following command:

```
nco_confpack -import -server server_name -user user_name -password  
password -package $NCHOME/omnibus/extensions/itmpredictive/  
predictive_events_menutools_native_gui.jar -nowarn
```

In this command, *server_name* is the ObjectServer name, and *user_name* and *password* are your login credentials.

The ObjectServer resources are added. Note that the triggers are enabled by default.

- Copy the customized `predictive_event.rules` file to the `$NCHOME/omnibus/probes/arch` directory on each computer where the Probe for Tivoli EIF is installed. This directory should already contain the customized `tivoli_eif.rules` file. Remove the default read-only permissions from the `predictive_event.rules` file.

Edit the `tivoli_eif.rules` file. Ensure that you uncomment the commented-out `include` statement that embeds the `predictive_event.rules` file, and update the path to the location where this file is stored. Also uncomment the commented-out `include` statement that embeds the `itm_event.rules` file. Re-read the rules file if the probe is currently running.

- For event visualization in the event list, copy the `predictive_event.elc`, `predictive_event.elv`, and `predictive_event.elf` files to a preferred location. Remove the default read-only permissions from these files.

As a system administrator, you can make the `predictive_event.elc` file available to event list operators as an event list configuration that can be loaded into the Event List monitor box window (by using **File > Open**). This configuration consists of a single **Predictions** monitor box with a Predictions filter and a Predictions view.

You can also load the `predictive_event.elf` filter and `predictive_event.elv` view into an existing event list configuration as follows:

- a. From the Event List monitor box window, click **Windows > Configuration** to open the Event List Configuration window.
- b. While viewing the filters that are part of this event list configuration, click **Load** (on UNIX or Linux), or click **Open** (on Windows).
- c. From the resulting window, navigate to the location where you saved the `predictive_event.elf` filter file, select the file, and then click **OK**.
- d. While viewing the views that are part of this event list configuration, click **Load** (on UNIX or Linux), or click **Open** (on Windows).
- e. From the resulting window, navigate to the location where you saved the `predictive_event.elv` view file, select the file, and then click **OK**.

- f. Save the event list configuration.

The filter and view are added as a **Predictions** monitor box, and will also be available for selection within all the event lists in that event list configuration.

- g. Apply the predictive eventing columns to the alerts.status table, and activate the predictive eventing triggers by entering the command for your operating system::

```
– UNIX ./nco_sql -user root -password password -server NCOMS <  
/opt/IBM/tivoli/netcool/omnibus/extensions/itmpredictive/  
predictive_event.sql
```

```
– Windows isql -S NCOMS -U root -P password -i C:\IBM\Tivoli\N  
etcool\omnibus\extensions\itmpredictive\predictive_event.sql
```

Where *NCOMS* is the name of the ObjectServer and *password* is the password of the root user.

- To configure predictive eventing for the Active Event List, see the instructions in “Enabling predictive eventing in the Web GUI” on page 532.

What to do next

You should now be able to monitor predictive events in the event list and Active Event List. If required, you can now complete the configuration steps for predictive analytics.

For information about monitoring predictive events in the event list, see the *IBM Tivoli Netcool/OMNIBus User's Guide*. For information about monitoring predictive events in the Active Event List, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Related tasks

“Configuring single sign-on” on page 530

Related reference

“Tivoli Netcool/OMNIBus configuration resources for predictive events” on page 425

“Importing configurations” on page 320

Configuring linear trending

After you have configured predictive eventing, you can run the setup for linear trending, which enables you to calculate problems with device event rates before the problems occur.

Before you begin

Your environment must fulfil the prerequisites for predictive eventing and predictive analytics. Additionally, you must have configured predictive eventing for your integrated environment, and Tivoli Netcool/OMNIBus must be running. The steps described in the following instructions build on the configuration setup for predictive eventing.

Note: Working knowledge of IBM Tivoli Monitoring is assumed.

To set up linear trending, you must install a domain that is provided, but not installed, with Tivoli Netcool/OMNIBus into IBM Tivoli Monitoring. The domain installs the following features that are required for linear trending:

- Two workspaces for viewing and analyzing the trending data
- Two situations: One that is generated up to seven days before the warning threshold is exceeded, and one that is generated up to seven days before the critical threshold is exceeded, based on the trending line.
- A trend. The trend supports forecast on a 7-day basis, a 30-day basis, and a 90-day basis. Situations are only provided for forecasting on a 7-day basis.

When you install Tivoli Netcool/OMNIBus, the domain is installed by default.

Some of the following instructions are applicable only to a test environment; this is indicated where applicable.

To configure predictive analytics in an integrated Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring environment:

1. In Tivoli Enterprise Portal, configure the collection of historical data in the IBM Tivoli Monitoring Agent for Tivoli Netcool/OMNIBus. You perform this step in the History Collection Configuration window.

- a. You must set the KNO EVENT RATE BY NODE attribute group to be archived hourly and set pruning to a week or more. Then, create new collection settings, in which the collection intervals and Warehouse intervals are as fast as possible. Set the attribute group for the collection settings to be KNO EVENT RATE BY NODE. On the **Distribution** tab, start the collection on the OMNIBUS_SERVER_AGENT group.

In the left pane, an icon is displayed next to the **Tivoli Netcool/OMNIBus** item to indicate that data is being collected. The IBM Tivoli Monitoring for Tivoli Netcool/OMNIBus Agent creates several tables and views in the IBM DB2 database. The tables are as follows:

- KNO_EVENT_RATE_BY_NODE
- KNO_EVENT_RATE_BY_NODE _H

The view is KNO_EVENT_RATE_BY_NODE_HV.

- b. Verify that the tables and views been created in the IBM DB2 database. The tables and views are not created instantly.
- c. Repeat step 1a for the KPA_T_NO8099_LTF and KPA_T_NO8099_LTS attribute groups.

2. In your Tivoli Netcool/OMNIBus installation, change to the extensions/itmpredictive directory and copy the itpa_trending directory into your IBM Tivoli Monitoring installation.

3. In your IBM Tivoli Monitoring installation, change to the itpa_trending directory and execute the following command to start the setup:

- **UNIX** domaintool.sh
- **Windows** domaintool.bat

Note: Ignore the bin directory.

4. Enter the location into which you installed IBM Tivoli Monitoring. For example, on Windows, C:\IBM\ITM.

The ITPA Domain Activation Tool detects the Tivoli Enterprise Portal Server installation, and the Tivoli Performance Analyzer installation.

5. In the next window, accept the **Tivoli Enterprise Portal Server (TEPS)**, option, the **Tivoli Performance Analyzer (TPA)** option, and the **Tivoli Enterprise Monitoring Server (TEMS)** option.

If any of these options are grayed out, the user running the installation does not have permission to modify the relevant database. You must add these permissions in the IBM DB2 database and in Tivoli Enterprise Portal Server, and rerun the **domaintool** command.

6. Select the **omnibus_event_rate** domain.

The domain is installed. After the installation has completed, a success message is displayed.

7. Click **Finish** and manually restart Tivoli Enterprise Portal Server and Tivoli Performance Analyzer.
8. To verify that the domain was installed successfully, log into Tivoli Enterprise Portal Server and proceed as follows:

- To verify that the workspaces were installed: From the left navigation pane, expand the node for the computer on which IBM Tivoli Monitoring is installed. Right click **Performance Analyzer Warehouse Agent** and click **Workspaces**. The following workspaces should be displayed for selection:
 - **Event_Rate Details**: This workspace displays a graph that shows the trending forecast. This is designed to display data on a per node (or device) basis so only select this using the link icon from a trend displayed in the Event_Rate Overview workspace.
 - **Event_Rate Overview**: This workspace displays a list of forecasts, for example, the 7-day forecast. Forecasts for 30 days and 90 days are also provided.

The workspaces will display errors until enough data has been archived by the Summarization and Pruning agent.

- To verify that the situations were installed: Click **Situation Editor**. Under **Performance Analyzer Warehouse Agent**, you should see the following two situations:
 - **Event_Rate_TTCT_1W**: The name stands for “time to critical threshold one week.”
 - **Event_Rate_TTWT_1W**: The name stands for “time to warning threshold one week.”

These situations are generated when the trend is forecast to exceed the defined thresholds (defined as a number of events) within seven days. For information about how to change the thresholds from the defaults, see step 9.

- To verify that the trend was installed: Click **Performance Analyzer Configuration** and then click **Analytics**. In the Performance Analyzer Configuration, you should see a trend called **Event Rate Forecast**.

If the following error message is displayed, it can be safely ignored:

SQL Error \$KPACN008099 AGENTNODE\$ not valid in context.

9. Optional: To change the threshold values at which situations are generated, in the Situation Editor, click **Output**. Change the number of events for **Time to Critical** and **Time to Warning** as required.

The default for the Time to Critical threshold is 10,000 events and the default for the Time to Warning threshold is 8000 events.

10. Optional: On a test environment, simulate linear trending by installing a probe into your Tivoli Netcool/OMNIbus for simulation purposes, or by using your existing simulation environment. Simulate a rising event rate that will meet the warning threshold and critical threshold within a reasonable period for testing purposes.

Tip: By default, the Tivoli Data Warehouse Collection Agent collects data every 15 minutes, and saves the minimum value, maximum value, and average value to the Tivoli Data Warehouse DB2® database every hour. If large volumes of data are generated, the load on the server might be increased. In an environment that already experiences heavy loads, you can reduce the load by reducing the collection interval for the Tivoli Data Warehouse Collection Agent.

11. Wait for the environment to build archived data and create a data trend. Typically, at least 10 hours of data are required to build up sufficient data. For testing purposes, set pruning to occur every two weeks. Note that, in a production environment, if you permit too much archived data to build up, the trending line will flatten. The collection intervals and the Tivoli Data Warehouse intervals must be set to the minimum possible values, that is, to be as fast as possible. The Tivoli Data Warehouse Collection Agent collects data every 15 minutes, and saves the minimum value, maximum value, and average value to the Tivoli Data Warehouse DB2 database every hour. If large volumes of data are generated, the load on the server might be increased. In an environment that already experiences heavy loads, you can reduce the load by reducing the collection interval for the Tivoli Data Warehouse Collection Agent.
12. Verify that the data flow is working correctly. You can verify the data flow as follows:
 - To verify that the correct data is being archived into the DB2 database used by Tivoli Data Warehouse, use the DB2 Control Center. Under WAREHOUS look under tables and KNO_EVENT_RATE_BY_NODE_H for the hourly archive of event rates.
 - To verify that data is being loaded through the Tivoli Performance Agent, in the Performance Analyzer Agent Statistics window, verify that the state of the analytical task is set to Computed.

What to do next

To view the graph for a particular trend, in the Event_Rate Overview workspace, click **Link > Details**. If an error is displayed when you view the graph in the Event_Rate Details workspace, you must make sure that the Tivoli Performance Analyzer data is being archived correctly. In particular, check the KPA_T_NO80099_LTF attribute group and the KPA_T_NO80099_LTS attribute group for the Performance Analyzer Warehouse Agent in Tivoli Enterprise Portal.

If required, you can create situations that are generated up to 30 days or 90 days before the threshold is exceeded.

You can also configure baselining for real-time reporting on device event rates.

Troubleshooting

If the performance of your system is slow, do not set the archiving to run at a fast rate because you can overload the performance of the system, especially as data is built up in the DB2 database.

Related concepts

“Prerequisites for predictive eventing and predictive analytics” on page 422

Related tasks

“Configuring baselining”

“Configuring predictive eventing in your integrated environment” on page 428

Configuring baselining

You can monitor the event rates received from probes by Tivoli Netcool/OMNIbus in real time, by setting up the baselining functionality in your integrated IBM Tivoli Monitoring. You can define upper and lower deviations on the baseline, which, when exceeded, trigger a situation from IBM Tivoli Monitoring. The Probe for Tivoli EIF converts the situation into an event that is received by the ObjectServer.

Before you begin

Make sure you have to hand the IBM Tivoli Enterprise Portal user ID and password and the IBM Tivoli Enterprise Portal host name or IP address. You also need the name of the directory into which Tivoli Netcool/OMNIbus is installed.

You must also have performed the configuration steps described in the following information:

- “Prerequisites for predictive eventing and predictive analytics” on page 422
- “Configuring predictive eventing in your integrated environment” on page 428
- “Configuring linear trending” on page 430

You set up baselining by installing two default situations into IBM Tivoli Monitoring. These situations are generated as follows:

- High_Event_Rate_Baseline is generated when the upper threshold set by the deviation from the corridor of normality is exceeded.
- Low_Event_Rate_Baseline is generated when the lower threshold set by the deviation from the corridor of normality is exceeded.

You install these situations into IBM Tivoli Monitoring by running a script that is provided with Tivoli Netcool/OMNIbus. In this script, you define the upper and lower thresholds. The script also sets up a task (a cron job on UNIX, a scheduled task on Windows) that updates the situations with the average hourly event rates that are measured against the thresholds. These average values are based on the event rates received during the current hour during previous weeks, for example, between 2 p.m. and 3 p.m. on Thursdays. The current event rate is calculated every 15 minutes. The script is run on the Tivoli Netcool/OMNIbus host and can connect to the IBM Tivoli Monitoring host remotely.

To set up baselining:

1. On the Tivoli Netcool/OMNIbus host computer, change to the `$NCHOME\itmpredictive\baseline` directory and run the `init_baseline.sh` script.
2. When prompted by the script, provide the following information:
 - The name of the directory into which Tivoli Netcool/OMNIbus is installed (the default is `$OMNIHOME`)
 - The number of previous weeks required to calculate the average event rate for the current hour (the default is five weeks)

- The level of deviation from the average event rate required to exceed the lower threshold and trigger the `Low_Event_Rate_Baseline` situation. This value is defined as the average event rate minus *numberofdeviations*. The default is 2.0.
 - The level of deviation from the average event rate required to exceed the upper threshold and trigger the `High_Event_Rate_Baseline` situation. This value is defined as the average event rate plus *numberofdeviations*. The default is 2.0.
 - IBM Tivoli Enterprise Portal user ID and password (the default user is `sysadmin`)
 - IBM Tivoli Enterprise Portal host name or IP address (the default is `localhost`)
 -
3. Confirm that the situations and the cron job or scheduled task were added:
 - a. In Tivoli Enterprise Portal, start the Situation Editor and verify that the situations `High_Event_Baseline` and `Low_Event_Baseline` were added to the **Tivoli OMNIBus Server** item.
 - b. Verify the existence of the cron job or scheduled task as follows:
 - **UNIX** Run the command `crontab -l` and check for the following line:
`1 * * * * $OMNIHOME/extensions/itmpredictive/baseline/dynamic_event_rate_baseline.sh`
 - **Windows** In the **Scheduled Tasks** list, check for the task **Dynamic Event Rate Baseline**.
 4. Allow the event data to build up over the number of weeks specified for the calculation of average event rates.

Related concepts

“Prerequisites for predictive eventing and predictive analytics” on page 422

Related tasks

“Configuring linear trending” on page 430

“Configuring predictive eventing in your integrated environment” on page 428

Enabling support for TADDM events

IBM Tivoli Application Dependency Discovery Manager (TADDM) is a configuration management tool that discovers both hardware and software systems in an IT environment. TADDM is a subsystem of the IBM Tivoli Change and Configuration Management Database product.

You can configure TADDM to generate notification events when it discovers a change to a configuration item in your IT environment, and to forward the events to the Probe for Tivoli EIF. The probe can then forward the events to the Tivoli Netcool/OMNIBus ObjectServer, for monitoring in the Active Event List or the desktop event list. Launch-in-context menu tools are provided to enable you to navigate from the event list or Active Event List back into the TADDM GUI in order to retrieve further information about the changes that were discovered.

Some usage scenarios are as follows:

- When your IT application infrastructure changes, you want to receive alerts to keep you informed about configuration changes in your IT application

infrastructure. Such alerts are generated following a TADDM discovery and identify changes that have occurred in the interim period since the previous discovery.

- From Tivoli Netcool/OMNIBus, you want to retrieve the details about the IT application infrastructure that is related to a specific configuration change so that you can know what was modified.
- From Tivoli Netcool/OMNIBus, you want to retrieve the change history of an IT application infrastructure item that is related to a configuration change alert so that you can analyze the stability of your IT application infrastructure.

Note: Working knowledge of TADDM is assumed.

Related concepts

“Integration with other Tivoli products” on page 31

Configuration setup for TADDM events

To monitor TADDM events in Tivoli Netcool/OMNIBus, you require Tivoli Netcool/OMNIBus, the Probe for Tivoli EIF, and TADDM to be installed within an integrated environment.

The following figure shows the required configuration setup for the product components in the integrated environment.

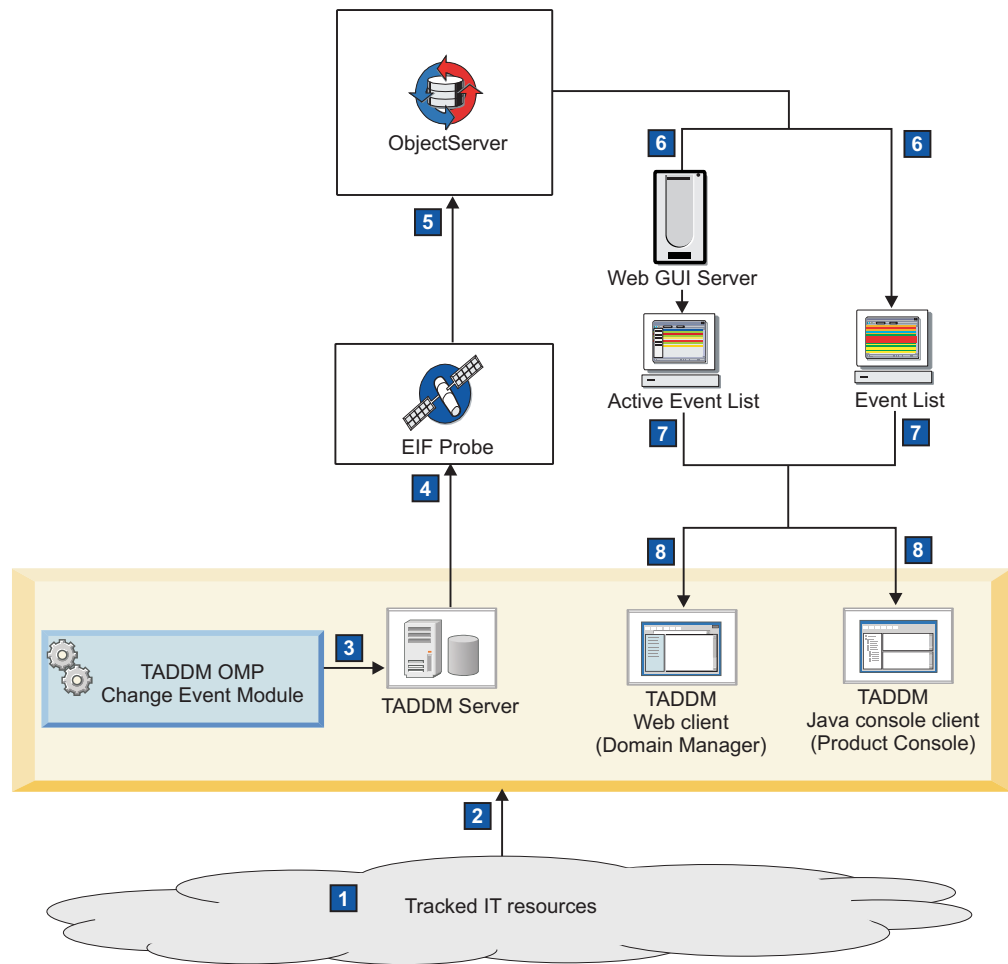


Figure 17. Tivoli Netcool/OMNIBus and TADDM configuration for monitoring TADDM events

The configuration flow is as follows:

- 1** The configuration of an IT resource (or item) is changed by a user.
- 2** The change is discovered by a TADDM sensor during the discovery process.
- 3** After the discovery process is completed, a configured add-on module for TADDM checks for changes to items that you want to track. (This module is called the TADDM OMP Change Event Module.) If changes are detected to one of the tracked items, the Change Event module generates an Event Integration Framework (EIF) event that contains details of the configuration changes.
- 4** The EIF event is forwarded to the Probe for Tivoli EIF.
- 5** The Probe for Tivoli EIF processes the event data, maps the data to ObjectServer fields, and then sends an alert to the ObjectServer.
- 6** The alert is displayed as a TADDM event within the Active Event List or desktop event list.
- 7** From the Active Event List and event list, launch-in-context menu tools can be used to request further details about the configuration item (CI) attributes or its change history.
- 8** The details can be viewed in the TADDM Web or console application.

Related tasks

“Configuring support for TADDM events in your integrated environment”

Related reference

“Tivoli Netcool/OMNIBus configuration files for TADDM events”

Tivoli Netcool/OMNIBus configuration files for TADDM events

When you install Tivoli Netcool/OMNIBus, a number of configuration files enable the monitoring of TADDM events. These configuration files are in the `$NCHOME/omnibus/extensions/taddm` directory.

Details of the configuration files are as follows:

- `taddm.elf`: This filter file can be used to filter TADDM events in the event list. The filter is defined with the following WHERE clause:
`where Class = 87721`
- `taddm_menutools_native_gui.jar` file: This package file creates the following ObjectServer resources:
 - A menu and tools that can be applied to TADDM events in the event list
 - A reserved class ID of 87721 for TADDM events
 - A conversion for class ID 87721: Tivoli Application Dependency Discovery Manager
- `taddm_menutools_web_gui.xml` file: This WAAPI command file creates the menu, tools, and filter that can be applied to TADDM events in the Active Event List. These menu, tools, and filter are added to the Web GUI server. Information about the Web GUI Administration Application Program Interface (WAAPI) is available in the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.
- `tivoli_eif_taddm.rules` file: This customized rules file is provided for the Probe for Tivoli EIF, and must be embedded within the main `tivoli_eif.rules` file. The `tivoli_eif_taddm.rules` file contains the logic to process details of configuration changes that were detected during a TADDM discovery, and to map the data to ObjectServer fields. This rules file also assigns a class ID of 87721 to the events.

Related tasks

“Configuring support for TADDM events in your integrated environment”

Configuring support for TADDM events in your integrated environment

You can configure Tivoli Netcool/OMNIBus, the Probe for Tivoli EIF, and TADDM so that TADDM events can be monitored in the Tivoli Netcool/OMNIBus event list or the Active Event List.

Before you begin

“Additional configuration for TADDM V7.1.2” on page 439

“Configuration steps for supporting TADDM events” on page 441

Before you begin

You require the following product versions:

- Tivoli Netcool/OMNIBus V7.3.1
- TADDM V7.1.2 with the TADDM OMP Change Event Module *or* TADDM V7.2 or later

Note: The TADDM OMP Change Event Module is available as a separate download for TADDM V7.1.2, but is integrated into TADDM V7.2 or later. (See the TADDM documentation for any configuration required.)

It is assumed that you have installed and configured Tivoli Netcool/OMNIbus and TADDM, so that they are in an operational state as follows:

- The designated ObjectServer hosts to which TADDM events should be forwarded have been set up. The Confpack feature must also be installed on each ObjectServer host.
- The Web GUI server is installed and configured on a host computer. The Web GUI server should, by default, contain an installation of the Web GUI Administration Application Program Interface (WAAPI) client, which is required for loading the required customizations to support TADDM events. For further information about the WAAPI client, see the *IBM Tivoli Netcool/OMNIbus Web GUI Administration and User's Guide*.
- Client workstations have been set up with access to the Tivoli Netcool/OMNIbus desktop tools or the Web GUI.
- TADDM has been set up. For information about installing and configuring TADDM, go to the IT Service Management (ITSM) Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v10r1/index.jsp>, open the *Application Dependency Discovery Manager* node in the navigation pane on the left, and locate the information for your required version.
- If you want to enable launch-in-context functionality so that Active Event List users can navigate to TADDM without needing to provide a separate login into TADDM, single sign-on must be configured between the Web GUI server and the TADDM server. To configure the Web GUI for single sign-on, see "Configuring single sign-on" on page 530. To configure TADDM for single sign-on, you must configure TADDM to use WebSphere federated repositories, as documented in the IT Service Management (ITSM) Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v10r1/index.jsp>.

In addition, a Probe for Tivoli EIF must be installed in your Tivoli Netcool/OMNIbus environment, as described in the `README.txt` and `description.txt` files in the probe download package. Edit the probe properties file to include details of the ObjectServer to which you want to forward TADDM events. A customized rules file for processing TADDM events is also provided for use with the probe; see the "Configuration steps for supporting TADDM events" on page 441 procedure for further information.

Related reference

"Importing configurations" on page 320

Additional configuration for TADDM V7.1.2

If you are using TADDM V7.1.2, you must configure TADDM with the TADDM OMP Change Event Module. The TADDM event module is an add-on that allows notifications to be sent to external event handling systems when configuration changes in your environment are discovered by TADDM.

To set up the required configuration:

1. Download the TADDM OMP Change Event Module from the following Web page on the Open Process Automation Library (OPAL) Web site:
<http://www-01.ibm.com/software/brandcatalog/portal/opal/details?catalog.label=1TW10CC1Q>. Note that access to downloads on the OPAL

Web site requires an IBM ID and password, so you will need to go through the IBM Registration process if you have not already done so.

2. Extract the files from the download package to a temporary location, and navigate to the doc subdirectory.
3. Open the index.html file to obtain the instructions for setting up the integration with Tivoli Netcool/OMNIBus.

Tip: The relevant instructions are contained in the section titled *Sending change events to ITM, TEC, and OMNIBus*; if you click any of the links, you will notice that the text is stored in the change_events.html file, which is also within the doc subdirectory. You can read through the text first to familiarize yourself with the contents, but only the text in the *Configuring TADDM* section is relevant.

4. From the change_events.html file, follow all the instructions in the *Configuring TADDM* section to add the event module files to your TADDM server, and to configure the server.

In particular, note that you must update the \$MODULE_PATH/taddmomp/properties/EventConfig.xml file as follows:

- In the Event Listeners section, use the <listener> element to specify details of resources to be tracked for changes.
- In the Event Recipients section, use the <recipient> element to specify details about the system to which event data should be forwarded and the configuration to be applied. The <recipient name=> value in this section must be identical to the <alert recipient=> value in the Event Listeners section; for example, omnibus.

The <address> and <port> elements can typically be used to specify the connection details for the Probe for Tivoli EIF host computer, but are redundant in this case, and can be deleted. Instead, ensure that you use the <config> element to specify the path to the configuration file that was extracted into \$MODULE_PATH/taddmomp/properties/omnibus.eif.properties. You must use this file to specify the connection details for the Probe for Tivoli EIF and other EIF configuration parameters.

Also update the default \$MODULE_PATH/taddmomp/properties/omnibus.eif.properties file (which is referenced in the EventConfig.xml file) as follows:

- At the top of the omnibus.eif.properties file, specify the fully-qualified name of the Probe for Tivoli EIF host computer as the ServerLocation value, and specify the port on which the probe listens as the ServerPort value. Also specify appropriate directory locations for the buffer, trace file, and log file.
- Complete the remainder of the omnibus.eif.properties file as follows. (Note that some of the default comments have been modified.)

```
...
# The event class
TADDMEventClass=TADDM

# EIF Slot definitions for the event

# Supported substitutions for event data in TADDM EIF Adapter. These can be
# used to compose the value of the event slots. The TEC slot name must be preceded
# with TADDMEvent_Slot_
#
# The values slots correspond to the fields that are processed in the
# tivoli_eif_taddm.rules file.
#
TADDMEvent_Slot_object_name=$TADDM_OBJECT_NAME
TADDMEvent_Slot_change_type=$TADDM_CHANGE_TYPE
TADDMEvent_Slot_change_time=$TADDM_CHANGE_TIME
TADDMEvent_Slot_class_name=$TADDM_CLASS_NAME
```

```
TADDMEvent_Slot_attribute_name=$TADDM_ATTRIBUTE_NAME
TADDMEvent_Slot_old_value=$TADDM_OLD_VALUE
TADDMEvent_Slot_new_value=$TADDM_NEW_VALUE
TADDMEvent_Slot_host=$TADDM_HOST
TADDMEvent_Slot_port=$TADDM_PORT
TADDMEvent_Slot_guid=$TADDM_GUID
```

```
# source must be defined to identify the TADDM events in the probe
TADDMEvent_Slot_source=TADDM
```

Configuration steps for supporting TADDM events

To enable TADDM events to be monitored in Tivoli Netcool/OMNIBus:

1. From the Tivoli Netcool/OMNIBus ObjectServer host, go to the \$NCHOME/omnibus/extensions/taddm directory.
2. Copy the sample `tivoli_eif_taddm.rules` file to the \$NCHOME/omnibus/probes/*arch* directory on the computer where the Probe for Tivoli EIF is installed, or to another preferred location.
3. Optional: Edit the `tivoli_eif_taddm.rules` file to set an expiry period for the TADDM events in the ObjectServer. These events are otherwise retained in the ObjectServer because there are no resolution events for the TADDM events. Proceed as follows:

- a. Remove the default read-only permissions from the `tivoli_eif_taddm.rules` file.

- b. Edit the file as follows:

Locate the following commented-out line at the end of the rules file:

```
# @ExpireTime = 7 * 24 * 60 * 60
```

Uncomment this line and then specify an expiry period in seconds for the TADDM events; the example configuration sets the expiry period to one week (that is, 60 seconds * 60 minutes * 24 hours * 7 days).

- c. Save and close the file.

4. Edit the main `tivoli_eif.rules` file for the Probe for Tivoli EIF. Within the `switch($source)` section of the file:

- a. Uncomment the line that contains an `include` statement for the `tivoli_eif_taddm.rules` file. Include the directory path to this file, if necessary.

```
include "tivoli_eif_taddm.rules"
```

- b. Comment out the following line, or leave it uncommented, depending on whether you have set up your system to receive events from Tivoli Enterprise Console.

```
include "tivoli_eif_default.rules"
```

If the line is uncommented, note that you will have to update the ObjectServer database schema by applying the `tec_db_update.sql` import file, which is included with the probe. For further information, see the publication for the Probe for Tivoli EIF. You can access this publication as follows from the IBM Tivoli Network Management Information Center (<http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>):

- 1) Expand the *IBM Tivoli Netcool/OMNIBus* node in the navigation pane on the left.
- 2) Expand the *Tivoli Netcool/OMNIBus probes and TSMs* node.
- 3) Go to the *IBM* node.

5. Edit the \$NCHOME/omnibus/probes/*arch*/`tivoli_eif.props` file to update the value of the **Inactivity** property. By default, the probe terminates if the port on which it is listening is inactive for 10 minutes. Event generation might be

sporadic depending on the TADDM discovery schedules, so configure the probe to run continuously when listening for TADDM events. To configure this setting, set the **Inactivity** property to 0 (zero).

6. Run (or restart) the probe.
7. Add the reserved class, and the menu and tools for TADDM events to the ObjectServer to which the events are sent, and to which the event lists connect:
 - a. From the ObjectServer host, change to the \$NCHOME/omnibus/bin directory.
 - b. Enter the following command:

```
nco_confpack -import -server server_name -user user_name -password password -package $NCHOME/omnibus/extensions/taddm/taddm_menutools_native_gui.jar -nowarn
```

In this command, *server_name* is the ObjectServer name, and *user_name* and *password* are your login credentials.
When the import is complete, a TADDM submenu is available within the **Alerts** menu in the event list.
8. To add the TADDM filter to your event lists, copy the \$NCHOME/omnibus/extensions/taddm/taddm.elf file to a preferred location. You can load the taddm.elf filter into an existing event list configuration as follows:
 - a. From the Event List monitor box window, click **Windows > Configuration** to open the Event List Configuration window.
 - b. While viewing the filters that are part of this event list configuration, click **Load** (on UNIX or Linux), or click **Open** (on Windows).
 - c. From the resulting window, navigate to the location where you saved the taddm.elf filter file, select the file, and then click **OK**.
 - d. Save the event list configuration.The filter is added with the name **TADDM**, and will be available for selection within all the event lists in that event list configuration.
9. To add the menu, tools, and a filter for TADDM events to the Web GUI server, see the instructions in “Enabling support for TADDM events in the Web GUI” on page 534.
10. On all UNIX and Linux computers that run the desktop event list, ensure that the OMNIBROWSER environment variable is set. The OMNIBROWSER setting is required for launching from a TADDM event in the event list to the TADDM Web client. You can set the OMNIBROWSER environment variable to specify the location and file name of the default Web browser as follows:
 - Add the following line to the \$HOME/.login file for a csh user:

```
setenv OMNIBROWSER browser_executable_path
```
 - Add the following line to the \$HOME/.profile file for a ksh or sh user:

```
OMNIBROWSER=browser_executable_path;export OMNIBROWSER
```
11. On all computers that run the desktop event list, ensure that a Java Runtime Environment (JRE) is installed and that the directory location of the Java Web Start utility (**javaws**) is included in the PATH environment variable. The menu tools that launch the TADDM Java console require this utility to be in the PATH environment in which the event list is launched. The **javaws** utility is typically found in the /bin directory of the Java Runtime Environment.
12. From TADDM, configure a discovery schedule for the configuration items to be tracked. When the discovery process finishes, details of detected changes are forwarded to the Probe for Tivoli EIF, which maps the data to ObjectServer fields, and inserts the data into the ObjectServer as events.

Further information about setting up a discovery in TADDM is available in the TADDM Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v10r1/index.jsp>.

What to do next

You should now be able to monitor TADDM events in the event list and Active Event List.

For information about monitoring TADDM events in the event list, see the *IBM Tivoli Netcool/OMNIBus User's Guide*. For information about monitoring TADDM events in the Active Event List, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Managing virtualized environments

As part of an integrated solution with IBM Tivoli Monitoring, you can configure Tivoli Netcool/OMNIBus to perform event management for a virtual environment.

The IBM Tivoli Monitoring system can be configured to use the Probe for Tivoli EIF to forward situation (fault) events that occur within a virtual environment, to the ObjectServer. The resulting alerts can then be monitored in the Active Event List or the desktop event list.

In this configuration, an IBM Tivoli Monitoring for Virtual Servers agent is required to provide the situation events, which can help you identify and resolve virtual server availability and performance issues.

The following hypervisors are supported by IBM Tivoli Monitoring VI Agents:

- VMware ESX
- Citrix
- Microsoft Virtual Server
- Microsoft Hyper-V
- System P (AIX Premium, CEC Base, HMC Base, and VIOS Premium)
- zVM
- Linux

Additionally, the IBM Tivoli Monitoring UNIX OS Agent, can capture Solaris Zone data.

Usage scenarios for managing a virtual environment include:

- Fault event correlation. Fault events that relate to the same original problem can be produced by the hypervisor, the physical computer, or the virtual machine. You require these events to be correlated so that only the root cause events are displayed in the event list or the Active Event List.
- Reduction of severity after virtual machine migration. The severity of hardware-related faults can be automatically lowered when the virtual machine is moved to a new physical host computer.

Note: Working knowledge of IBM Tivoli Monitoring and the configuration of virtual environments is assumed.

Related concepts

“Integration with other Tivoli products” on page 31

Configuration setup for a virtual environment

You can set up a virtualization fault event management system by using Tivoli Netcool/OMNIbus, the Probe for Tivoli EIF, IBM Tivoli Monitoring, and an IBM Tivoli Monitoring for Virtual Servers agent in an integrated environment.

The different components of this system can be distributed across several host computers or can all run on a single powerful host. The components can also run on virtual computers within the cluster being managed, but this is not recommended best practice for a production environment.

The following figure depicts a basic configuration setup for the components in the integrated environment where VMware virtualization software is used to create a virtual environment.

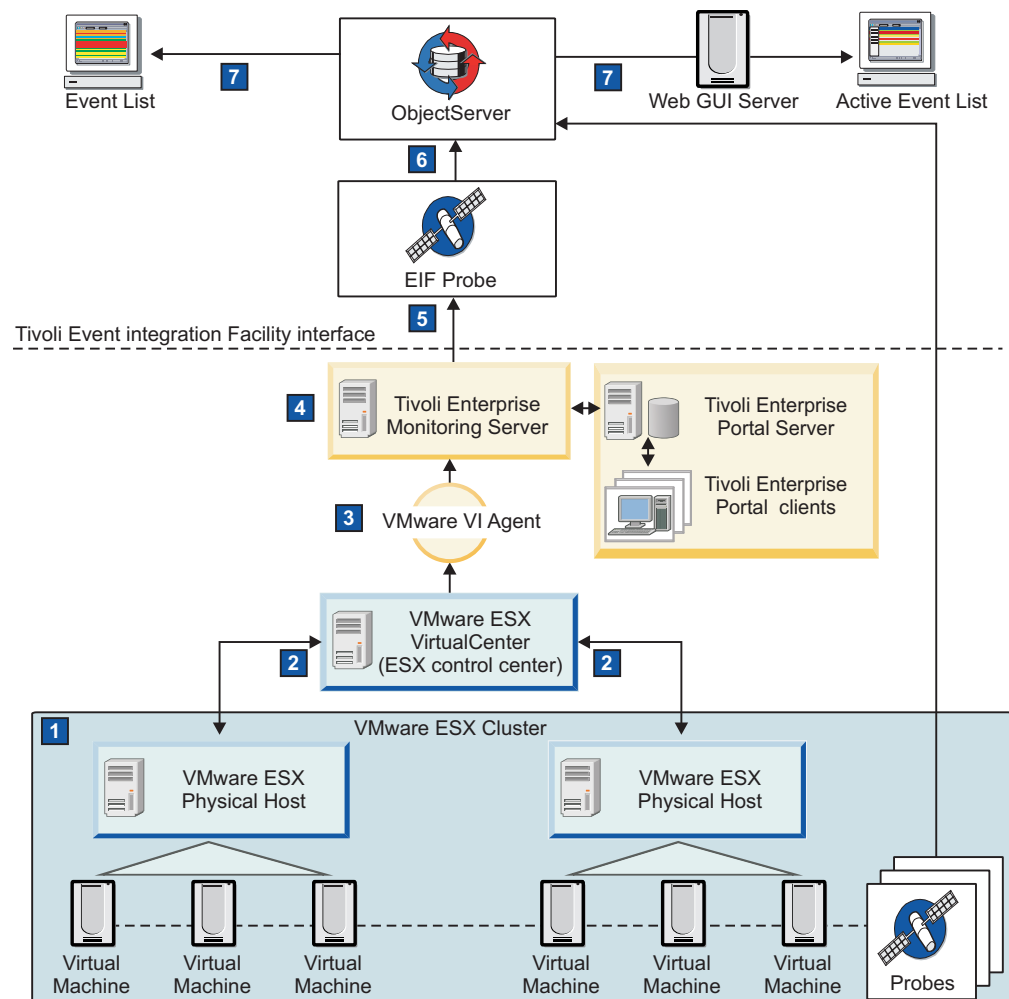


Figure 18. Tivoli Netcool/OMNIbus and IBM Tivoli Monitoring configuration for managing a virtual environment

The configuration flow is as follows:

- 1 A VMware ESX cluster is configured with two or more physical server

hosts. A VMware ESX hypervisor is installed on each physical server and is used to partition the servers into multiple virtual machines that share the hardware resources.

The virtual machines also all run probes, which are configured to acquire event data and to forward the data directly to the ObjectServer, as alerts.

- 2** The VMware ESX cluster is managed from a single, central ESX control center. The VMware VirtualCenter application is used to manage and monitor the VMware servers and virtual machines, and to migrate the virtual machines between servers, as required. (The ESX control centre can be run on a virtual machine on the cluster.)
- 3** The VMware VI Agent monitors the ESX control centre from a remote host. The agent collects monitoring information for memory, CPU, system, disk, and network usage for the VMware ESX servers and the virtual machines. The agent also monitors events and alarms related to faults on the VMware ESX servers and virtual machines.
- 4** The Tivoli Enterprise Monitoring Server acts as a collection and control point for situation events received from the VMware VI Agent. One or more remote and hub monitoring servers could be set up based on your requirements.

A Tivoli Enterprise Portal Server provides the presentation layer for the data collected. The portal server retrieves data from the monitoring server in response to user actions from one or more Tivoli Enterprise Portal clients, and sends the data to the portal clients for presentation, analysis, and manipulation.

- 5** The monitoring server can be configured to forward the situation events to Tivoli Netcool/OMNIBus ObjectServers. The monitoring server uses the Tivoli Event Integration Facility (EIF) interface to forward the situation events to an EIF receiver, which, in this case, is the Probe for Tivoli EIF.
- 6** The Probe for Tivoli EIF receives the situation events, processes the event data, maps the data to ObjectServer fields, and then sends alerts to the ObjectServer. Modifications are required to the probe rules file to map the event data to ObjectServer fields.

The ObjectServer also requires some configuration to process and store the alerts.

- 7** The situation events that are inserted into the alerts.status table can be viewed in the Active Event List within the Web GUI, or in the event list.

Related tasks

“Configuring event management of a virtual environment” on page 447

Related reference

“Tivoli Netcool/OMNIBus configuration files for managing virtualization” on page 446

Tivoli Netcool/OMNIBus configuration files for managing virtualization

When you install Tivoli Netcool/OMNIBus, a number of configuration files are provided for the event management of virtual environments. These resources are available as sample files that are located in the \$NCHOME/omnibus/extensions/itmvirtualization directory.

Details of the configuration files are as follows:

- `virtualization_automations.sql` file: This file creates the following Objectserver resources:
 - The `custom.vmstatus` table. This table is used to store the details about the status of the virtual machines in the virtual environment. The status of the virtual machines is kept up to date by the situation events from the VMware VI Agent. Information in this table can be duplicated because different situations can provide the same information. The `custom.vmstatus` table contains the following columns:

Column name	Data type	Description
VMHostName	varchar(64)	The host name of the virtual machine. Primary key.
HyperHostName	varchar(64)	The host name of the physical server on which the virtual machine is configured.
VMStatus	int	The status of the virtual machine. The values are: <ul style="list-style-type: none">• 0: offline. Indicates that the virtual machine is powered off, in a stuck state, or in a suspended state.• 1: active
StateChange	time	The last time that the entry was modified.

- The triggers `vms_new_row`, `vms_deduplication`, `vms_state_change`, `vms_remove_old_enties`, and `vm_correlate`. These triggers perform error event correlation and resolution based on the virtual machine host name and hypervisor host name. The triggers are assigned to the `default_triggers` and `vm_triggers` trigger groups.
- `remove_virtualization_automations.sql` file: This file removes the virtualization table and automations from the ObjectServer schema, if required.
- `tivoli_eif_virtualization_pt1.rules` and `tivoli_eif_virtualization_pt2.rules` files: These customized rules files are provided for the Probe for Tivoli EIF, and can be used by uncommenting the appropriate lines in the standard rules file (`tivoli_eif.rules`), which is distributed with the probe. The `tivoli_eif_virtualization_pt1.rules` and `tivoli_eif_virtualization_pt2.rules` files contain the logic to process the situation events for errors and resolutions that originate from an IBM Tivoli Monitoring hypervisor agent. These files also map the situation data to ObjectServer fields in the `alerts.status` table, and additionally insert data into the `custom.vmstatus` table.

Summary of the sample configuration provided

The `virtualization_automations.sql` file focuses on correlating events between the virtual machines and the hypervisor, and on processing events that are associated with migrated virtual machines. This sample configuration was written for the VMware VI Agent, but can be adapted for other VI Agents.

Hardware faults can be collected by probes running on the virtual machines, or by the VMware VI Agent. After the events are inserted into the `alerts.status` table, they are correlated by using a temporal trigger that runs every 20 seconds. If a hypervisor situation event is correlated with the same type of situation event from a virtual machine running on it, these two events are modified in the following ways:

- The hypervisor event is marked as a root cause by setting the value of the `NmosCauseType` field to 1. The severity is increased to 5 because the root cause event is causing other faults on the virtual machines and needs to be resolved quickly.
- The virtual machine events are marked as symptoms by setting the value of the `NmosCauseType` field to 2. The severity is lowered to 2 because the symptom is fixed when the root cause is fixed. Finally, the `LocalRootObj` field is set to be the Root Cause events identifier. This means that if IBM Tivoli Network Manager IP Edition extensions to the Web GUI have been installed, click-through from symptom to root cause events is possible.

If a virtual machine is migrated, any associated events with an `AlertGroup` value of `Memory Allocation Status`, `CPU Status`, or `Network Link Status` are reduced in severity to 2 to indicate that they are no longer a significant problem. This is because a virtual machine migration fixes these faults in due course.

Note: Review the contents of the `virtualization_automations.sql` file to understand how the automations work, and to determine whether the sample configuration meets your requirements, or whether additional configuration is required to accommodate other types of hardware faults. You can copy the sample file provided, remove its default read-only permissions, and then edit your copy of the file as required.

Related tasks

“Configuring event management of a virtual environment”

Configuring event management of a virtual environment

IBM Tivoli Monitoring situation events that are generated for a virtual environment can be managed within Tivoli Netcool/OMNIBus.

Before you begin

Note: This procedure provides an end-to-end sample configuration for the VMware virtual environment. The documented configuration steps relate specifically to the use of VMWare ESX and the VMware VI Agent. However, you can modify the configuration steps to use any of the supported hypervisors and their associated IBM Tivoli Monitoring for Virtual Servers agent.

At a minimum, you require the following product versions: Tivoli Netcool/OMNIBus V7.3.1, IBM Tivoli Monitoring V6.2.2, and VMware ESX V3.5.

It is assumed that you have installed and configured Tivoli Netcool/OMNIbus, so that it is in an operational state as follows:

- The designated ObjectServer hosts to which virtualization events should be forwarded have been set up.
- Client workstations have been set up with access to the Tivoli Netcool/OMNIbus desktop tools and the Web GUI.

In addition, one or more Probes for Tivoli EIF must be installed in your Tivoli Netcool/OMNIbus environment, as described in the README.txt and description.txt files in the probe download package. Each ObjectServer to which you want to forward situation events must have a Probe for Tivoli EIF associated with it. A customized rules file for processing the situation events is provided for use with the probes; see the procedure that follows for further information.

To configure event management of your virtual environment:

1. If your integrated environment contains IBM Tivoli Monitoring V6.2.2, copy the itm_event.rules file from the following location into the \$NCHOME/omnibus/probes/arch directory on each computer where the Probe for Tivoli EIF is installed.
<http://www.ibm.com/support/docview.wss?uid=swg21459938>
2. Set up your virtual IT infrastructure with the virtual machines, VMware ESX Servers, and VMware VirtualCenter:
 - a. Install and configure the VMWare ESX cluster.
 - b. Install the VMware VirtualCentre application on the designated host from which the cluster will be centrally managed.

Note:

- At this stage, virtual machines should not be running on the cluster. The VMware VI Agent, ObjectServer, and Probe for Tivoli EIF must be configured and running before any virtual machines are started. This is to ensure that the virtualization status table (custom.vmstatus) in the ObjectServer is populated correctly. If virtual machines are already running on the cluster, either suspend them for the duration of the configuration setup, or migrate them to another host if uninterrupted service is required.
 - When setting up virtual machines, the name that is used to identify a virtual machine in the ESX control center must match the network host name defined for the virtual machine.
3. Ensure that IBM DB2 is installed as the prerequisite RDBMS for IBM Tivoli Monitoring. Then install IBM Tivoli Monitoring. Set up one or more remote and hub monitoring servers, and Tivoli Enterprise Portal (server and clients). For installation and configuration information, go to the *IBM Tivoli Monitoring Information Center* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>. Then locate the *Installation and Setup Guide* for your IBM Tivoli Monitoring product version.
 4. From your IBM Tivoli Monitoring installation, enable event forwarding to the Tivoli Netcool/OMNIbus ObjectServer.

On the monitoring server from which situation events should be forwarded, enable the Tivoli Event Integration Facility, and then specify the host name of the computer where the Probe for Tivoli EIF is installed, and the port number on which the probe is listening. For further information, go to the *IBM Tivoli Monitoring Information Center* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>. For your supported IBM Tivoli Monitoring product version, expand the subnodes as follows: *Installation and Configuration Guides* >

Installation and Setup Guide > Integrating event management systems > Setting up event forwarding to Netcool/OMNIBus > Configuring the monitoring server [to forward events].

5. Install the IBM Tivoli Monitoring for Virtual Servers: VMware VI Agent, which provides the capability to monitor the ESX cluster by using the VMware VirtualCenter application, and to perform basic actions with VMware VirtualCenter. During the installation, ensure that you choose the Monitoring Agent for VMware VI Agent template, which is designed to connect to VMware VirtualCenter. Do not choose the Monitoring Agent for VMware ESX template.

For installation and configuration information, see the *IBM Tivoli Monitoring for Virtual Servers: VMware VI Agent User's Guide* in the IBM Tivoli Monitoring for Virtual Servers Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

6. Enable SSL communication between the VMware VI Agent and the VMware VirtualCenter data source by adding the signer certificate of the VMware VirtualCenter data source to the key database for the VMware VI Agent:
 - a. From the VMware VirtualCentre host, locate the default VMware certificate file named `rui.crt`. For example, on Windows, the default location is `C:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter\SSL\rui.crt`.
 - b. Copy this file to a temporary location on the VMware VI Agent host computer.
 - c. From the VMware VI Agent host computer, run the `gsk7capicmd` command to add the signer certificate to the key database for the agent.

Note: Steps 6a to 6c apply only to the IBM Tivoli Monitoring V6.2.1 VI Agent. For information about how to enable SSL communication for the IBM Tivoli Monitoring V6.2.1 VI Agent, see the *Composite Application Manager for Applications* information center at http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.tivoli.itmvs.doc_6.2.2/vmware622_user.htm.

You can now start the monitoring agent.

For further information, see the *Installing and configuring the monitoring agent* section of the *IBM Tivoli Monitoring for Virtual Servers: VMware VI Agent User's Guide* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

7. From the Manage Tivoli Monitoring Services window in your IBM Tivoli Monitoring installation, set up the configuration for the agent by creating a Monitoring Agent for VMware VI instance and defining the data sources to monitor. Specify a sample interval of 1 minute, ensure that the SSL connection setting is Yes, and also ensure that the VirtualCentre user ID and password are specified.

For further information, see the *Installing and configuring the monitoring agent* section of the *IBM Tivoli Monitoring for Virtual Servers: VMware VI Agent User's Guide* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

8. Start the Tivoli Enterprise Portal and check whether situation events are received from the VMware VirtualCentre host after a one-minute time period. In the Situation Event Console panel, the relevant situation names are prefixed with `KVM_`.
9. From the Tivoli Enterprise Portal, create two additional situations, which are required for the Tivoli Netcool/OMNIBus virtualization configuration:
 - a. Open the Situation Editor.
 - b. From the Situation tree, expand the **VMware VI** node.

- c. Right-click the KVM_VM_Powered_Off situation and click **Create Another** from the pop-up menu.
- d. Create a situation called KVM_VM_Down with a formula of `!= 'poweredOn'` and a sampling interval of 1 minute. Also select **Run at startup**.
- e. On the EIF tab, ensure that events are forwarded to an EIF receiver, with an EIF severity of Critical.
- f. Click the **EIF Slot Customization** button, and ensure that **Map all attributes** is selected.
- g. Create another situation called KVM_VM_Up with similar settings to the KVM_VM_Down situation apart from the following exceptions: set the formula to `== 'poweredOn'` and set the EIF severity to Harmless.

For further information about creating situations, see the *IBM Tivoli Monitoring User's Guide* at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

10. From Tivoli Netcool/OMNIBus, ensure that the ObjectServer is running. Then go to the `$NCHOME/omnibus/extensions/itmvirtualization` directory.
11. Copy the `virtualization_automations.sql` file to the `$NCHOME/omnibus/etc` directory, or another preferred location. Apply the virtualization configuration to the ObjectServer schema by running the following command from the SQL interactive interface:

```

UNIX      Linux      $NCHOME/omnibus/bin/nco_sql -user username -password
password -server servername < directory_path/
virtualization_automations.sql

```

```

Windows    "%NCHOME%\omnibus\bin\isql" -U username -P password -S
servername -i directory_path\virtualization_automations.sql

```

In these commands, *username* is a valid user name, *password* is the corresponding password, *servername* is the name of the ObjectServer, and *directory_path* is the fully-qualified directory path to the .sql file.

If the ObjectServer is part of a failover pair, ensure that the `custom.vmmstatus` table (which is added to the schema) is also replicated by the gateway.

12. Copy the customized `tivoli_eif_virtualization_pt1.rules` and `tivoli_eif_virtualization_pt2.rules` files to the `$NCHOME/omnibus/probes/arch` directory on the computer where the Probe for Tivoli EIF is installed.
13. Edit the `tivoli_eif.rules` rules file, which was distributed with the probe, by uncommenting the lines that contain an include statement for the `tivoli_eif_virtualization_pt1.rules` and `tivoli_eif_virtualization_pt2.rules` files. For example:

```
include "tivoli_eif_virtualization_pt1.rules"
```

Also uncomment the commented-out include statement that embeds the `itm_event.rules` file. Re-read the rules file if the probe is currently running.

14. Remove the default read-only permissions from your copy of the `tivoli_eif_virtualization_pt1.rules` and `tivoli_eif_virtualization_pt2.rules` files. Review the contents of this file, and edit or customize it as appropriate. In particular, edit the `registertarget` statements to specify the name of the ObjectServer to which you want to forward situation events.
15. Edit the `$NCHOME/omnibus/probes/arch/nco_p_tivoli_eif.props` file as follows:

- Set the value of the **RulesFile** property to the path and name of the `tivoli_eif_virtualization.rules` file. Ensure that the **RulesFile** property is specified after the **Name** property.
 - Set the value of the **Inactivity** property to 0 (zero). By default, the probe terminates if the port on which it is listening is inactive for 10 minutes. Setting this property to 0 configures the probe to run continuously when listening for situation events.
 - Ensure that the remainder of the probe properties, such as **Server** and **PortNumber**, are appropriately set.
16. Start the Probe for Tivoli EIF.

Tip: Before starting the probe, run `java -version` from a command window on the probe host computer to ensure that the prerequisite Java 1.5 or later is being used.

17. Start the virtual machines.

You should now be able to monitor the situation events from your virtual environment in the event list and Active Event List.

18. Install other probes on the virtual machines in the cluster. For the example configuration provided, high memory usage and high CPU usage faults are correlated. Therefore, whichever probe is used should be able to identify this kind of error. In the probe rules file, set `@AlertGroup` to Memory Allocation Status or CPU Status to allow correlation between these types of error.

Related reference

"Tivoli Netcool/OMNIBus configuration files for managing virtualization" on page 446

Deploying probes remotely

You can deploy probes from a single centralized computer to one or more remote computers by using the remote deployment mechanism provided by IBM Tivoli Monitoring. You can also update the configuration of the deployed probes from the centralized computer, and uninstall the probes when no longer required.

Probes and Tivoli Netcool/OMNIBus can be remotely deployed as non-agent bundles that you generate using the IBM Tivoli Monitoring Agent Builder.

Note: In this information, examples are provided for the documented tasks to show how the IBM Tivoli Monitoring commands and steps might be applied to the remote deployment of probes. These examples are intended for guidance only. Wherever IBM Tivoli Monitoring commands and steps are documented, the IBM Tivoli Monitoring documentation set should always be the first point of reference, and takes precedence over the information shown in the Tivoli Netcool/OMNIBus examples.

Working knowledge of IBM Tivoli Monitoring is assumed for the remote deployment of probes.

Related concepts

"Integration with other Tivoli products" on page 31

Related tasks

"Installing probes and gateways into the Tivoli Netcool/OMNIBus environment (UNIX and Linux)" on page 103

Prerequisites for remote deployment

Take note of the following prerequisites for remote deployment.

You require the following product and component versions to remotely deploy and manage probes:

- Tivoli Netcool/OMNIBus V7.3.1
- IBM Tivoli Monitoring V6.2.2
- IBM Tivoli Monitoring Agent Builder V6.2.2.1, or later
- Probes that have been packaged for use in Tivoli Netcool/OMNIBus V7.3.1

IBM Tivoli Monitoring must be set up with the following components:

- At least one hub Tivoli Enterprise Monitoring Server
A hub monitoring server is required for probe remote deployment because the deployable bundles that you create are stored in, and distributed from, the monitoring server depot.
- The Agent Builder
Deployable bundles are generated using the Agent Builder.

Note: The Agent Builder is supported on Windows, AIX, and Linux operating systems, and can generate bundles for all the supported operating systems for operating system (OS) agents. For the most current information about the supported operating systems, see the *IBM Tivoli Monitoring Agent Builder User's Guide*.

- Tivoli Enterprise Portal (server and clients)
Optionally, these components can be installed to aid with postinstallation configuration and management; for example, viewing deployment status or creating take action commands.

For information about installing and configuring these IBM Tivoli Monitoring components, go to the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Each remote computer to which a probe is deployed requires:

- A Tivoli Netcool/OMNIBus installation
At a minimum, the **Probe Support** feature is required to provide the infrastructure for probes.
- An IBM Tivoli Monitoring operating system (OS) agent
The OS agent provides the required infrastructure for managing the deployment. For information about installing and configuring OS agents, go to <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>. Locate and expand the *IBM Tivoli Monitoring* node in the navigation pane on the left. Then expand the subnodes that are nested within the product version node as follows: *Installation and Configuration Guides > Installation and Setup Guide > Installation and initial configuration of base components and agents > Deploying monitoring agents across your environment*.

Workflow for deploying probes remotely

This workflow assumes that IBM Tivoli Monitoring has been set up and that an OS agent is installed on the remote computer to which you want to deploy the probe.

The workflow for deploying probes to remote computers is as follows:

1. From the IBM Passport Advantage Web site, obtain the Tivoli Netcool/OMNIBus and probe download packages for your required operating systems.
2. Use the Agent Builder to create deployable bundles for each Tivoli Netcool/OMNIBus download package, and add these bundles to the monitoring server depot.
3. Use the Agent Builder to create deployable bundles for each probe download package, and add these bundles to the monitoring server depot.
4. Deploy Tivoli Netcool/OMNIBus and the probe to a remote computer as follows:
 - a. Log in to the monitoring server.
 - b. Issue the command that deploys the Tivoli Netcool/OMNIBus bundles to the remote computer. The bundles are transported from the monitoring server depot to the OS agent depot on the remote computer.
The Tivoli Netcool/OMNIBus installer then runs in silent mode to install Tivoli Netcool/OMNIBus and its configuration files in the NCHOME directory.
 - c. Issue the command that deploys the probe bundles to the same remote computer. The bundles are transported from the monitoring server depot to the OS agent depot on the remote computer.
The probe installer then runs in silent mode to install the probe and its configuration files into the relevant NCHOME subdirectory.

A note about upgrading: If a remote computer currently has a V7.2.1, or earlier Tivoli Netcool/OMNIBus installation, you must manually upgrade to V7.3.1 before deploying probes. As part of the upgrade process, your existing probe configuration files are migrated into the \$NCHOME/omnibus/probes/migrated or %NCHOME%\omnibus\probes\migrated directory in the V7.3.1 location. After deploying probes to the V7.3.1 location, review the migrated files. If you require similar configurations for the newly-deployed probes, you must update the configuration files of these newly-deployed probes.

Overview of deployable bundles

The Agent Builder generates a package bundle and a configuration bundle for each Tivoli Netcool/OMNIBus or probe download package.

The package bundle is automatically created. The configuration files are added as a configuration project that you can review or update before you generate the configuration package. In the configuration project, the directory structure of the files reflects the directory structure into which the files will be installed on a remote computer.

Bundles are stored in separate directories on the monitoring server.

Tivoli Netcool/OMNIBus bundles

A Tivoli Netcool/OMNIBus package bundle contains the Tivoli Netcool/OMNIBus installation image, which is made up of the following resources:

- The basic probe runtime environment
- A Java Runtime Environment (JRE) that is suitable for running Java probes
- Process control for managing probes as Tivoli Netcool/OMNIBus processes
- Gateway support
- The files required to support installation and uninstallation of probes

A default Tivoli Netcool/OMNIBus configuration bundle contains a connections data file (sql.ini or omni.dat) for configuring server communications, and a process agent configuration file (nco_pa.conf) that can be used to configure process control.

Probe bundles

A default probe package bundle contains the probe installation image, including the probe binary and relevant IBM dependency patches. A probe configuration bundle typically contains a probe properties file (.props), a rules file (.rules), and any other probe-specific configuration files.

About the KDY.INSTALLDIR property

The **KDY.INSTALLDIR** property defines the installation location of Tivoli Netcool/OMNIBus on remote computers. You can specify this property when running IBM Tivoli Monitoring **tacmd** commands to deploy bundles.

IBM Tivoli Monitoring tacmd commands used for remote deployment

A subset of the IBM Tivoli Monitoring **tacmd** commands can be used in remote deployment operations.

These commands are shown in the following table. You must fully understand the syntax and use of these commands.

Table 98. IBM Tivoli Monitoring tacmd commands for remote deployment

Command	Description
tacmd addBundles	Add one or more deployment bundles to the local agent deployment depot.
tacmd addgroupmember	Add a group member to the specified group.
tacmd addSystem	Deploy a monitoring agent to a computer in your IBM Tivoli Monitoring environment.
tacmd clearDeployStatus	Remove entries from the table that stores the status of the asynchronous agent deployment operations.
tacmd creategroup	Create a new group on the server.
tacmd createNode	Deploy an OS agent to a remote computer.
tacmd getDeployStatus	Display the status of the asynchronous agent deployment operations.
tacmd listBundles	Display the details of one or more deployment bundles that are available for deployment to the local deployment depot.
tacmd login	Log on to a monitoring server and create a security token used by subsequent commands.

Table 98. IBM Tivoli Monitoring tacmd commands for remote deployment (continued)

Command	Description
tacmd removeSystem	Remove one or more instances of an agent or uninstall an agent from a managed system.
tacmd viewDepot	Display the types of agents you can install from the deployment depot on the server which you are logged on to.

For full details about these commands, see the *IBM Tivoli Monitoring Command Reference* documentation in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Creating deployable bundles with the default configuration

You can use the Agent Builder to create deployable bundles with the default configuration that is supplied for Tivoli Netcool/OMNIBus or probes.

You can deploy this default configuration to remote computers. A default configuration is also useful because you can use the configuration project as a template for generating deployable bundles with different configurations.

To create deployable bundles with the default configuration:

1. Start the Agent Builder as follows:
 - **UNIX** **Linux** From the Agent Builder installation directory, run the **agentbuilder** command.
 - **Windows** Click **Start > All Programs > IBM Tivoli Monitoring > Agent Builder**.
2. From the Workspace Launcher window, specify a workspace folder for storing the project files that are used for generating the bundles. Click **OK**.
3. From the IBM Tivoli Monitoring Agent Builder window, click **File > New > Other**.
4. From the New window, expand the **IBM Tivoli OMNIBus Wizards** node in the tree and click **Package Bundle**. Click **Next** to start the OMNIBus Install Bundle Wizard.
5. From the OMNIBus Install Package page, specify the location of the package that you downloaded from the Passport Advantage Web site. Click **Next**.
6. From the Remote Deploy Bundle Destination page, specify the location where you want to create the package bundle:
 - If there is a monitoring server on the computer where you are running the Agent Builder, click **Install the Remote Deploy bundle into a local TEMS depot**. Ensure that the IBM Tivoli Monitoring installation location is specified in the **Directory** field.
You must provide login information for the monitoring server.
 - To create the bundle in a directory on your computer, click **Generate the Remote Deploy bundle in a local directory**. After you complete this task, you can transfer the bundle directory to a monitoring server system and use the **tacmd addBundles** command to add the bundles to the monitoring server depot.

For details about this command, see the *IBM Tivoli Monitoring Command Reference* documentation in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

7. Click **Finish**. A message confirms that the package bundle has been added to the depot and that a configuration bundle project has been created.
The configuration bundle project contents are then shown in a tree structure in the **Navigator** tab of the IBM Tivoli Monitoring Agent Builder window, and bundle information is shown in the **Remote Deploy Bundle Editor** tab.
8. Expand the Navigator tree to view the names of the default configuration files provided for Tivoli Netcool/OMNIBus or the probe. Assuming you want to generate a bundle with this default configuration, you do not need to make any changes to these files or to the default settings shown in the **Remote Deploy Bundle Editor** tab. This tab shows the following details:

Bundle Identification Information

Bundle identifier

This field shows a unique identifier for the configuration bundle that will be generated. This ID is constructed from the ID of the associated package bundle, with the suffix -cb.

Bundle description

This field shows descriptive text for the bundle.

Version (VVRRMMFFF)

This field shows an auto-generated version number for the bundle, which is based on the version of Tivoli Netcool/OMNIBus or the probe.

Build This value reflects the build ID of the package that was downloaded from the IBM Passport Advantage Web site.

Commands

For the default configuration, no commands need to be specified.

Prerequisite Bundles

The Tivoli Netcool/OMNIBus or probe package bundle is automatically defined as a prerequisite bundle for the configuration bundle. This indicates that Tivoli Netcool/OMNIBus or the probe will be installed before the configuration files.

9. To generate the configuration bundle, perform one of the following actions:
 - From the **Remote Deploy Bundle Editor** tab, click the **generate the final Remote Deploy bundle** hyperlink.
 - From the Navigator tree, right-click the project and then click **IBM Tivoli Monitoring Remote Deploy > Generate Remote Deploy Bundle**.
10. From the Generate Final Remote Deploy Bundle window, specify the location where you want to create the configuration bundle:
 - If there is a monitoring server on the computer where you are running the Agent Builder, click **Install the Remote Deploy bundle into a local TEMS depot**. Ensure that the IBM Tivoli Monitoring installation location is specified in the **Directory** field.
You must provide login information for the monitoring server.
 - To create the bundle in a directory on your computer, click **Generate the Remote Deploy bundle in a local directory**. You can later add the bundle directory to the monitoring server depot as described in step 6 on page 455.
11. Click **Finish**. You receive confirmation that the configuration bundle has been added to the depot.

What to do next

You can view the contents of the depot in one of the following ways:

- Use the **tacmd viewDepot** command.
- From Tivoli Enterprise Portal, use the Deploy Depot Package List workspace to view details about the deployable packages that are available in the monitoring server depot.

For information about the **tacmd viewDepot** command and about accessing the workspace, see the *IBM Tivoli Monitoring Command Reference* and the *IBM Tivoli Monitoring Tivoli Enterprise Portal User's Guide* in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Related concepts

Chapter 4, “Installing, upgrading, and uninstalling (UNIX and Linux),” on page 45
Chapter 5, “Installing, upgrading, and uninstalling (Windows),” on page 111

Related tasks

“Deploying Tivoli Netcool/OMNIBus and probes to remote computers” on page 460

Creating deployable bundles with updated configuration

From the Agent Builder, you can copy any of the configuration projects that are shown in the Navigator tree and then update the configuration files and settings in the copied project.

When you update the configuration, it is good practice to amend the bundle identification information for the project before generating a configuration bundle. Otherwise, your changes will be written to an existing bundle that has the same bundle identification information. Adopt a versioning mechanism by at least amending the version number of an updated project.

Before you begin

Some considerations for updating your configuration are as follows:

- For Tivoli Netcool/OMNIBus bundles, consider which server components (ObjectServers, proxy servers, process agents, and gateways) are available in your environment. Collate the following information, which is needed to enable the components to communicate with one another: server names, host names, port numbers, and SSL details. You can use this information to update the default connections data file (`sql.ini` or `omni.dat`) in your configuration bundle project.
- For Tivoli Netcool/OMNIBus bundles, you can also update the default process agent configuration file (`nco_pa.conf`) to define how the process agent on a remote computer should manage processes, including any processes for probes that are to be remotely deployed to the remote computer.
- For probe bundles, consider how you want to configure the probe, including the ObjectServer to which the probe should connect and the rules file settings for alerts to be forwarded to this ObjectServer. You can use this information to update the default properties and rules files in your configuration bundle project.

To create deployable bundles with updated configuration:

1. From the Agent Builder, open the workspace where your projects are stored.

2. From the Navigator tree of the IBM Tivoli Monitoring Agent Builder window, right-click the project you want to copy and click **Copy**.
3. In the Navigator tree area, right-click and click **Paste**.
4. Complete the Copy Project window as follows:

Project name

Enter a unique project name.

Use default location

Leave this check box selected if you want to save the project in your default workspace folder for project files. Clear this check box to save the project to a different location.

Location

Specify a different location to which you want to save the project. This field is available only if you clear the **Use default location** check box.

5. Click **OK**. The project is shown in the Navigator tree. The ordering of projects is alphabetical by project name.
6. Expand the contents of this new project, which you want to update.
7. Double-click the **.bundle** entry in the tree to open the **Remote Deploy Bundle Editor** tab for that project.
8. Amend the bundle information and bundle dependencies as follows:

Bundle Identification Information

Bundle identifier

Specify a unique and meaningful ID. Retain the -cb suffix as a naming convention.

Note: Bundle IDs must be alphanumeric strings between 3 and 32 characters, including hyphens. The first character in the ID cannot be the letter k or a number.

Bundle description

Edit the description for the bundle.

Version (VRRMMFFF)

Update the version number for the bundle by incrementing the count from the right. For example, if the original number was 022000000, you can increment the number to 022000001. This number is used to determine whether a bundle needs to be deployed to a remote computer.

Build This value reflects the build ID of the package that was downloaded from the IBM Passport Advantage Web site.

Tip: Click **Save** in the toolbar after updating your bundle information.

Commands

Specify commands that you want to run on the remote computer at various stages when deploying the bundle.

To specify a command, click **Add**. Then enter the command that you want to run and select one of the following command types:

- **Pre-Install:** Specifies that the command should run before the bundle is installed on the remote computer.
- **Install:** Installs the bundle. (This command type is for system use and invokes the installer command for installing bundles.)

- **Post-Install:** Specifies that the command should run after the bundle has been installed.
- **Uninstall:** Specifies that the command should run when the bundle is removed. (This command type is for system use and invokes the installer command for uninstalling bundles.)

Click **OK** to add the command to the **Commands** table. You can click **Remove** to delete a selected command from the **Commands** table.

Prerequisite Bundles

Specify any prerequisite bundles that need to be installed on the remote computer before the configuration bundle is installed. If you had previously deployed a Tivoli Netcool/OMNIBus or probe package bundle, you can remove the package bundle as a prerequisite bundle. To remove a bundle, select the bundle in the **Prerequisite Bundles** table and click **Remove**.

To add a prerequisite bundle, click **Add**. Then specify an identifier and version for the bundle. Click **OK** to add the bundle to the **Prerequisite Bundles** table.

9. From the Navigator tree, edit any of the configuration files for Tivoli Netcool/OMNIBus or the probe. You can use the default editor or another preferred text editor on your system, as follows:
 - Double-click the file to open the file in a text editor window.
 - Right-click the file and click **Open** to open the file in a text editor window.
 - Right-click the file and click **Open With** to choose an editor. Depending on the editor chosen, the file opens as a tab in the right pane, or in a separate text editor window.

Save your changes after editing the files.

10. To add other files to the bundle, perform one of the following actions:
 - From the **Remote Deploy Bundle Editor** tab, click the **Add files** hyperlink.
 - From the Navigator tree, right-click the project and then click **IBM Tivoli Monitoring Remote Deploy > Add Files to Bundle**.

From the Import Bundle Files window, you can specify individual files or directories containing files. When you click **Finish**, these files or directories are copied into the project directory.

11. To generate the configuration bundle, perform one of the following actions:
 - From the **Remote Deploy Bundle Editor** tab, click the **generate the final Remote Deploy bundle** hyperlink.
 - From the Navigator tree, right-click the project and then click **IBM Tivoli Monitoring Remote Deploy > Generate Remote Deploy Bundle**.
12. From the Generate Final Remote Deploy Bundle window, specify the location where you want to create the configuration bundle:
 - If there is a monitoring server on the computer where you are running the Agent Builder, click **Install the Remote Deploy bundle into a local TEMS depot**. Ensure that the IBM Tivoli Monitoring installation location is specified in the **Directory** field.

You must provide login information for the monitoring server.

- To create the bundle in a directory on your computer, click **Generate the Remote Deploy bundle in a local directory**. After you complete this task, you can transfer the bundle directory to a monitoring server system and use the **tacmd addBundles** command to add the bundles to the monitoring server depot.

For details about this command, see the *IBM Tivoli Monitoring Command Reference* documentation at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

13. Click **Finish**. You receive confirmation that the configuration bundle has been added to the depot.

What to do next

You can view the contents of the depot in one of the following ways:

- Use the **tacmd viewDepot** command.
- From Tivoli Enterprise Portal, use the Deploy Depot Package List workspace to view details about the deployable packages that are available in the monitoring server depot.

For information about the **tacmd viewDepot** command and about accessing the workspace, see the *IBM Tivoli Monitoring Command Reference* and the *IBM Tivoli Monitoring Tivoli Enterprise Portal User's Guide* in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Deploying Tivoli Netcool/OMNIbus and probes to remote computers

Any remote computer to which you deploy a probe must already have Tivoli Netcool/OMNIbus installed. The Tivoli Netcool/OMNIbus installation can be remotely deployed or can be manually installed.

Before you begin

For remote deployment, you must have deployable bundles available on the depot of the monitoring server from which you want to deploy either Tivoli Netcool/OMNIbus or probes.

An OS agent must also be running on the remote computer to which you want to deploy the bundle. For information about running OS agents, see the *Operating System Agent User's guides* in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Important: Before deploying a bundle (in particular, a Tivoli Netcool/OMNIbus bundle) to a remote computer, verify that there is sufficient disk space to accommodate the installation.

When you issue the command to deploy a bundle, if more than one version of that bundle exists on the monitoring server depot, the latest version is always deployed.

Related concepts

"Disk space requirements" on page 17

Deploying Tivoli Netcool/OMNIBus to a remote computer

You can use the **tacmd addSystem** command to deploy Tivoli Netcool/OMNIBus bundles.

To deploy Tivoli Netcool/OMNIBus:

1. Log in to the monitoring server by using the **tacmd login** command.
2. Issue the **tacmd addSystem** command to the monitoring server to deploy Tivoli Netcool/OMNIBus to the remote computer.

The bundles are transferred from the monitoring server depot to the OS agent depot. The Tivoli Netcool/OMNIBus installer then runs in silent mode and uses the package bundle to install the product in the NCHOME location. The configuration files are then installed into the appropriate NCHOME subdirectories. You can check the deployment status of the bundles.

Example: Using tacmd to deploy Tivoli Netcool/OMNIBus to a remote computer:

Suppose you want to deploy Tivoli Netcool/OMNIBus to a remote UNIX computer on which a UNIX OS agent is running. You want to deploy Tivoli Netcool/OMNIBus into the `/opt/IBM/tivoli/netcool` installation directory on the remote computer.

The package and configuration bundles for Tivoli Netcool/OMNIBus have been created and are in the depot on the monitoring server named `hubserv.london.ibm.com`. When you created the bundles, the package bundle was automatically allocated the bundle identifier `omnibus`, and the configuration bundle was allocated the identifier `omnibus-cb`. The package bundle, which contains the installation image for the product, is also a prerequisite bundle for the configuration bundle, which contains configuration files.

To deploy Tivoli Netcool/OMNIBus to the remote computer:

1. Log in to the monitoring server as user `myname` with password `secret`:

```
tacmd login -s hubserv.london.ibm.com -u myname -p secret
```
2. If necessary, list details about your known managed systems, including the remote computers to which Tivoli Netcool/OMNIBus can be deployed:

```
tacmd listSystems
```

Make a note of the node where the OS agent is installed on the computer to which you want to deploy Tivoli Netcool/OMNIBus. In this example, for the `hubserv.london.ibm.com` computer, the node name includes the product code for the UNIX OS agent, as follows:

```
hubserv.london.ibm.com:UX
```

3. Deploy the Tivoli Netcool/OMNIBus bundles by entering the following command:

```
tacmd addSystem -n hubserv.london.ibm.com:UX -t omnibus-cb -p  
KDY.INSTALLDIR=/opt/IBM/tivoli/netcool
```

Tip: Make a note of the transaction ID for the deployment, which is written to the screen. You can use this ID for monitoring the status of the deployment.

The bundles are added to the OS agent depot and the Tivoli Netcool/OMNIBus installer then runs in silent mode to install the product and its configuration files into the `/opt/IBM/tivoli/netcool` location on the remote computer.

For further information about using the **tacmd login**, **tacmd listSystems**, and **tacmd addSystem** commands, see the *IBM Tivoli Monitoring Command Reference* documentation at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Deploying a probe to a remote computer

You can use the **tacmd addSystem** command to deploy probe bundles to remote computers.

To deploy a probe:

1. Log in to the monitoring server by using the **tacmd login** command.
2. Issue the **tacmd addSystem** command to the monitoring server to deploy the probe to the remote computer. If you have multiple versions of a probe bundle on the monitoring server depot, you can specify which version of the bundle you want to deploy to the OS agent depot. If no version is specified, the latest version is sent.

The bundles are transferred from the monitoring server depot to the OS agent depot. The probe installer then runs in silent mode and uses the package bundle to install the probe in the `$NCHOME/omnibus/probes` or `%NCHOME%\omnibus\probes\win32` directory. The configuration files for the probe are then installed into these subdirectories. You can check the deployment status of the bundles.

Example: Using tacmd to deploy a probe to a remote computer:

Suppose you want to deploy the Simnet probe to a remote UNIX computer on which a UNIX OS agent is running. You want to deploy the probe into an existing Tivoli Netcool/OMNIBus installation directory (`/opt/IBM/tivoli/netcool`).

The package and configuration bundles for the probe have been created and are in the depot on the monitoring server named `hubserv.london.ibm.com`. When you created the bundles, the package bundle was automatically allocated the bundle identifier `nco-p-simnet`, and the configuration bundle was allocated the identifier `nco-p-simnet-cb`. The package bundle, which contains the installation image for the probe, is also a prerequisite bundle for the configuration bundle, which contains the configuration files.

To deploy the Simnet probe to the remote computer:

1. Log in to the monitoring server as user `myname` with password `secret`:

```
tacmd login -s hubserv.london.ibm.com -u myname -p secret
```
2. If necessary, list details about your known managed systems, including the remote computers to which probes can be deployed:

```
tacmd listSystems
```

Make a note of the node where the OS agent is installed on the computer to which you want to deploy the probe. In this example, for the `hubserv.london.ibm.com` computer, the node name includes the product code for the UNIX OS agent, as follows:

```
hubserv.london.ibm.com:UX
```

3. Deploy the probe bundles by entering the following command:

```
tacmd addSystem -n hubserv.london.ibm.com:UX -t nco-p-simnet-cb -p  
KDY.INSTALLDIR=/opt/IBM/tivoli/netcool
```

Tip: Make a note of the transaction ID for the deployment, which is written to the screen. You can use this ID for monitoring the status of the deployment.

The bundles are added to the OS agent depot and the probe installer then runs in silent mode to install the probe and its configuration files into the `/opt/IBM/tivoli/netcool/omnibus/probes` directory on the remote computer.

For further information about using the **tacmd login**, **tacmd listSystems**, and **tacmd addSystem** commands, see the *IBM Tivoli Monitoring Command Reference* documentation at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Deploying a probe to multiple remote computers

To deploy a probe to multiple remote computers, you must create a deploy group that defines the set of remote computers. You must also create a bundle group as a container for the probe bundles. You can then deploy the probe bundle group to the computers in the deploy group.

When creating deploy groups, consider whether logical groupings such as function (for example, test computers), operating system, or geographical location are appropriate for your requirements.

Before you begin

The prerequisites for deploying a single probe to a remote computer apply.

Tip: You can use a similar set of steps to those described here if you want to deploy Tivoli Netcool/OMNIBus to multiple remote computers before deploying a probe to those computers. You will be required to create a bundle group to hold your Tivoli Netcool/OMNIBus package and configuration bundles, and to deploy this bundle group to the remote computers in the deploy group.

To deploy a probe to multiple remote computers:

1. Use the **tacmd login** command to log in to the monitoring server that holds the deployable bundles for the probe.
2. Create the deploy group for the remote computers by using the **tacmd creategroup** command. When creating the deploy group, you can set the **KDY.INSTALLDIR** property so that it applies to all the computers in the group. This property defines the installation location of Tivoli Netcool/OMNIBus and probes on the remote computers. You can specify the **KDY.INSTALLDIR** property at this stage if identical installation directory locations are being used on all or most of the remote computers.
3. Add each of the remote computers as a member of the deploy group by using the **tacmd addgroupmember** command. When adding individual group members, you can optionally apply the **KDY.INSTALLDIR** property setting to a group member to override any **KDY.INSTALLDIR** property setting that is applied to the deploy group.
4. Create the bundle group for the probe by using the **tacmd creategroup** command.
5. Add the probe to the bundle group by using the **tacmd addgroupmember** command.
6. Issue the **tacmd addSystem** command to the monitoring server to deploy the probe to the remote computer.

The probe bundles are transferred from the monitoring server depot to the OS agent depot on each of the remote computers. The probe installer then runs in

silent mode and uses the package bundle to install the probe in the `$NCHOME/omnibus/probes` or `%NCHOME%\omnibus\probes\win32` directory. The configuration files for the probe are then installed into these subdirectories. You can check the deployment status of the bundles.

Example: Using tacmd to deploy a probe to multiple remote computers:

Suppose you want to deploy the Simnet probe to three remote UNIX computers at your London site. You can group the computers within a deploy group called `LondonHosts` with a group type name of `DEPLOY`. Also assume that a UNIX OS agent is running on each of the remote computers, and the probe is to be deployed into an existing Tivoli Netcool/OMNIBus installation directory (`/opt/IBM/tivoli/netcool`) on each computer.

The package and configuration bundles for the probe have been created and are in the depot on the monitoring server named `hubserv.london.ibm.com`. You must additionally create a bundle group and add the probe package and configuration bundles to this group in order to deploy the probe to the three computers. When you created the probe bundles, the package bundle was automatically allocated the bundle identifier `nco-p-simnet`, and the configuration bundle was allocated the identifier `nco-p-simnet-cb`. The package bundle, which contains the installation image for the probe, is also a prerequisite bundle for the configuration bundle, which contains the configuration files.

1. Log in to the monitoring server as user `myname` with password `secret`:
`tacmd login -s hubserv.london.ibm.com -u myname -p secret`
2. Create the deploy group and then add the computers to the deploy group:
`tacmd creategroup -g LondonHosts -t DEPLOY -p KDY.INSTALLDIR=/opt/IBM/tivoli/netcool -d "Probe hosts in London"`
`tacmd addgroupmember -g LondonHosts -m host1.london.ibm.com -t DEPLOY`
`tacmd addgroupmember -g LondonHosts -m host2.london.ibm.com -t DEPLOY`
`tacmd addgroupmember -g LondonHosts -m host3.london.ibm.com -t DEPLOY`

Note: If your Tivoli Netcool/OMNIBus installation on `host2.london.ibm.com` was in a different location (for example, `/space/mysubdir/tivoli/netcool`), you could use the following command when adding the computer to the deploy group:

```
tacmd addgroupmember -g LondonHosts -m host2.london.ibm.com -t DEPLOY -p
KDY.INSTALLDIR=/space/mysubdir/tivoli/netcool
```

3. Create a bundle group and add the probe to the bundle group:
`tacmd creategroup -g SIMNETprobeBundle -t BUNDLE -d "Simnet probe for LondonHosts"`
`tacmd addgroupmember -g SIMNETprobeBundle -m ProbeforSimnet -t BUNDLE -y nco-p-simnet-cb`
4. Deploy the Simnet probe to the three remote computers in the `LondonHosts` deploy group as follows:
`tacmd addSystem -g LondonHosts -b SIMNETprobeBundle`
The bundles are added to the OS agent depot on each of the remote computers and the probe installer then runs in silent mode to install the probe and its configuration files into the `/opt/IBM/tivoli/netcool/omnibus/probes` location on the computers.

For further information about using the **`tacmd login`**, **`tacmd creategroup`**, **`addgroupmember`**, and **`tacmd addSystem`** commands, see the *IBM Tivoli Monitoring*

Command Reference documentation at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Deploying multiple probes to one or more remote computers

To deploy multiple probes to one or more remote computers, you must create a bundle group to act as a container for all of the probe bundles. You must also create a deploy group that defines the single or multiple remote computers to which the probes should be deployed. You can then deploy the bundle group containing the probes to each computer in the deploy group.

Before you begin

The prerequisites for deploying a single probe to a remote computer apply.

To deploy multiple probes to one or more remote computers:

1. Use the **tacmd login** command to log in to the monitoring server that holds the deployable bundles for the probes.
2. Create the deploy group for the single or multiple remote computers by using the **tacmd creategroup** command. When creating the deploy group, you can set the **KDY.INSTALLDIR** property so that it applies to all the computers in the group. This property defines the installation location of Tivoli Netcool/OMNIBus and probes on the remote computers. You can specify the **KDY.INSTALLDIR** property at this stage if identical installation directory locations are being used on all or most of the remote computers.
3. Add each of the remote computers as a member of the deploy group by using the **tacmd addgroupmember** command. When adding individual group members, you can optionally apply the **KDY.INSTALLDIR** property setting to a group member to override any **KDY.INSTALLDIR** property setting that is applied to the deploy group.
4. Create the bundle group for the probes by using the **tacmd creategroup** command.
5. Add each probe to the bundle group by using the **tacmd addgroupmember** command.
6. Issue the **tacmd addSystem** command to the monitoring server to deploy the probes to the single or multiple remote computers.

The bundles for the probes are transferred from the monitoring server depot to the OS agent depot on each of the remote computers. The probe installer then runs in silent mode and uses the package bundles to install each probe (in turn) in the `$NCHOME/omnibus/probes` or `%NCHOME%\omnibus\probes\win32` directory. The configuration files for each probe are installed into these subdirectories after the associated package bundle has been installed. You can check the deployment status of the bundles.

Example: Using tacmd to deploy multiple probes to multiple remote computers:

Suppose you want to deploy two IBM probes (IBM probe A and IBM probe B) to three remote UNIX computers at your London site. You can group the computers within a deploy group called `LondonHosts` with a group type name of `DEPLOY`. Also assume that a UNIX OS agent is running on each of the remote computers, which all have an existing Tivoli Netcool/OMNIBus installation directory (`/opt/IBM/tivoli/netcool`).

The package and configuration bundles for the probes have been created and are in the depot on the monitoring server named `hubserv.london.ibm.com`. You must

additionally create a bundle group called IBMprobeBundles and add the probes to this group. When you created the probe bundles for IBM probe A, the package bundle was automatically allocated the bundle identifier nco-p-probea, and the configuration bundle was allocated the identifier nco-p-probea-cb. Similarly, bundle identifiers nco-p-probeb and nco-p-probeb-cb were generated for the IBM probe B bundles. Each package bundle, which contains the installation image for the probe, is also a prerequisite bundle for the associated configuration bundle, which contains the configuration files for the probe.

1. Log in to the monitoring server as user myname with password secret:
`tacmd login -s hubserv.london.ibm.com -u myname -p secret`
2. Create the deploy group and add the computers to the deploy group:
`tacmd creategroup -g LondonHosts -t DEPLOY -p KDY.INSTALLDIR=/opt/IBM/tivoli/netcool -d "Probe hosts in London"`
`tacmd addgroupmember -g LondonHosts -m host1.london.ibm.com -t DEPLOY`
`tacmd addgroupmember -g LondonHosts -m host2.london.ibm.com -t DEPLOY`
`tacmd addgroupmember -g LondonHosts -m host3.london.ibm.com -t DEPLOY`
3. Create the bundle group and add the probes as members of the bundle group:
`tacmd creategroup -g IBMprobeBundles -t BUNDLE -d "IBM probes"`
`tacmd addgroupmember -g IBMprobeBundles -m IBMprobeA -t BUNDLE -y nco-p-probea-cb`
`tacmd addgroupmember -g IBMprobeBundles -m IBMprobeB -t BUNDLE -y nco-p-probeb-cb`
4. Deploy the IBM probes to the three remote computers in the LondonHosts deploy group as follows:
`tacmd addsystem -g LondonHosts -b IBMprobeBundles`
The bundles are added to the OS agent depot on each of the remote computers and the probe installer then runs in silent mode to install the probe and its configuration files into the /opt/IBM/tivoli/netcool/omnibus/probes location on the computers.

Note: If you wanted to deploy the two IBM probes to a single remote computer, you would create a deploy group and add the computer to this group before creating a bundle group for the probes and deploying the probes. For example, you can create the deploy group as follows:

```
tacmd creategroup -g SingleHost -t DEPLOY -p KDY.INSTALLDIR=/opt/IBM/tivoli/netcool -d "Single probe host"

tacmd addgroupmember -g SingleHost -m lone_host.london.ibm.com -t DEPLOY
```

Monitoring the status of your deployments

When you deploy Tivoli Netcool/OMNIBus or probe bundles, you can view the status of the deployment operation and verify that it completes successfully. Bundles that you deploy are added to a deployment queue on the monitoring server pending deployment.

The status information for deployments typically provides a summary of your pending, in-progress, retrying (after a failure), failed, and successful deployments. When viewing deployment status, you can use the transaction ID that was returned with the **tacmd addSystem** command to identify a specific deployment to be monitored.

To check the deployment status, perform either of the following actions:

- Run the **tacmd getDeployStatus** command. You can run this command without any command-line options to view the status of all deployment operations. For further information, see the *IBM Tivoli Monitoring Command Reference* documentation in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.
- From Tivoli Enterprise Portal, access the Deployment Status Summary By Transaction workspace to view the status. For information about accessing this workspace, see the *IBM Tivoli Monitoring Tivoli Enterprise Portal User's Guide* in the IBM Tivoli Monitoring Information Center.

Additional notes:

- The **tacmd clearDeployStatus** command can be used to clear all the entries, or a filtered list of entries in the status table.
- By default, four deployment retries are attempted before a deployment is assigned a failed status. Some of the causes for failure are:
 - Issues with file permissions
 - Issues with the Deployment Engine component (and associated files) that are used for Tivoli Netcool/OMNIBus and probe installations

These issues cannot be detected before the deployment operation, so failures will be recorded in the deployment status. If a failure occurs, review the Tivoli Netcool/OMNIBus installation log files on the remote computer to help identify the cause of the failure. Also review the IBM Tivoli Monitoring Deployment Engine log files.

Related tasks

“Viewing the installation log files (UNIX and Linux)” on page 60

“Viewing the installation log files (Windows)” on page 124

Running remotely-deployed probes

Various methods are available for running a remotely-deployed probe.

These methods are as follows:

- You can create a take action command to start a process agent that is configured to run the probe as a process on the remote computer.
- You can create take action commands to add the probe as a process while the process agent is running, and to start the process.
- You can use Netcool/OMNIBus Administrator to connect to a running process agent and to set up the probe process.
- You can create a take action command to install the probe as a Windows service, and use a predefined take action command to run the Windows service.

Two examples are used to illustrate how to run probes.

To create and run a take action command to start a process agent that is configured to run the probe as a process:

1. From Tivoli Enterprise Portal, create a new take action command for the relevant OS agent type, with the following settings:
 - **Name** and **Description**: Specify a meaningful name and description.
 - **Type**: Select System Command to enable you to issue a command on the operating system for the selected OS agent type.

- **Command:** Enter the following command to start the process agent on the remote computer:

UNIX	Linux	\$NCHOME/omnibus/bin/nco_pad -name <i>process_agent</i>
Windows		%NCHOME%\omnibus\bin\nco_pad.exe -name <i>process_agent</i>

Where *process_agent* is the name of the process agent, as defined in the omni.dat or sql.ini file.

2. To run the take action command on the remote computer:
 - a. Select the OS agent type in the Navigator.
 - b. Right-click and then click **Take Action > Select**.
 - c. Select the name of the action, and select the remote computer name as a destination.

The process agent is started on the remote computer, and the probe process starts according to any defined dependencies.

To create and run take action commands that run the probe as a Windows service:

1. From Tivoli Enterprise Portal, create a new take action command to install the probe as a service. For the Windows OS item in the Navigator, specify the following settings:

- **Name** and **Description:** Specify a meaningful name and description.
- **Type:** Select System Command to enable you to issue a command for the Windows operating system.
- **Command:** Enter the command that installs the probe as a service:

%NCHOME%\omnibus\bin\probe_name.exe -install

Where *probe_name* is the name of the executable file for the probe.

Tip: You will need to verify the service name of the probe because this name is needed to run the probe remotely.

2. To run the take action command to install the probe service on the remote computer:
 - a. Select the Windows OS item in the Navigator.
 - b. Right-click and then click **Take Action > Select**.
 - c. Select the name of the action, and select the remote computer name as a destination.
3. To run the take action command to run the probe service on the remote computer:
 - a. Select the Windows OS item in the Navigator.
 - b. Right-click and then click **Take Action > Select**.
 - c. Select the predefined action name of Start Service, and enter the service name of the probe.
 - d. Select the remote computer name as a destination.

What to do next

For detailed information about creating and running take action commands, see the *IBM Tivoli Monitoring Tivoli Enterprise Portal User's Guide* in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Using the file transfer utility (nco_cftp) to update files

The **nco_cftp** utility is provided with Tivoli Netcool/OMNIBus to facilitate direct access to files across your Tivoli Netcool/OMNIBus environment. IBM Tivoli Monitoring is a prerequisite.

You can use the **nco_cftp** utility to retrieve files from one computer (the source) and send them to another computer (the target). The transfer utility also enables you to send files back to the computer from which they were retrieved. This file transfer process can be useful for retrieving files that are deployed to remote computers. For example, you can retrieve log files from a remote computer for further investigation.

The **nco_cftp** utility can also be used for transferring files that are external to the Tivoli Netcool/OMNIBus installation directory.

The **nco_cftp** utility and its accompanying files are provided in the `$NCHOME/omnibus/extensions/itmdeploy` directory. These files provide a sample configuration that can be tailored to your requirements.

Before you begin

You can use either of the following configuration scenarios to set up the **nco_cftp** utility for use:

Tivoli Netcool/OMNIBus and the Tivoli Monitoring Enterprise Server are coresident on the same host

UNIX

Linux

Set up the customization as follows:

1. Set the `$CANDLE_CJ_HOME` environment variable to the directory path that leads to the `cj` directory in your IBM Tivoli Monitoring installation.

For example, if the `cj` directory path is `opt/IBM/ITM/platform_code/cj`, set the environment variable to `opt/IBM/ITM/platform_code`.

2. In the Tivoli Netcool/OMNIBus installation directory:
 - a. Copy the `nco_cftp` and `nco_cftp.props` files from the `$NCHOME/omnibus/extensions/itmdeploy/scripts` directory to the `$NCHOME/omnibus/bin` directory.
 - b. Copy the `cftp.jar` file from the `$NCHOME/omnibus/extensions/itmdeploy` directory to the `$NCHOME/omnibus/java/jars` directory.
 - c. Remove the default read-only permissions from the `$NCHOME/omnibus/bin/nco_cftp.props` file.

Windows

In the Tivoli Netcool/OMNIBus installation directory, set up the customization as follows:

1. Copy the `nco_cftp.vbs` and `nco_cftp.props` files from the `%NCHOME%\omnibus\extensions\itmdeploy\scripts` directory to the `%NCHOME%\omnibus\bin` directory.
2. Copy the `cftp.jar` file from the `%NCHOME%\omnibus\extensions\itmdeploy` directory to the `%NCHOME%\omnibus\java\jars` directory.

Tivoli Netcool/OMNIBus and the Tivoli Monitoring Enterprise Server are on different hosts

UNIX

Linux

Set up the customization as follows:

1. On the Tivoli Netcool/OMNIBus host:

- a. Copy the `nco_cftp` and `nco_cftp.props` files from the `$NCHOME/omnibus/extensions/itmdeploy/scripts` directory to the `$NCHOME/omnibus/bin` directory.
 - b. Copy the `cftp.jar` file from the `$NCHOME/omnibus/extensions/itmdeploy` directory to the `$NCHOME/omnibus/java/jars` directory.
 - c. Remove the default read-only permissions from the `$NCHOME/omnibus/bin/nco_cftp.props` file.
2. Copy the following directories in the Tivoli Netcool/OMNIBus host, and paste them into a location on the monitoring server host. The directory structure of the pasted directories on the monitoring server host must mirror the Tivoli Netcool/OMNIBus directory structure.

`$NCHOME/omnibus/bin`
`$NCHOME/omnibus/java/jars`

For example, if the Tivoli Netcool/OMNIBus locations are `/opt/IBM/tivoli/netcool/omnibus/bin` and `/opt/IBM/tivoli/netcool/omnibus/java/jars`, you require this same structure on the monitoring server.
3. On the monitoring server host:
 - a. Set the `$NCHOME` and `$OMNIHOME` environment variables to point to the `netcool` and `omnibus` directory paths respectively.
 For example, set `$NCHOME` to `/opt/IBM/tivoli/netcool` and set `$OMNIHOME` to `/opt/IBM/tivoli/netcool/omnibus`.
 - b. Set the `$CANDLE_CJ_HOME` environment variable to the directory path that leads to the `cj` directory in your IBM Tivoli Monitoring installation.
 For example, if the `cj` directory path is `opt/IBM/ITM/platform_code/cj`, set the environment variable to `opt/IBM/ITM/platform_code`.

Windows Set up the customization as follows:

1. On the Tivoli Netcool/OMNIBus host:
 - a. Copy the `nco_cftp.vbs` and `nco_cftp.props` files from the `%NCHOME%\omnibus\extensions\itmdeploy\scripts` directory to the `%NCHOME%\omnibus\bin` directory.
 - b. Copy the `cftp.jar` file from the `%NCHOME%\omnibus\extensions\itmdeploy` directory to the `%NCHOME%\omnibus\java\jars` directory.
2. Copy the following directories in the Tivoli Netcool/OMNIBus host, and paste them into a location on the monitoring server host. The directory structure of the pasted directories on the monitoring server host must mirror the Tivoli Netcool/OMNIBus directory structure.

`%NCHOME%\omnibus\bin`
`%NCHOME%\omnibus\java\jars`

For example, if the Tivoli Netcool/OMNIBus locations are `C:\IBM\tivoli\netcool\omnibus\bin` and `C:\IBM\tivoli\netcool\omnibus\java\jars`, you require this same structure on the monitoring server.
3. On the monitoring server host, set the `%NCHOME%` and `%OMNIHOME%` environment variables to point to the `netcool` and `omnibus` directory paths respectively.
 For example, set `%NCHOME%` to `C:\IBM\tivoli\netcool` and set `%OMNIHOME%` to `C:\IBM\tivoli\netcool\omnibus`.

You can run the **nco_cftp** utility with a properties file that defines the transfer operation as a GET or PUT action and specifies other settings. The **nco_cftp** utility needs to authenticate against a Tivoli Monitoring Portal Server before attempting to transfer files. The portal server connects to the monitoring server, which then connects to the OS agent that is running on the remote computer.

Note: The maximum size of the files that can be transferred in a single action is 32 MB. To maintain a good level of performance, restrict this size to 10 MB.

A suggested method for use is to maintain a pair of properties files for the GET and PUT actions. You can make copies of the default `nco_cftp.props` file and rename the copies with meaningful names that indicate their purpose:

- In the properties file that defines the GET action, specify the source location from which the files should be retrieved, and specify the target location to which they should be copied.
- In the properties file that defines the PUT action, specify the source location of the files to be transferred back, and specify the target location to which the files should be returned.

You can maintain several pairs of properties files based on your file transfer requirements.

This task acts an example to illustrate how to retrieve, update, and replace files that are installed on a remote computer:

- If Tivoli Netcool/OMNIBus and the monitoring server are coresident on the same host:
 1. From your Tivoli Netcool/OMNIBus installation, make two copies of the `$NCHOME/omnibus/bin/nco_cftp.props` (or `%NCHOME%\omnibus\bin\nco_cftp.props`) file, and rename the copied files. For example:

```
UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp1_get.props
$NCHOME/omnibus/bin/nco_cftp1_put.props
```

```
Windows   %NCHOME%\omnibus\bin\nco_cftp1_get.props
%NCHOME%\omnibus\bin\nco_cftp1_put.props
```
 2. Edit the `nco_cftp1_get.props` file to define the settings for your transfer operation; for example, your authentication credentials, the files to be retrieved for editing, and the source and target locations of the files. In particular, ensure that you set the value of the **transfer.action** property to GET.
 3. Save and close the `nco_cftp1_get.props` file.
 4. To retrieve the files from the specified computer, run the following command from a command prompt:

```
UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp -properties
nco_cftp1_get.props
```

```
Windows   %NCHOME%\omnibus\bin\nco_cftp.vbs -properties
nco_cftp1_get.props
```

Tip: You can optionally run this command by specifying individual command-line options. The command-line options that you specify at the command line will override the equivalent property settings in the `nco_cftp1_get.props` file.

5. Go to the location where the files were retrieved and edit them as required.

6. When you are ready to transfer any of the updated files to the remote computer, edit the `nco_cftp1_put.props` file to define the settings for your transfer operation; for example, your authentication credentials, the files to be transferred, and the source and target locations of the files. In particular, ensure that you set the value of the **transfer.action** property to PUT.
7. Save and close the `nco_cftp1_put.props` file.
8. To transfer updated files back to the remote computer, run the following command from a command prompt:

```

UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp -properties
nco_cftp1_put.props

Windows    %NCHOME%\omnibus\bin\nco_cftp.vbs -properties
nco_cftp1_put.props

```

- If Tivoli Netcool/OMNIbus and the monitoring server are on different hosts:

1. From your monitoring server, make two copies of the `$NCHOME/omnibus/bin/nco_cftp.props` (or `%NCHOME%\omnibus\bin\nco_cftp.props`) file, and rename the copied files. For example:

```

UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp1_get.props
$NCHOME/omnibus/bin/nco_cftp1_put.props

Windows    %NCHOME%\omnibus\bin\nco_cftp1_get.props
%NCHOME%\omnibus\bin\nco_cftp1_put.props

```

2. Edit the `nco_cftp1_get.props` file to define the settings for your transfer operation; for example, your authentication credentials, the files to be retrieved for editing, and the source and target locations of the files. In particular, ensure that you set the value of the **transfer.action** property to GET.
3. Save and close the `nco_cftp1_get.props` file.
4. To retrieve the files from the specified computer, run the following command from a command prompt:

```

UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp -properties
nco_cftp1_get.props

Windows    %NCHOME%\omnibus\bin\nco_cftp.vbs -properties
nco_cftp1_get.props

```

Tip: You can optionally run this command by specifying individual command-line options. The command-line options that you specify at the command line will override the equivalent property settings in the `nco_cftp1_get.props` file.

5. Go to the location where the files were retrieved and edit them as required.
6. When you are ready to transfer any of the updated files to the remote computer, edit the `nco_cftp1_put.props` file to define the settings for your transfer operation; for example, your authentication credentials, the files to be transferred, and the source and target locations of the files. In particular, ensure that you set the value of the **transfer.action** property to PUT.
7. Save and close the `nco_cftp1_put.props` file.
8. To transfer updated files back to the remote computer, run the following command from a command prompt:

```

UNIX      Linux      $NCHOME/omnibus/bin/nco_cftp -properties
nco_cftp1_put.props

```

```
Windows %NCHOME%\omnibus\bin\ncftp.vbs -properties  
ncftp1_put.props
```

Example: Retrieving a log file

Suppose you want to retrieve a log file from a Tivoli Netcool/OMNIBus installation that was deployed to a remote UNIX computer. Assuming your Tivoli Netcool/OMNIBus and IBM Tivoli Monitoring installations have been appropriately configured for the **ncftp** utility, you can retrieve the file as follows:

1. Copy the `ncftp.props` file and rename the copy `ncftp_logfile_get.props`.
2. Update the `ncftp_logfile_get.props` file with the relevant settings, including the log file name, and the source and target locations. Also set the value of the **transfer.action** property to GET.
3. To retrieve the log file, run the following command:
`$NCHOME/omnibus/bin/ncftp -properties ncftp_logfile_get.props`

ncftp properties and command-line options

The **ncftp** utility contains a set of properties and command-line options for transferring files between computers.

The **ncftp** properties file is called `ncftp.props`, and its location is user dependent. In the unedited properties file, all properties are commented out with a number sign (#) at the beginning of the line. Uncomment the properties that you want to set by removing the number sign (#).

The properties and command-line options for **ncftp** are described in the following table. The properties are listed in the order in which they are displayed in the properties file.

Table 99. *ncftp.props* properties and command-line options

Property	Command-line option	Description
teps.server.name <i>string</i>	-server	Specifies the host name of the Tivoli Enterprise Portal Server to which you must authenticate, and through which communication to the monitoring server is established.
teps.server.port <i>integer</i>	-port	Specifies the port on which the portal server is listening.
username <i>string</i>	-user	Specifies the user name that is used to authenticate to the portal server. This user name must be a valid name that is known to the portal server and the monitoring server.
user.password <i>string</i>	-password	Specifies the password for the user connecting to the portal server.
os.agent <i>string</i>	-agent	Specifies the IP address of the OS agent on the remote computer against which the transfer operations are performed.

Table 99. *nco_ftp.props* properties and command-line options (continued)

Property	Command-line option	Description
transfer.action GET PUT	-action	Defines the type of transfer operation for the files. The options are: <ul style="list-style-type: none"> • GET: Retrieve files from a source location. • PUT: Transfer files back to their original location.
src.dir <i>string</i>	-src	Specifies the directory where the files are held on the source computer. Note: If you are transferring multiple files, all the files must be held in this location.
tgt.dir <i>string</i>	-dst	Specifies the directory to which files should be added on the target computer.
file.name <i>string1,...</i>	-file	Specifies a single file, or a comma-separated list of files to be transferred by the GET or PUT action. Note: If you are transferring multiple files, all the files must be held in the location specified by the src.dir property.
derestricted TRUE FALSE	N/A	Controls whether a file transfer operation can be conducted on files that are outside the Tivoli Netcool/OMNIbus installation directory. The options are: <ul style="list-style-type: none"> • TRUE: Restrict the file transfer operation to files in the Tivoli Netcool/OMNIbus directory only. • FALSE: Allow file transfer operations on files that are outside the Tivoli Netcool/OMNIbus directory.
N/A	-help	Displays help information relating to nco_ftp .
N/A	-properties	Specifies the fully-qualified file name of the <i>nco_ftp.props</i> file, which nco_ftp uses to control the file transfer. If the <i>nco_ftp.props</i> file and nco_ftp are in the same location, the path can be omitted.
N/A	-version	Displays version information relating to nco_ftp .

Removing probes from remote computers

You can remove the probes that have been deployed to remote computers.

To remove a probe from a remote computer:

1. If the probe is currently running under process control, stop the probe process in one of the following ways:

- Use Netcool/OMNIBus Administrator (**nco_config**) or run the **nco_pa_stop** command locally.

For information about stopping processes from Netcool/OMNIBus Administrator or by using **nco_pa_stop**, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

- Create and run a take action command that invokes the **nco_pa_stop** command remotely.

For detailed information about creating and running take action commands, see the *IBM Tivoli Monitoring Tivoli Enterprise Portal User's Guide* in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

2. If the probe is currently running as a Windows service, stop the service in one of the following ways:

- Stop the service from the Windows Services window.
- Run a take action command that uses the predefined Stop Service action name, and specify the service name of the probe.

3. From the monitoring server, run the **tacmd removeSystem** command, and then confirm that you want to remove the probe.

For example, to remove the Simnet probe with the bundle identifier **nco-p-simnet** from the installation location **/opt/IBM/tivoli/netcool** on the UNIX computer **hubserv.london.ibm.com**, enter:

```
tacmd removeSystem -t nco-p-simnet -n hubserv.london.ibm.com:UX -p  
KDY.INSTALLDIR=/opt/IBM/tivoli/netcool
```

For detailed information about the **tacmd removeSystem** command, see the *IBM Tivoli Monitoring Command Reference* documentation in the IBM Tivoli Monitoring Information Center at <http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp>.

Results

The command to remove the probe is passed to the OS agent on the remote computer. The OS agent then invokes the **uninstall** command in silent mode to remove the probe from the Tivoli Netcool/OMNIBus installation location.

Additional notes:

The uninstallation of a probe can primarily fail due to these causes:

- The probe files are in use
- Issues with file permissions
- Issues with the Deployment Engine component (and associated files) that are used for probe uninstallations

Any failures should be returned as IBM Tivoli Monitoring status messages.

If a failure occurs, review the Tivoli Netcool/OMNIBus installer log files on the remote computer to help determine the cause of the failure.

Tip: If required, you can run the **nco_cftp** utility with a GET action to retrieve the Tivoli Netcool/OMNIbus installer log files for review.

Related tasks

“Viewing the installation log files (UNIX and Linux)” on page 60

“Viewing the installation log files (Windows)” on page 124

“Using the file transfer utility (nco_cftp) to update files” on page 469

Chapter 26. Configuring the Web GUI component

After installing the Web GUI component, you must configure the data sources that the Web GUI uses. Depending on your requirements, you might also need to perform other configuration tasks to set up the system, such as defining secure connections between your servers and setting up password encryption.

Changing user registries after installation

Optional: After installation, you can change the user registry that you specified during the installation process, for example, from an ObjectServer user registry to an LDAP registry. If you change user registries, you must perform the configuration tasks associated with the new registry.

Related concepts

"Synchronization of user repositories" on page 29

Related tasks

"Troubleshooting user registries" on page 222

Removing user registries from the federated repository

If you no longer require a user registry, you can remove it from the Websphere federated repository.

To remove user registries from the federated repository:

1. Click **Settings > WebSphere Admin Console**, and click **Launch WebSphere Admin Console**.
2. Click **Security > Secure administration, applications, and infrastructure**.
3. From the **Available realms definition** list, select **Federated repositories** and click **Configure**.
4. In the Federated window, select the required entry from **Repositories in the realm:** and click **Remove**.
5. Click **OK** and then click **Save directly to master configuration**.

What to do next

If required, you can add new user registries, or select a different user registry in the federated repository to which the user credentials of new users and groups are written.

Related tasks

"Adding user registries to the federated repository" on page 478

"Switching the user registry to which user credentials are written" on page 479

Adding user registries to the federated repository

You can add user registries to the Websphere federated repository.

To add a user registry:

1. Click **Settings > WebSphere Admin Console** and then click **Launch WebSphere Admin Console**
2. Click **Security > Global Security**.
3. From the **Available realm definitions** list, select **Federated Repositories** and click **Configure**.
4. Click **Add Base Entity to Realm**.
5. Click **Add Repository** and complete the fields as follows:

Repository identifier

Type the name or ID of the user registry. This entry must not contain spaces.

Directory type

Select the type of user registry from the list.

Primary host name

Type the fully-qualified host name of the user registry.

Bind distinguished name

Specify the bind distinguished name. The default is o=ibm,cn=us.

Bind password

Type the password for the bind name.

6. Click **OK**.
7. In the **Distinguished name of a base entry that uniquely identifies this set of entries in the realm** field, type the following entry:
o=registryidentifier
8. In the **Distinguished name of a base entry in this repository** field, type the following entry:
cn=localhost
9. Click **OK**.
10. Under **Messages**, click **Save directly to master configuration**.
11. To verify that the repository has been added:
 - a. Click **Security > Global Security**.
 - b. On the "Secure administration, applications, and infrastructure" page, make sure that the DN was added to the **Repositories in the realm** table.
12. Restart the server.

Results

The user registry is added to the federated repository.

What to do next

You must perform the required configuration steps for the user registry that you added. If required, you can set the newly-added user registry as the registry to which new users and user groups are written.

If you want the users stored in the newly-added federated repository to have full access to Web GUI functions, enable the synchronization of the users with the ObjectServer.

Related tasks

“Restarting the server” on page 568

“Adding an external LDAP repository” on page 481

“Configuring VMM for the ObjectServer” on page 485

“Switching the user registry to which user credentials are written”

“Enabling user synchronization between a federated repository and the ObjectServer” on page 480

Switching the user registry to which user credentials are written

You can change the default user registry to which the credentials of new users and user groups are written.

You can select only one user registry to which the credentials of users and user groups are written when new users and user groups are created.

To change to a different registry:

1. Click **Settings > WebSphere Administration Console** and click on **Launch WebSphere Administration console**.
2. Click **Security > Global security**.
3. From the **Available realms definition** list, select **Federated repositories** and click **Configure**.
4. Under **Additional Properties**, click **Supported entity types**.
5. In the table, click the **Group** entity type and replace the properties in the **Base entry for the default parent** field and the **Relative Distinguished Name properties** field.
6. Click **OK** and then click **Save directly to the master configuration**.
7. Repeat steps 5 and 6 for the **OrgContainer** and **PersonAccount** entity types.

What to do next

If you switched from an ObjectServer to an LDAP server, you can enable the synchronization of user credentials between the LDAP server and the ObjectServer.

Related tasks

“Enabling user synchronization between a federated repository and the ObjectServer” on page 480

Enabling user synchronization between a federated repository and the ObjectServer

To enable users with profiles stored in an LDAP repository or the Tivoli Integrated Portal file-based repository to access all Web GUI functions, enable user synchronization.

Before you begin

Perform this task when you have switched authentication service from the ObjectServer to an LDAP repository or the Tivoli Integrated Portal file-based repository, and want to enable user synchronization.

If you specified the ObjectServer as the authentication service during installation, and want to switch to an LDAP or the Tivoli Integrated Portal repository and then use the synchronization process, disable the ObjectServer as the authentication service and replace it with a federated repository.

After user synchronization is enabled, the users and groups that are stored in LDAP or the Tivoli Integrated Portal repository are automatically synchronized with the ObjectServers that are configured in the `ncwDataSourceDefinitions.xml` file. In an ObjectServer, to distinguish between already-existing users and users that are created by the synchronization process, all users that are created by the synchronization process are assigned to a user group called “vmmusers”. If an ObjectServer does not already contain this user group, it is created automatically. The name “vmmusers” is a default, and can be changed.

To enable the synchronization of user credentials:

1. Edit the `webgui_home_dir/etc/server.init` file and set the **users.credentials.sync** property to TRUE.
2. Optional: To change the name of the vmmusers user group, assign the required value to the **users.credentials.sync.groupname** property.
3. Specify the intervals at which synchronization occurs:
 - a. Edit the `ncwDataSourceDefinitions.xml` file.
 - b. Set the `maxAge` attribute of the **config** property to the required time in seconds. For example:

```
<config maxAge="time"/>
```

The default is 3600 seconds.
4. Restart the server.

What to do next

If required, to trigger a synchronization request manually, use the `webgui_home_dir/bin/webtop_osresynch` tool. This tool uses the WAAPI API. The required **methodName** attribute is `osresync.refreshOSCache`. Before you use the `webtop_osresynch` tool, WAAPI must be configured properly.

Related tasks

“Restarting the server” on page 568

“Adding user registries to the federated repository” on page 478

Related reference

Appendix E, “Web GUI initialization file properties,” on page 633

Adding an external LDAP repository

After installation, you can add an IBM Tivoli Directory Server or Active Directory Microsoft Active Directory Server as an LDAP repository for Tivoli Netcool/OMNIBus Web GUI.

To add a new LDAP repository:

1. Log in to the Tivoli Netcool/OMNIBus Web GUI.
2. In the navigation pane, click **Settings > Websphere Admin Console** and click **Launch Websphere Admin Console**.
3. In the WebSphere Application Server administrative console, select **Settings > Global security**.
4. From the **Available realm definitions** list, select **Federated repositories** and click **Configure**.
5. In the Related Items area, click the **Manage repositories** link and then click **Add** to add a new LDAP repository.
6. In the **Repository identifier** field, provide a unique identifier for the repository. The identifier uniquely identifies the repository within the cell, for example, LDAP1.
7. From the **Directory type** list, select the type of LDAP server. The type of LDAP server determines the default filters that are used by WebSphere Application Server.

Note: IBM Tivoli Directory Server users can choose either IBM Tivoli Directory Server or SecureWay as the directory type. For better performance, use the IBM Tivoli Directory Server directory type.

8. In the **Primary host name** field, enter the fully qualified host name of the primary LDAP server. The primary host name and the distinguished name must contain no spaces. You can enter either the IP address or the domain name system (DNS) name.
9. In the **Port** field, enter the server port of the LDAP directory.

The host name and the port number represent the realm for this LDAP server in a mixed version nodes cell. If servers in different cells are communicating with each other using Lightweight Third Party Authentication (LTPA) tokens, these realms must match exactly in all the cells.

Note:

The default port value is 389, which is not a Secure Sockets Layer (SSL) connection port. Use port 636 for a Secure Sockets Layer (SSL) connection. For some LDAP servers, you can specify a different port. If you do not know the port to use, contact your LDAP server administrator.

10. Optional: In the **Bind distinguished name** and **Bind password** fields, enter the bind distinguished name (DN) (for example, cn=root) and password.

Note: The bind DN is required for write operations or to obtain user and group information if anonymous binds are not possible on the LDAP server. In most cases, a bind DN and bind password are needed, except when an anonymous bind can satisfy all of the required functions. Therefore, if the LDAP server is set up to use anonymous binds, leave these fields blank.

11. Optional: In the **Login properties** field, enter the property names used to log into the WebSphere Application Server. This field takes multiple login properties, delimited by a semicolon (;). For example, cn.

12. Optional: From the **Certificate mapping** list, select your preferred certificate map mode. You can use the X.509 certificates for user authentication when LDAP is selected as the repository.

Note: The **Certificate mapping** field is used to indicate whether to map the X.509 certificates into an LDAP directory user by EXACT_DN or CERTIFICATE_FILTER. If you select EXACT_DN, the DN in the certificate must match the user entry in the LDAP server, including case and spaces.

13. Click **OK**.
14. In the Messages area at the top of the Global security page, click the **Save** link and log out of the WebSphere Application Server console.

What to do next

Configure the Tivoli Integrated Portal Server to communicate with an external LDAP repository.

Related tasks

- “Troubleshooting user registries” on page 222
- “Configuring an SSL connection to an LDAP server” on page 487
- “Adding an external OpenLDAP repository”
- “Adding user registries to the federated repository” on page 478

Adding an external OpenLDAP repository

Instructions for configuring an OpenLDAP directory server as a repository.

To add an OpenLDAP repository:

1. Follow the instructions in “Adding an external LDAP repository” on page 481. When selecting the **Directory type** choose Custom.
2. Navigate to *tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/wim/config*.
3. Make a back up copy of wimconfig.xml.
4. Edit wimconfig.xml.
5. Locate the element that begins with:

```
<config:repositories xsi:type="config:LdapRepositoryType"
and ends with:
</config:repositories>
```

6. Replace that element and its child elements with the following:

```
<config:repositories xsi:type="config:LdapRepositoryType"
    adapterClassName="com.ibm.ws.wim.adapter.ldap.LdapAdapter"
    id="rep-identifier" isExtIdUnique="true" supportAsyncMode="false"
    supportExternalName="false" supportPaging="false"
    supportSorting="false"
    supportTransactions="false" certificateFilter=""
    certificateMapMode="exactdn" ldapServerType="CUSTOM"
    translateRDN="false">
    <config:baseEntries name="ldap-dn"
        nameInRepository="ldap-dn"/>
    <config:loginProperties>uid</config:loginProperties>
    <config:ldapServerConfiguration primaryServerQueryTimeInterval="15"
        returnToPrimaryServer="true" sslConfiguration="">
        <config:ldapServers authentication="simple" bindDN="bind-dn"
            bindPassword="bind-password" connectionPool="false"
        </config:ldapServers>
    </config:ldapServerConfiguration>
    connectTimeout="0"
    derefAliases="always" referral="ignore" sslEnabled="false">
    <config:connections host="primary-host" port="port-number"/>
```

```

        </config:ldapServers>
    </config:ldapServerConfiguration>
    <config:ldapEntityTypes name="Group">
        <config:objectClasses>groupOfNames</config:objectClasses>
    </config:ldapEntityTypes>
    <config:ldapEntityTypes name="OrgContainer">
        <config:rdnAttributes name="o" objectClass="organization"/>
        <config:rdnAttributes name="ou" objectClass="organizationalUnit"/>
        <config:rdnAttributes name="dc" objectClass="domain"/>
        <config:rdnAttributes name="cn" objectClass="container"/>
        <config:objectClasses>organization</config:objectClasses>
        <config:objectClasses>organizationalUnit</config:objectClasses>
        <config:objectClasses>domain</config:objectClasses>
        <config:objectClasses>container</config:objectClasses>
    </config:ldapEntityTypes>
    <config:ldapEntityTypes name="PersonAccount">
        <config:objectClasses>inetOrgPerson</config:objectClasses>
    </config:ldapEntityTypes>
    <config:groupConfiguration>
        <config:memberAttributes dummyMember="uid=dummy" name="member"
objectClass="groupOfNames"
scope="direct"/>
    </config:groupConfiguration>
    <config:cacheConfiguration>
        <config:attributesCache/>
        <config:searchResultsCache/>
    </config:cacheConfiguration>
</config:repositories>

```

Replace the following items in the elements:

rep-identifier

with a unique identifier for the repository. The identifier cannot contain spaces.

ldap-dn

with the distinguished name of the OpenLDAP server.

bind-dn

with the bind distinguished name.

bind-password

with the bind password.

primary-host

with the fully qualified name or TCP-IP address of the OpenLDAP host.

port-number

with the server port of the OpenLDAP directory.

7. Save `wimconfig.xml`.

8. Restart the server.

Related tasks

“Adding an external LDAP repository” on page 481

“Restarting the server” on page 568

Enabling LDAP authentication of ObjectServer users

You can enable users stored in an ObjectServer repository to be authenticated against an LDAP registry.

1. Enable the ObjectServer as the Tivoli Integrated Portal user repository.
2. Navigate to the directory *tip_home_dir/TIPProfile/config/cells/TIPCell/wim/config*.
3. Edit the file *wimconfig.xml*.
4. Locate the `<config:repositories>` element that has an `id` attribute with a value of `netcoolObjectServerRepository`.

For example:

```
<config:repositories
  adapterClassName="com.ibm.tivoli.tip.vmm4ncos.ObjectServerAdaptor"
  id="netcoolObjectServer" supportPaging="False">
  <config:baseEntries name="o=netcoolObjectServerRepository" />
  <config:CustomProperties name="password"
    value="{AES}F3A75EB49DC87013C11C6B021BA6B33" />
  <config:CustomProperties name="username" value="root" />
  <config:CustomProperties name="host1" value="localhost" />
  <config:CustomProperties name="port1" value="4100" />
</config:repositories>
```

5. Add the following `<config:CustomProperties>` elements to this element:

```
<config:CustomProperties name="LDAP.host" value="ldap-host" />
<config:CustomProperties name="LDAP.port" value="ldap-port" />
<config:CustomProperties name="LDAP.distinguishedName"
  value="user-dn-format" />
<config:CustomProperties name="LDAP.sslEnabled" value="ssl-enabled" />
```

Replace:

- *ldap-host* with the full name of the LDAP host server.
- *ldap-port* with the port number that the LDAP server uses. If the connection to the LDAP server uses SSL, specify the SSL port of the LDAP server (for example, 636).
- *user-dn-format* with the LDAP attributes that make up a user entry in the LDAP server. Depending on the LDAP implementation this consists of the string `uid=%username`, or the string `gid=%username`, followed by the LDAP attributes that identify the user.

Examples:

```
<config:CustomProperties name="LDAP.distinguishedName"
  value="uid=%username,cn=u50000g3000,cn=test,cn=ncw,o=ibm,c=uk" />
```

```
<config:CustomProperties name="LDAP.distinguishedName"
  value="gid=%username,cn=u50000g3000,cn=test,cn=ncw,o=ibm,c=uk" />
```

- *ssl-enabled* with `true` if the connection to the LDAP server uses SSL, otherwise use `false`.

Be sure to use the `%username` syntax. When a user logs in to the Web GUI the system replaces that syntax with the actual user name in the authentication request to the LDAP server.

Example:

```
<config:repositories
  adapterClassName="com.ibm.tivoli.tip.vmm4ncos.ObjectServerAdaptor"
  id="netcoolObjectServer" supportPaging="False">
  <config:baseEntries name="o=netcoolObjectServerRepository" />
  <config:CustomProperties name="password"
    value="{AES}F3A75EB49DC87013C11C6B021BA6B33" />
  <config:CustomProperties name="username" value="root" />
  <config:CustomProperties name="host1" value="localhost" />
```



```

<config:CustomProperties name="port1" value="4100" />
<config:CustomProperties name="LDAP.host" value="ldapserver.host.com" />
<config:CustomProperties name="LDAP.port" value="636" />
<config:CustomProperties name="LDAP.distinguishedName"
    value="uid=%username,cn=u50000g3000,cn=test,cn=ncw,o=ibm,c=uk" />
<config:CustomProperties name="LDAP.sslEnabled" value="true" />
</config:repositories>

```

- Before saving the file, carefully check the edits you have made. Pay particular attention to the syntax of all the elements.

Attention: If the syntax of `wimconfig.xml` is incorrect, you might not be able to log in to the Web GUI, to stop the server using the `stopServer` script. In this instance manually terminating the Tivoli Integrated Portal process may be necessary.

- Restart the server.

Results

Users can log in using their ObjectServer user ID and their LDAP password. They can no longer use their ObjectServer passwords.

What to do next

If the connection to the LDAP server uses SSL, configure that connection.

Related tasks

“Restarting the server” on page 568

“Switching the user registry to which user credentials are written” on page 479

“Configuring an SSL connection to an LDAP server” on page 487

Configuring VMM for the ObjectServer

When you want the ObjectServer to be in a federated repository, use a script to configure the Virtual Member Manager (VMM) adapter for the ObjectServer.

Before you begin

Obtain the following information about the ObjectServer:

- The user name
- The password
- The IP address
- The port number

If you have a second ObjectServer, you need its IP address and port number also.

The script assumes that the `tip_v2` installation directory is the parent directory and that the profile and cell names are `TIPProfile` and `TIPCell`. Run the VMM configuration script on every computer where the Web GUI is installed.

- Change to the `tip_home_dir\bin` directory, which contains the script to run:

- UNIX Linux `confvmm4ncos.sh`
- Windows `confvmm4ncos.bat`

- Enter the following command at the command line:

```
confvmm4ncos user password address port [address2 port2]
```

Where:

- user is the user ID of a user with administrative privileges for this ObjectServer.
- password is the password for the user ID.
- address is the IP address of the ObjectServer.
- port is the port number used by the ObjectServer.
- Optional: If there is a failover ObjectServer address2 and port2 are the IP address and port number of that ObjectServer.

Results

The VMM adapter is configured for the ObjectServer. Thereafter, whenever the user registry needs to be accessed, the VMM adapter is called for this information.

Related tasks

“Configuring an SSL connection to the ObjectServer” on page 488

“Adding user registries to the federated repository” on page 478

Configuring encryptions

Use the encryption functions to configure the Web GUI to the required level of security.

Configuring access for HTTP and HTTPS

By default, the application server requires HTTPS (Hypertext Transfer Protocol Secure) access. If you want some users to be able to log in and use the console with no encryption of transferred data, including user ID and password, configure the environment to support both HTTP and HTTPS modes.

Before you begin

After installing Tivoli Netcool/OMNIBus Web GUI and before beginning this procedure, log in to the portal to ensure that it has connectivity and can start successfully.

Configuring for HTTP and HTTPS console access involves editing the web.xml file of Web components. Use this procedure to identify and edit the appropriate Web XML files.

1. Change to the following directory: *tip_home_dir*/profiles/TIPProfile/config/cells/TIPCell/applications.
2. From this location, locate the web.xml files in the following directories:
 - For the Integrated Solutions Console Web application archive: *isclite.ear/deployments/isclite/isclite.war*/WEB-INF
 - For the Tivoli Integrated Portal Change Password Web application archive: *isclite.ear/deployments/isclite/TIPChangePasswd.war*/WEB-INF
3. Open one of the web.xml files using a text editor.
4. Find the <transport-guarantee> element. The initial value of all <transport-guarantee> elements is CONFIDENTIAL, meaning that secure access is always required.
5. Change the setting to NONE to enable both HTTP and HTTPS requests. The element now reads: <transport-guarantee>NONE</transport-guarantee>.
6. Save the file, and then repeat these steps for the other web.xml deployment files.

7. Stop and restart the application server.

Example

The following example is a section of the web.xml file for TIPChangePasswd where the transport-guarantee parameter is set to NONE:

```
<security-constraint>
  <display-name>
    ChangePasswdControllerServletConstraint</display-name>
  <web-resource-collection>
    <web-resource-name>ChangePasswdControllerServlet</web-resource-name>
    <url-pattern>/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <description>Roles</description>
    <role-name>administrator</role-name>
    <role-name>operator</role-name>
    <role-name>configurator</role-name>
    <role-name>monitor</role-name>
    <role-name>iscadmins</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>NONE</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

What to do next

Users must now specify a different port, depending on the mode of access. The default port numbers are as follows:

http://<host_name>:16310/ibm/console

Use the HTTP port for logging in to the Tivoli Integrated Portal on the HTTP port .

https://<host_name>:16311/ibm/console

Use the HTTPS secure port for logging in to the Tivoli Integrated Portal.

Note: If you want to use single sign-on (SSO) then you must use the fully qualified domain name of the Tivoli Integrated Portal host.

Configuring an SSL connection to an LDAP server

If your implementation of Tivoli Netcool/OMNIBus Web GUI uses an external LDAP-based user repository, such as Microsoft Active Directory, you can configure it to communicate over a secure SSL channel.

Before you begin

This task assumes that you have already an existing connection to an LDAP server set up.

Your LDAP server (for example, an IBM Tivoli Directory Server Version 6 or an Microsoft Active Directory server), must be configured to accept SSL connections and be running on secured port number (636). Refer to your LDAP server documentation if you need to create a signer certificate, which as part of this task, must be imported from your LDAP server into the trust store of the Tivoli Integrated Portal Server.

Follow these instructions to configure the Tivoli Integrated Portal Server to communicate over a secure (SSL) channel with an external LDAP repository. All application server instances must be configured for the LDAP server.

1. Log in to the portal.
2. Follow these steps to import your LDAP server's signer certificate into the application server trust store.
 - a. In the navigation pane, click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
 - b. Click **Security > SSL certificate and key management**.
 - c. In the Related Items area, click the **Key stores and certificates** link and in the table click the **NodeDefaultTrustStore** link.
 - d. In the Additional Properties area, click the **Signer certificates** link and click the **Retrieve from port** button.
 - e. In the relevant fields, provide hostname, port (normally 636 for SSL connections), SSL configuration details, as well as the alias of the certificate for your LDAP server and click the **Retrieve signer information** button and then click **OK**.
3. Follow these steps to enable SSL communications to your LDAP server:
 - a. In the navigation pane, click **Security > Secure administration, applications, and infrastructure**.
 - b. Select **Federated repositories** from the **Available realm definitions** drop down list and click **Configure**.
 - c. Select your LDAP server from the **Repository** drop down list.
 - d. Enable the **Require SSL communications** check box and the select the **Centrally managed** option.
 - e. Click **OK**.
4. For the changes to take effect, save, stop, and restart all Tivoli Integrated Portal Server instances.

What to do next

If you intend to enable single sign-on (SSO) so that users can log in once and then traverse to other applications without having to re-authenticate, configure SSO.

Related tasks

"Adding an external LDAP repository" on page 481

Configuring an SSL connection to the ObjectServer

For environments that include a Tivoli Netcool/OMNIBus ObjectServer user registry, you need to set up encrypted communications on the Tivoli Integrated Portal Server.

Follow these steps to establish a secure channel for communications between the Tivoli Integrated Portal Server and the ObjectServer.

1. Retrieve the ObjectServer certificate information, as follows:
 - a. In the Tivoli Integrated Portal navigation pane, click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
 - b. Click **Security > SSL certificate and key management**.
 - c. On the SSL certificate and key management page, click **Key stores and certificates** and on the page that is displayed, click **NodeDefaultTrustStore**.

- d. On the NodeDefaultTrustStore page, click **Signer certificates** and on the page that is displayed, click **Retrieve from port**.
- e. In the relevant fields, enter **Host**, **Port**, and **Alias** values for the ObjectServer and click **Retrieve signer information**.

The signer information is retrieved and stored. For your reference, when the signer information has been retrieved, the following details are displayed:

Serial number

Specifies the certificate serial number that is generated by the issuer of the certificate.

Issued to

Specifies the distinguished name of the entity to which the certificate was issued.

Issued by

Specifies the distinguished name of the entity that issued the certificate. This name is the same as the issued-to distinguished name when the signer certificate is self-signed.

Fingerprint (SHA digest)

Specifies the Secure Hash Algorithm (SHA hash) of the certificate, which can be used to verify the certificate's hash at another location, such as the client side of a connection.

Validity period

Specifies the expiration date of the retrieved signer certificate for validation purposes.

2. Open *tip_home_dir*/profiles/TIPProfile/etc/com.sybase.jdbc3.SybDriver.props in a text editor and change these parameters:
 - a. Enable SSL for ObjectServer primary host: USESSLPRIMARY=TRUE
 - b. Enable SSL for ObjectServer backup host: USESSLBACKUP=TRUE
3. Stop and restart the Tivoli Integrated Portal Server:
 - a. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:
 - Windows stopServer.bat server1
 - UNIX Linux stopServer.sh server1

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.
 - b. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:
 - Windows startServer.bat server1
 - UNIX Linux startServer.sh server1

Related tasks

“Configuring VMM for the ObjectServer” on page 485

Configuring SSL connections for the event feed from the ObjectServer

You can configure a Secure Socket Layer (SSL) connection for the feed of event data between the ObjectServer and the Web GUI

Before you begin

Tivoli Netcool/OMNIbus must be configured for SSL, and a certificate must have been created (or migrated if the Tivoli Netcool/OMNIbus installation is upgraded from an earlier version).

To create the secure connection, add the Tivoli Netcool/OMNIbus public certificate, which includes the Tivoli Netcool/OMNIbus public key, to the Tivoli Integrated Portal (unless Tivoli Integrated Portal already contains the signer certificate of the certificate authority (CA) that signed the certificate). You must also edit the Web GUI `server.init` file and identify the port used for SSL communication. After you have edited the `server.init` file, restart the Tivoli Integrated Portal server.

You can use either JKS or PKCS12 truststores, but the standard option described here is to use the default Tivoli Integrated Portal PKCS12 truststore. The default truststore is located in `tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes/TIPNode/trust.p12`. The default truststore password is WebAS.

Note: Alternatively, you can configure Web GUI certificates by using the IBM JRE-based Ikeyman tool provided with IBM JRE.

To configure an SSL connection for the event feed from the ObjectServer:

1. Add the Tivoli Netcool/OMNIbus certificate to the Tivoli Integrated Portal truststore:
 - a. Click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
 - b. Click **Security > SSL certificate and key management > Key stores and certificates > NodeDefaultTrustStore > Signer certificates**.
 - c. Click **Add**.
 - d. In the **Alias Name** field, type an alias for the certificate.
 - e. In the **File Name** field, type the path to the certificate.
 - f. From the **Data Type** list, select **Base64-encoded ASCII data** and click **OK**.
 2. Edit the `server.init` file:
 - a. Open `webgui_home_dir/etc/server.init`.
 - b. Set the `webtop.password.encryption` property to either None or AES.
 - c. Set the `webtop.fips` property to Off.
 - d. To ensure that the default Tivoli Integrated Portal truststore location is used, leave the `webtop.ssl.trustStore` property blank.
 - e. In the `webtop.ssl.trustStorePassword` property, leave the default Web GUI truststore password.
- Remember:** If you change the password using Tivoli Integrated Portal at any time, also edit the `server.init` file to reflect that change.
- f. Leave the default Web GUI trust manager type, IbmX509, and the default trust store type, PKCS12.

3. Define the ObjectServer port to be used for the SSL connection:
 - a. Open the `ncwDataSourceDefinitions.xml` file.
 - b. Set the **ncwPrimaryServer** property as shown in the following example:


```
<ncwPrimaryServer>
  <ncwOSConnection host="ObjectServerhost" port="ObjectServerport"
    ssl="true"/>
</ncwPrimaryServer>
```
4. Restart the server.

Related tasks

"Restarting the server" on page 568

Related reference

Appendix E, "Web GUI initialization file properties," on page 633

Replacing the default SSL certificate for connections to Web GUI clients

The Tivoli Integrated Portal includes a certificate for use in authenticating SSL connections to Web GUI clients. You can replace this certificate with one of your, either a certificate created by a Certification Authority (CA) or a self-signed certificate.

CA-signed certificate

Use this procedure to replace the default certificate with a CA-signed certificate.

The procedure has the following parts:

1. Create a request for the certificate.
2. Obtain the certificate from the CA.
3. Receive the certificate.
4. Add the certificate to the store.
5. Activate the certificate.

Creating a request for the certificate:

1. In the navigation pane of the Tivoli Integrated Portal, click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
2. Click **Security > SSL certificate and key management**.
3. On the "SSL certificate and key management" page, click **Key stores and certificates**, then click **NodeDefaultKeyStore**.
4. On the "NodeDefaultKeyStore" page, click **Personal certificate requests** and on the page that appears, click **New**.
5. In **File for certificate request** enter the path name for the file to hold the certificate request. Use the following form:


```
tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes/
request_file_name.p12
```

 Replace *request_file_name* with a suitable name for the request. For example: `ca-cert-request`.
6. Complete the fields in the "Certificate information" panel as follows:

Key label

Enter an alias name for the certificate request in the key store. Ensure it is unique among any other entries in the key store.

Common name

Enter the name of the entity that the certificate represents. For example the fully-qualified domain name where the Web GUI resides. For example: `webgui.server.mycompany.com`.

Organization

Enter the name of your organization to identify the organization part of the distinguished name. For example: My Company.

Organizational unit

Enter the name of the unit within the organization to identify the organizational unit part of the distinguished name. For example: Operations.

Locality

Enter the location of the organizational unit to identify the locality part of the distinguished name. For example: Armonk.

State or province

Enter the state or province of the locality to identify the state part of the distinguished name. For example: NY.

Zip code

Enter the zip or postal code of the locality to identify the zip code part of the distinguished name. For example: 10504.

Country

Select the code for your country from the drop-down list to identify the country part of the distinguished name. For example: US.

7. Click **Apply**.
8. On the "SSL certificate and key management" page, click **Save**.
9. Set the check box for the entry containing the new key label and click **Extract**.
10. On the "Extract certificate request" page enter the path of the file to hold the certificate request that you can send to the CA. Use the following form:
`tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes/
ca_request_file_name.p12`
 Replace `ca_request_file_name` with a suitable name for the request. For example: `cert-request-to-send-to-CA`.
11. Click **OK**.

Results

The system creates the file containing the request to send to the CA.

Obtaining the certificate from the CA:

Apply to your chosen Certification Authority for the certificate, typically using their web site. When asked to supply the request use the complete contents of the certificate request file. This is the file `tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes/ca_request_file_name.p12`. Replace `ca_request_file_name` with the name you gave the file that contains the certificate request. When you receive the certificate from the CA, copy it to a suitably named file, with a filename extension of `.p12`, in:
`tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes`

Receiving the certificate:

1. In the navigation pane of the Tivoli Integrated Portal, click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
2. Click **Security > SSL certificate and key management**.
3. On the "SSL certificate and key management" page, click **Manage endpoint security configurations**.
4. On the "Manage endpoint security configurations" page expand the **Inbound** node, if necessary, then click on **TIPNode(NodeDefaultSSLSettings)** under that node.
5. On the "TIPNode" page, click **Key stores and certificates** and on the page that appears, click **NodeDefaultKeyStore** in the table at the center of the page.
6. On the "NodeDefaultKeyStore" page, click **Personal certificates** and on the page that appears, click **Receive a certificate from a certificate authority**.
7. In the displayed form, enter the path of the file that contains the certificate from the CA then click **Apply**. For example:
`/opt/IBM/tivoli/tipv2/profiles/TIPProfile/config/cells/TIPCell/nodes/cert-from-ca.p12`
8. On the "SSL certificate and key management" page, click **Save**.

Results

The new certificate appears in the list of certificates on the "Personal certificates" page.

Adding the signer certificate to the store:

1. On the "Manage personal certificates" page, click **TIPNode** in the series of links at the top of the page.
2. On the "TipNode" page, click **Key stores and certificates** and on the page that appears click **NodeDefaultTrustStore** in the table at the center of the page.
3. Click **Signer Certificates** and on the page that appears click **Add**.
4. Complete the fields in the "Configuration" panel as follows:

Alias Enter an alias name for the certificate that is unique among the signer certificates in the key store.

File name

Enter the path of the file where you stored the certificate you received from the CA. For example:

`/opt/IBM/tivoli/tipv2/profiles/TIPProfile/config/cells/TIPCell/nodes/cert-from-ca.p12`

5. Click **Apply**.
6. On the "SSL certificate and key management" page, click **Save**.

Results

The certificate appears in the list of certificates on the "Signer certificates" page.

Activating the SSL certificate:

1. On the "Signer certificates" page, click **Manage endpoint security configurations** in the series of links at the top of the page.
2. On the "Manage endpoint security configurations" page expand the **Inbound** node, if necessary, then click on **TIPNode(NodeDefaultSSLSettings)** under that node.
3. On the "TIPNode" page choose the alias name of the certificate from the drop-down list in **Certificate alias in key store** and click **Apply**.
4. On the "TIPNode" page, click **Save**.

Self-signed certificate

Use this procedure to replace the default certificate with a self-signed certificate.

The procedure has the following parts:

1. Generate the certificate.
2. Assign the certificate.

Generating the certificate:

1. In the navigation pane of the Tivoli Integrated Portal, click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
2. Click **Security > SSL certificate and key management**.
3. On the "SSL certificate and key management" page, click **Manage endpoint security configurations**.
4. On the "Manage endpoint security configurations" page expand the **Inbound** node, if necessary, then click on **TIPNode(NodeDefaultSSLSettings)** under that node.
5. On the "TIPNode" page, click **Key stores and certificates** and on the page that appears, click **NodeDefaultKeyStore** in the table in the center of the page.
6. On the "NodeDefaultKeyStore" page, click **Personal certificates** and on the page that appears, click **Create > Self-signed certificate**.
7. Complete the fields in the "General Properties" panel as follows:

Alias Enter an alias name for the certificate request in the key store. Ensure it is unique among any other entries in the key store.

Common name

Enter the name of the entity that the certificate represents, such as the fully-qualified domain name where the Web GUI resides. For example: `webgui.server.mycompany.com`.

Organization

Enter the name of your organization to identify the organization part of the distinguished name. For example: My Company.

Organization unit

Enter the name of the unit within the organization to identify the organizational unit part of the distinguished name. For example: Operations.

Locality

Enter the location of the organizational unit to identify the locality part of the distinguished name. For example: Armonk.

State/Province

Enter the state or province of the locality to identify the state part of the distinguished name. For example: NY.

Zip code

Enter the zip or postal code of the locality to identify the zip code part of the distinguished name. For example: 10504.

Country

Select the code for your country from the drop-down list to identify the country part of the distinguished name. For example: US.

8. Click **Apply**.

9. On the "SSL certificate and key management" page, click **Save**.

Results

The new certificate appears in the list of certificates on the "Manage personal certificates" page.

Assigning the certificate:

1. On the "Personal certificates" page click the **TIPNode** link in the series of links at the top of the page.
2. Choose the alias name for the certificate from the drop-down list in **Certificate alias in key store** and click **Apply**.
3. On the "TIPNode" page, click **Save**.
4. Click the **Manage endpoint security considerations** link in the series of links at the top of the page.

The alias name of the certificate appears in brackets after the entry for TIPNodeNodeDefaultSSLSettings under Inbound.

Encrypting Web GUI passwords

To encrypt Web GUI passwords for non-SSL and SSL connections, use the **ncw_aes_crypt** tool.

Before you begin

You can use AES encryption only if FIPS 140-2 mode has not been enabled.

You must encrypt the ObjectServer passwords that are stored in the `ncwDataSourceDefinitions.xml` file, and the Tivoli Integrated Portal truststore password that is stored in the `server.init` file. The default truststore password is WebAS. After you have edited the `server.init` file, restart the Tivoli Integrated Portal server.

To encrypt the Web GUI passwords:

1. Enable AES encryption in the ObjectServer used as a data source:
 - a. Edit the ObjectServer properties file and set the value of the **PasswordEncryption** property to AES.

The path of the ObjectServer properties file is as follows:

- **UNIX** **Linux** \$NCHOME/omnibus/etc/*servername*.props
- **Windows** %NCHOME%\omnibus\etc*servername*.props

Replace *servername* with the name of the ObjectServer.

- b. Reset all ObjectServer user account passwords.

- c. Restart the ObjectServer.
2. Encrypt the ObjectServer password:
 - a. Run `webgui_home_dir/bin/ncw_aes_crypt`.
 - b. Type the ObjectServer password.
An encrypted ObjectServer password is generated.
 - c. Copy the encrypted password.
3. Add the encrypted ObjectServer password to the data source configuration file:
 - a. Open the `ncwDataSourceDefinitions.xml` file.
 - b. Edit the **ncwDataSourceCredentials** property, as shown in the following example:


```
<ncwDataSourceCredentials
  userName="root" password="encryptedObjectServerpassword"
  encrypted="true"
  algorithm="AES"/>
```

 Replace `encryptedObjectServerpassword` with the encrypted password you copied in step 1 on page 495
4. Encrypt the Tivoli Integrated Portal truststore password:
 - a. Run `webgui_home_dir/bin/ncw_aes_crypt`.
 - b. Type the default Tivoli Integrated Portal truststore password, WebAS.
An encrypted password is generated.
 - c. Copy the encrypted password.
5. Add the encrypted truststore password to the initialization file:
 - a. Open the `webgui_home_dir/etc/server.init` file.
 - b. Set the **webtop.password.encryption** property to `aes`.
 - c. Set the **webtop.ssl.trustStorePassword** property to the password encrypted in step 4b.
 - d. To ensure that the default Tivoli Integrated Portal truststore is used, leave **webtop.ssl.trustStore** empty .
 - e. Set **webtop.fips** to `off`.
6. Restart the server.

Related tasks

“Restarting the server” on page 568

Related reference

Appendix E, “Web GUI initialization file properties,” on page 633

Enabling FIPS 140-2 mode for the Web GUI

To enable the Web GUI in FIPS 140–2 mode, you must perform several configuration steps.

Enabling FIPS 140–2 mode for the Tivoli Integrated Portal Server

You can configure the Tivoli Integrated Portal Server to use Federal Information Processing Standard Java Secure Socket Extension files.

Follow these steps to enable FIPS 140–2 for the Tivoli Integrated Portal Server.

1. Configure the application server to use FIPS.
 - a. In the portal, click **Security > SSL certificate and key management**.
 - b. Select the **Use the United States Federal Information Processing Standard (FIPS) algorithms** option and click **Apply**. This option makes IBMJSSE2 and IBMJCEFIPS the active providers.
2. Configure the application server to use FIPS algorithms for Java clients that must access enterprise beans:
 - a. Open the *install_dir/profiles/TIPProfile/properties/ssl.client.props* file in a text editor.
 - b. Change the `com.ibm.security.useFIPS` property value from `false` to `true`.
3. Configure the application server to use FIPS algorithms for SOAP-based administrative clients that must access enterprise beans:
 - a. Open the *install_dir/profiles/TIPProfile/properties/soap.client.props* file in a text editor.
 - b. Add this line: `com.ibm.ssl.contextProvider=IBMJSSEFIPS`.
4. Configure `java.security` to enable IBMJCEFIPS:
 - a. Open the *install_dir/java/jre/lib/security/java.security* file in a text editor.
 - b. Insert the IBMJCEFIPS provider (`com.ibm.crypto.fips.provider.IBMJCEFIPS`) before the IBMJCE provider, and also renumber the other providers in the provider list. The IBMJCEFIPS provider must be in the `java.security` file provider list. See the example at the end of this topic.
5. Enable your browser to use Transport Layer Security (TLS) 1.0:
 - a. Microsoft Internet Explorer: Open the Internet Explorer and click **Tools > Internet Options**. On the **Advanced** tab, select the **Use TLS 1.0** option.
 - b. Firefox: TLS 1.0 is enabled by default.
6. Export Lightweight Third Party Authentication keys so applications that use these LTPA keys can be reconfigured.
 - a. In the navigation pane, click **Settings > Websphere Admin Console** and click **Launch Websphere Admin Console**.
 - b. In the WebSphere Application Server administrative console, select **Settings > Global security**.
 - c. In the Global security page, from the Authentication area, click the **LTPA** link.
 - d. Under **Cross-cell single sign-on**, specify a key file and provide a filename and password for the file that will contain the exported LTPA keys.
 - e. Click **Export keys**.
7. Reconfigure any applications that use Tivoli Integrated Portal Server LTPA keys: To reconfigure the Tivoli SSO service with the updated LTPA keys, run this script: *tip_home_dir/profiles/TIPProfile/bin/setAuthnSvcLTPAKeys.jacl*.
 - a. Change directory to *tip_home_dir/profiles/TIPProfile/bin/*
 - b. Start the Tivoli Integrated Portal Server:
 - Windows `startServer.bat server1`
 - UNIX Linux `startServer.sh server1`

- c. Run the following command:

```
wsadmin -username tipadmin -password tipadmin_password -f  
setAuthnSvcLTPAKeys.jacl exported_key_path key_password
```

Where:

exported_key_path is name and full path to the key file that was exported.

key_password is the password that was used to export the key.

8. For SSO, enable FIPS for any other application servers, then import the updated LTPA keys from the first server into these servers:
- Copy the LTPA key file from step 4 above to another application server computer.
 - In the navigation pane, click **Settings > Websphere Admin Console** and click **Launch Websphere Admin Console**.
 - In the WebSphere Application Server administrative console, select **Settings > Global security**.
 - In the Global security page, from the Authentication area, click the **LTPA** link.
 - Under **Cross-cell single sign-on**, provide the filename and password from above for the file that contains the exported LTPA keys.
 - Click **Import keys**.
9. Run the ConfigureCLI command:

```
Windows tip_home_dir\bin\tipcli.bat ConfigureCLI --useFIPS true  
Linux UNIX tip_home_dir/bin/tipcli.sh ConfigureCLI --useFIPS  
true
```

Example

The IBM SDK *tip_home_dir*/java/jre/lib/security/java.security file looks like this when IBMJCEFIPS is enabled.

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS  
security.provider.2=com.ibm.crypto.provider.IBMJCE  
security.provider.3=com.ibm.jsse.IBMJSSEProvider  
security.provider.4=com.ibm.jsse2.IBMJSSEProvider2  
security.provider.5=com.ibm.security.jgss.IBMJGSSProvider  
security.provider.6=com.ibm.security.cert.IBMCertPath  
security.provider.7=com.ibm.crypto.pkcs11.provider.IBMPKCS11  
security.provider.8=com.ibm.security.cmskeystore.CMSProvider  
security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
```

Configuring the Web GUI client for FIPS-0142 mode

To use the Web GUI client in FIPS-0142 mode, make sure that the client is configured correctly.

To configure the Web GUI client for FIPS-0142 mode:

- Make sure that browsers have Transport Layer Security (TLS) enabled.
- If the client-side browser uses IBM JRE 1.5, add the **-Dhttps.protocols=TLSv1** Java runtime parameter to enforce the use of TLS. To add this parameter:
 - Click **Java Control Panel > Java > View**
 - In the Java Runtime Parameter window, double click the field for the IBM JRE 1.5 under **Java Runtime Parameter** and type the following entry:
-Dhttps.protocols=TLSv1

Encrypting passwords using FIPS 140–2 mode encryption

To encrypt Web GUI passwords in FIPS 140–2 mode for non-SSL and SSL connections, use the **ncw_fips_crypt** FIPS 140–2 encryption tool.

Before you begin

To use the FIPS 140–2 encryption tool, you must have IBM JRE installed.

The default Web GUI vault key is located in *webgui_home_dir/etc/encrypt/vault.key*. This key is used to encrypt the ObjectServer password stored in the *ncwDataSourceDefinitions.xml* file, and the Tivoli Integrated Portal truststore password stored in the *server.init* file. The default truststore password is WebAS. After you have edited the *server.init* file, restart the Tivoli Integrated Portal server.

1. To encrypt the ObjectServer password, enter the following command:

```
webgui_home_dir/bin/ncw_fips_crypt -password netcool -key  
webgui_home_dir/etc/encrypt/vault.key
```

If you use the default vault key, omit the **key** parameter.
An encrypted password is generated.
2. Copy the encrypted password.
3. Add the encrypted ObjectServer password:
 - a. Open the *ncwDataSourceDefinitions.xml* file.
 - b. Edit the `<ncwDataSourceCredentials>` element, as shown in the following example:

```
<ncwDataSourceCredentials  
  userName="root" password="encryptedObjectServerpassword" encrypted="true"  
  algorithm="FIPS"/>
```
4. To encrypt the Tivoli Integrated Portal truststore password, enter the following command:

```
webgui_home_dir/bin/ncw_fips_crypt -password WebAS -key  
webgui_home_dir/etc/encrypt/vault.key
```

If you use the default vault key, omit the **key** parameter.
An encrypted password is generated.
5. Copy the encrypted password.
6. Add the encrypted Tivoli Integrated Portal truststore password to the initialization file:
 - a. Open the *webgui_home_dir/etc/server.init* file.
 - b. Set the **webtop.password.encryption** property to **fips**.
 - c. Set the **webtop.ssl.trustStorePassword** property to the encryption generated in step 4.
 - d. To ensure that the default Tivoli Integrated Portal truststore is used, leave the **webtop.ssl.trustStore** property empty.
 - e. Set the **webtop.fips** property to **on**.
7. Restart the server.

Related reference

Appendix E, “Web GUI initialization file properties,” on page 633

Configuring SSL connections in FIPS 140–2 mode for the event feed from the ObjectServer

You can configure a Secure Socket Layer (SSL) connection in FIPS 140–2 mode for the feed of event data between the ObjectServer and the Web GUI

Before you begin

Tivoli Netcool/OMNIBus must be configured for SSL, and a certificate must have been created (or migrated if the Tivoli Netcool/OMNIBus installation is upgraded from an earlier version).

You can use either JKS or PKCS12 truststores, but the standard option described here is to use the default Tivoli Integrated Portal PKCS12 truststore. The default truststore is located in *tip_home_dir*/profiles/TIPProfile/config/cells/TIPCell/nodes/TIPNode/trust.p12. The default truststore password is WebAS.

Note: In addition to the method described here, you can also configure Web GUI certificates using the IBM JRE-based Ikeyman tool provided with IBM JRE.

1. Add the Tivoli Netcool/OMNIBus certificate to the Tivoli Integrated Portal truststore:
 - a. Click **Settings > WebSphere Administrative Console**, and click **Launch WebSphere administrative console**.
 - b. Click **Security > SSL certificate and key management > Key stores and certificates > NodeDefaultTrustStore > Signer certificates**.
 - c. Click **Add**.
 - d. In the **Alias Name** field, type an alias for the certificate.
 - e. In the **File Name** field, type the path to the certificate.
 - f. From the **Data Type** list, select **Base64-encoded ASCII data** and click **OK**.
2. Edit the `server.init` file:
 - a. Open `webgui_home_dir/etc/server.init`.
 - b. Set the `webtop.password.encryption` property to either None or AES.
 - c. Set the `webtop.fips` property to 0n.
 - d. To ensure that the default Tivoli Integrated Portal truststore location is used, leave the `webtop.ssl.trustStore` property blank.
 - e. In the `webtop.ssl.trustStorePassword` property, leave the default Web GUI truststore password.

Remember: If you change the password using Tivoli Integrated Portal at any time, you must also edit the `server.init` file to reflect that change.

 - f. Leave the default Web GUI trust manager type, IbmX509, and the default trust store type, PKCS12.
3. Define the ObjectServer port to be used for the SSL connection:
 - a. Open the `ncwDataSourceDefinitions.xml` file.
 - b. Set the `ncwPrimaryServer` property as shown in the following example:

```
<ncwPrimaryServer>
<ncwOSConnection host="ObjectServerhost" port="ObjectServerport"
ssl="true"/>
</ncwPrimaryServer>
```
4. Enable FIPS 140–2 in the Tivoli Integrated Portal server:
 - a. Open `install_dir/java/jre/lib/security/java.security`.
 - b. In the list of providers, uncomment the following line:

security.provider:<x>=com.ibm.crypto.fips.provider.IBMJCEFIPS

- c. Replace the <x> variable with a number that reflects the order of preference you want to set, and renumber the subsequent security providers.
 - d. On the Tivoli Integrated Portal console, click **Security > SSL Certificate and key management**.
 - e. Under **Configuration Settings**, select **Use the United States Federal Information Processing Standard (FIPS) algorithms**.
 - f. Click **Apply**, then **Save**.
5. Restart the server.

Related tasks

“Restarting the server” on page 568

Configuring Tivoli Access Manager in Tivoli Integrated Portal

You can configure Tivoli Integrated Portal to use Tivoli Access Manager WebSEAL Version 6.1 to manage authentication.

You must install and configure Tivoli Access Manager WebSEAL Version 6.1. To set up and configure Tivoli Access Manager WebSEAL, see http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc/am611_install196.htm#webseal.

For more information on administering Tivoli Access Manager WebSEAL, see http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc/am611_webseal_admin.htm.

Configuring single sign-on using ETai

In a WebSphere Application Server (WAS) environment, Tivoli Access Manager WebSEAL can be used as a reverse proxy to intercept incoming http or https requests to ensure that users are authenticated and authorized and are passed to the relevant Tivoli Integrated Portal Server .

ETai is the component that implements the WebSphere Application Server trust association interceptor interface to achieve single sign on from WebSEAL to the Tivoli Integrated Portal Server.

Tivoli Integrated Portal supports single sign-on (SSO) with perimeter authentication services such as reverse proxies through trust associations. When trust associations are enabled, the WebSphere Application Server is not required to authenticate a user if a request arrives from a trusted source that has already performed authentication.

Once a trust association is configured between WebSEAL and the Tivoli Integrated Portal Server, a user can login into Tivoli Access Manager and then access the Tivoli Integrated Portal Server without having to re-authenticate. The ETai must be configured in Tivoli Integrated Portal Server and is responsible for establishing trust against the WebSEAL server. ETai simplifies the use of Tivoli Access Manager and the configuration required to achieve SSO. One advantage is that Tivoli Access Manager and Tivoli Integrated Portal can use different user registries and still be able to perform SSO. It also provides the mapping between different registry formats.

Installing ETai:

Use these instructions, to install the Tivoli Access Manager Extended Trust Association Interceptor in a Tivoli Integrated Portal environment.

Before you begin

Source a copy of `com.ibm.sec.authn.tai.etai_6.0.jar` from your installation media.

To install ETai:

1. Copy `com.ibm.sec.authn.tai.etai_6.0.jar` to the `plugins` directory.
2. At the command line, depending on your operating system, run the relevant command:
 - **Windows** `tip_home_dir\bin\0sgicfginit.bat`
 - **UNIX** **Linux** `tip_home_dir/bin/0sgicfginit.sh`
3. Copy `pd.jar` to `tip_home_dir/java/jre/lib/ext`

What to do next

Configure ETai in a Tivoli Integrated Portal environment.

Enabling a trust association for ETai:

You must enable a trust association between the Tivoli Access Manager Extended Trust Association Interceptor in the Tivoli Integrated Portal environment.

To configure a trust association for ETai:

1. Log in to the portal and click **Settings > WebSphere Administrative Console**.
2. In the WebSphere Administrative Console page, click **Launch WebSphere administrative console**.
3. In the WebSphere Administrative Console navigation pane, click **Global security**.
4. In the Global security page, expand **Web security** and click **Trust association**.
5. In the General Properties area, click the **Enable trust association** option if it is disabled and click **Apply**.

Your update is saved and you are returned to the Global security page.
6. In the Global security page, expand **Web security** and click **Trust association** to display the Trust association page.
7. In the Additional properties area, click the **Interceptors** link to display the Interceptors page.
8. If `com.ibm.sec.authn.tai.TAMETai` is not listed on the page, click **New**.
9. In the **Interceptor class name** field enter the string `com.ibm.sec.authn.tai.TAMETai` and click **Apply**.
10. In the Messages area, click the **Save** link to commit your change.

What to do next

Configure ETai in the a Tivoli Integrated Portal environment.

Configuring custom properties for ETai:

Once you have enabled a trust association for the Tivoli Access Manager Extended Trust Association Interceptor in the Tivoli Integrated Portal environment, you must configure its custom properties.

To configure custom properties for the ETai:

1. Log in to the portal and click **Settings > WebSphere Administrative Console**.
2. In the WebSphere Administrative Console page, click **Launch WebSphere administrative console**.
3. In the WebSphere Administrative Console navigation pane, click **Global security**.
4. In the Global security page, expand **Web security** and click **Trust association** to display the Trust association page.
5. In the Additional properties area, click the **Interceptors** link to display the Interceptors page.
6. From the list of interceptor classes, select the `com.ibm.sec.authn.tai.TAMETai` entry.
7. In the Additional properties area, click the **Custom properties** link to display the Custom properties page.
8. Review the details for the custom properties listed in Table 1:

Table 100. ETai custom properties

Property details	Notes®
Property name: com.ibm.websphere .security.webseal .useWebSphereUserRegistry Type: string Required: Yes Values: true or false Default value: true	ETai authenticates the trusted user against the WebSphere Application Server user registry or the Tivoli Access Manager Authorization Server. If this property is set to true, the resulting Subject will not contain a PDPrincipal as the Tivoli Access Manager Authorization Server is required to build the PDPrincipal. Any other value for this property will result in a PDPrincipal being added to the Subject.
Property name: com.ibm.websphere .security.webseal .tamUserDnMapping Required: Yes Value: WAS Default value: TAM	The ETai adds users' credential information into the JAAS Subject. This information includes the users dn. Maps this dn to the WebSphere Application Server dn, or (Value = WAS). If a mapping is attempted for a user that does not exist in the WebSphere Application Server user registry, it is ignored and not added to the JAAS Subject.

Table 100. ETai custom properties (continued)

Property details	Notes®
Property name: com.ibm.websphere .security.webseal .tamGroupDnMapping Required: Yes Value: WAS Default value: TAM	<p>The ETai adds users' credential information into the JAAS Subject. This information includes the group dn's. The ETai can be configured to either:</p> <p>Map these dn's to the WebSphere Application Server dn's, or (Value = WAS).</p> <p>If a mapping is attempted for a group that does not exist in the WebSphere Application Server user registry, it is ignored and not added to the JAAS Subject.</p>
Property name: com.ibm.websphere .security.webseal .loginId Type: String Required: Yes Value: websealSSID Default value: None	<p>The value of this property must exist as a valid user in the user registry.</p> <p>If necessary, create a new user in the Tivoli Integrated Portal registry called websealSSID.</p> <p>The ETai must be configured with the username of the WebSEAL trusted user. This is the single sign-on user that is authenticated using the password in the Basic Authentication header inserted by WebSEAL in the request. The format of the username is the short name representation.</p> <p>This property interacts with the following property:</p> <p>com.ibm.websphere.security .webseal.useWebSphereUserRegistry</p> <p>If com.ibm.websphere.security .webseal.useWebSphereUserRegistry is set to true then the specified user must exist in either the WebSphere Application Server user registry or the Tivoli Access Manager user registry.</p>
Property name: com.ibm.websphere .security.webseal .checkViaHeader Type: String Required: Yes Value: true Default value: false	<p>The ETai can be configured so that the Via header can be ignored when validating trust for a request. This property is required, if WebSEAL is to allow requests into the Tivoli Integrated Portal only from particular hosts.</p> <p>This property interacts with the following properties:</p> <ul style="list-style-type: none"> com.ibm.websphere.security.webseal.hostnames com.ibm.websphere.security.webseal.ports <p>If com.ibm.websphere.security .webseal.checkViaHeader is set to false then the values set for the two associated properties are not used.</p>

Table 100. ETai custom properties (continued)

Property details	Notes®
Property name: <code>com.ibm.websphere</code> <code>.security.webseal.id</code> Required: Yes Value: <code>iv-creds</code> Default value: <code>iv-creds</code>	<p>Iv-creds carries end user credentials, which is used by Tivoli Integrated Portal for authorization.</p> <p>Note: Any additional values set for this property are added to a list along with Iv-creds, that is, Iv-creds is a required header for the ETai.</p>
Property name: <code>com.ibm.websphere</code> <code>.security.webseal</code> <code>.hostnames</code> Required: Yes Value: A comma separated list of strings. Default value: There is no default value for this property.	<p>The ETai can be configured so that the request must arrive from a list of expected hosts. If any of the hosts in the Via header of the HTTP request are not listed in the values set for this property, the request is ignored by the ETai.</p> <p>This property interacts with the following property: <code>com.ibm.websphere.security.webseal.ports</code></p> <p>All of the values listed for <code>com.ibm.websphere.security.webseal.hostnames</code> are used with the ports listed for <code>com.ibm.websphere.security.webseal.ports</code> to indicate a trusted host.</p> <p>For example, if:</p> <pre>com.ibm.websphere.security.webseal.hostnames is set to abc,xyz com.ibm.websphere.security.webseal.ports is set to 80,443</pre> <p>Then, the Via header is checked for these hostname/port combinations: <code>abc:80</code>; <code>abc:443</code>; <code>xyz:80</code>; <code>xyz:443</code>.</p> <p>If <code>com.ibm.websphere.security.webseal.checkViaHeader</code> is set to false then the values set for <code>com.ibm.websphere.security.webseal.hostnames</code> are not used.</p>
Property name: <code>com.ibm.websphere</code> <code>.security.webseal</code> <code>.ports</code> Required: Yes Value: <code>443</code> Default value: There is no default value for this property.	<p>This property interacts with the following property: <code>com.ibm.websphere.security.webseal.hostnames</code></p> <p>All of the values listed for <code>com.ibm.websphere.security.webseal.hostnames</code> are used with the ports listed for <code>com.ibm.websphere.security.webseal.ports</code> to indicate a trusted host.</p> <p>For more information, see the notes for <code>com.ibm.websphere.security.webseal.hostnames</code>.</p>

Table 100. ETai custom properties (continued)

Property details	Notes®
Property name: com.ibm.websphere .security.webseal .ssoPwdExpiry Required: No Value: A positive integer. Default value: 600	Once trust has been established for a request, the password for the Single sign-on user is cached for subsequent trust validation of requests. This saves the ETai from having to re-authenticate the single sign-on user with the user registry for every request, therefore increasing performance. The cache timeout period can be modified by setting this property to the required time in seconds. If the password expiry property is set to 0, the cached password does not expire.
Property name: com.ibm.websphere .security.webseal .groupRealmPrefix Required: Yes Value: "group:" Default value: 600	This property is needed to map the group realm prefix from Tivoli Access Manager to group realm prefix in WebSphere Application Server registry.
Property name: com.ibm.websphere .security.webseal .userRealmPrefix Required: Yes Value: "user:" Default value: 600	This property is needed to map the user realm prefix from Tivoli Access Manager to group realm prefix in WebSphere Application Server registry.

9. If a custom property does not exist, click **New** to configure a custom property and provide a name, value, and optional description and click **Apply** to add the custom property.
10. If the custom property exists, but is not in line with the details provided in the table above, click on the custom property entry, update its details and click **Apply** to modify the custom property.
11. Stop and restart the Tivoli Integrated Portal Server:
 - a. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - Windows `stopServer.bat server1`
 - UNIX Linux `stopServer.sh server1`

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.
 - b. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - Windows `startServer.bat server1`

- UNIX Linux `startServer.sh server1`

What to do next

Configure the Tivoli Access Manager WebSEAL by creating a WebSEAL junction and creating a junction mapping table.

Checking your Tivoli Access Manager configuration

To ensure that your Tivoli Access Manager configuration is valid, you can carry out a number of checks.

Before you begin

Ensure that you have the following software versions installed:

- Tivoli Access Manager version 6.1
- Tivoli Integrated Portal Server, version 1.1 fix pack 11 or later

This topic describes how to check the following items:

- The status of the Tivoli Access Manager server.
 - Connecting to the Tivoli Integrated Portal Server.
1. To check the status of the Tivoli Access Manager server, at the command line, enter `pd start status`.

The following output indicates that the Tivoli Access Manager server is running:

```
pdmgrd yes yes
pdacld yes no (sometimes yes)
pdmgrproxyd no no
webseald-ip1 yes yes
```

2. To check if the Lightweight Directory Access Protocol (LDAP) user registry is active:

- a. At the command line, enter `pdadmin -a sec_master -p sec_master_password`.

Note: This command assumes that `pdadmin` is in the path.

Expected output:

```
pdadmin -a sec_master -p sec_master_password
```

- b. At the command line, enter `user list * 10`.

Example output:

```
sec_master
ivmgrd/master
ivaclld/ip1
ip1-webseald/ip1
```

- c. To quit, at the command line, enter `quit`.
3. If the Tivoli Access Manager processes are not started, at the command line enter `pd start start`.

If the processes are already started, the following output can be expected:

```
Starting the: Access Manager authorization server
Could not start the server
```

4. To check that you can connect from the Tivoli Integrated Portal Server to the Tivoli Access Manager computer:

- a. On the Tivoli Integrated Portal Server use a Web browser to connect to `http://tam_server_hostname`. A security message may be displayed, confirm the Tivoli Access Manager self-signed certificate to display an authorization dialog.
- b. Enter a username and password to display the Tivoli Access Manager WebSEAL splash screen (username = `sec_master`, password = `sec_master_password`).

What to do next

Configure the WebSEAL keystore.

Configuring the WebSEAL keystore

To allow the application server to use Tivoli Access Manager WebSEAL, you must import Tivoli Integrated Portal Server security certificate to the WebSEAL keystore.

To export the Tivoli Integrated Portal Server security certificate and import it into the WebSEAL keystore:

1. Log in to the Tivoli Integrated Portal console.
2. Export the Tivoli Integrated Portal X.509 certificate. The process for exporting varies depending on your browser. Refer to your browser documentation for assistance. For example, the following substeps describe how you can export the certificate using a Firefox browser:
 - a. Double-click on lock icon on lower right hand side of browser window to display the Security dialog for the Web page.
 - b. Click **View Certificate** and in the Certificate Viewer dialog and then click the **Details** tab.
 - c. Click **Export** and in the Save Certificate To File dialog and select a directory to export the Tivoli Integrated Portal X.509 certificate.
3. Copy the exported certificate file to the Tivoli Access Manager computer.
4. On the Tivoli Access Manager computer, at the command line, change to the directory that hosts the IKeyman utility.
5. Start the IKeyman utility and complete the substeps:
 - **UNIX** **Linux** At the command line, enter `./ikeyman.sh`
 - **Windows** At the command line, enter `ikeyman.exe`
 - a. On the toolbar, click **Open** to display the Open window.
 - b. Select CMS as the key database type.
 - c. Click **Browse** and from `/var/pdweb/www-ip1/certs`, select `pdsrv.kdb` to display the Password Prompt dialog. The default password reflects the file name, that is, `pdsrv`.
 - d. In the Key database content section, select **Signer Certificates** and click **Add**.
 - e. In the Add CA's Certificate from a File dialog, for the **Data type**, select the Base64-encoded ASCII data option and click **Browse**.
 - f. Locate the Tivoli Integrated Portal X.509 certificate and enter a label for the certificate (for example, `tipmachine`).
 - g. Click **Save** to add the certificate to the WebSEAL keystore (do not change the certificate's file name).
6. To restart Tivoli Access Manager WebSEAL, at the command line, enter `pdweb restart`.

The following is the expected output:

Stopping the: webseald-ip1
Starting the: webseald-ip1

What to do next

Create a WebSEAL junction.

Creating a WebSEAL junction

A WebSEAL junction is an HTTP or HTTPS connection between a front-end WebSEAL server and a back-end Web application server, for example the Tivoli Integrated Portal Server.

Junctions logically combine the Web space of the back-end server with the Web space of the WebSEAL server, resulting in a unified view of the entire Web object space. To create a junction:

1. On the Tivoli Access Manager computer, at the command line, enter `pdadmin -a sec_master_account -p sec_master_password`.
2. At the command line, enter `s l`.

The following is the expected output:

```
ivacld-ip1
ip1-webseald-ip1
```

Note: Where `ip1` is the hostname of the Tivoli Access Manager computer.

3. Enter `s t ip1-webseald-ip1 list`.

The following is the expected output:

```
/
```

4. Enter `s t ip1-webseald-ip1 create -t ssl -c iv-creds -b supply -h tip_hostname/ip -p tip_admin_console_secure_port /tip`.

Where:

```
s t = server task
ip1-webseald-ip1 = WebSEAL instance name
-t ssl = transport type is SSL
-c iv-creds = needed for single sign on (SSO) to work, carry credential
of user
-b supply = basic authorization header needed for SSO to work
```

The following is the expected output:

```
Created junction at /tip
```

Note: If you want to delete a junction, enter `s t ip1-webseald-ip1 delete /tip`.

Note: If you want to show details for a junction, enter `s t ip1-webseald-ip1 show /tip`.

What to do next

Create a WebSEAL junction mapping table.

Creating a WebSEAL junction mapping table

A junction mapping table maps specific target resources to junction names. Junction mapping is an alternative to a cookie-based solution for filtering dynamically generated server-relative URLs.

To create a WebSEAL junction mapping table:

1. On the Tivoli Access Manager computer, in a text editor open the WebSEAL configuration file, `/opt/pdweb/etc/webseald-ip1.conf`.
2. In the `[junction]` section, edit the `jmt-map` path so that it reads `jmt-map = lib/jmt.conf`.

Note: This path is relative to the server root path. Check the server root path in the `[server]` section of the file and take a note of the full `jmt-map` path. For example, `/opt/pdweb/www-ip1/lib/jmt.conf`.

3. In a text editor create or edit open the `jmt.conf` file and add or modify the following:

- `/tip /ibm/console/*`

Note: The `/ibm/console/` element of the path shown assumes that the Tivoli Integrated Portal root context path was not reconfigured at installation time.

- `/tip /ibm/sla/*`
- `/tip /TCR/reports/*`

4. To load the `jmt.conf` file into WebSEAL, enter `s t ip1-webseald-ip1 jmt load`. The following is the expected output:

```
DPWWM1462I JMT Table successfully loaded
```

5. To restart the WebSEAL server, enter `pdweb restart`.

The following is the expected output:

```
Stopping the: webseald-ip1
Starting the: webseald-ip1
```

What to do next

Test the WebSEAL junction.

Testing the WebSEAL junction

Once you have created a WebSEAL junction, you can test it.

To test a WebSEAL junction:

1. In your Web browser's address bar, enter `https://tam_server_hostname/tip/ibm/console`, where `tip` is the name of the WebSEAL junction. The Tivoli Integrated Portal login page is displayed.
2. To test if Tivoli Access Manager challenges you when you try to access the Tivoli Integrated Portal:
 - a. Close all instances of your Web browser.
 - b. Start your Web browser and go to `https://tam_server_hostname/tip/ibm/console/`.

Note: The `/ibm/console/` element of the URL shown assumes that the Tivoli Integrated Portal root context path was not reconfigured at installation time.

If the WebSEAL junction is working as expected, an Authentication

Required dialog is displayed and you have to provide Tivoli Access Manager account (sec_master) details to proceed.

What to do next

Edit `customizationProperties.xml` to ensure that when you log out of Tivoli Integrated Portal that you also log out from Tivoli Access Manager.

Configuring single sign off for Tivoli Access Manager and Tivoli Integrated Portal

To ensure that you when you log out from the Tivoli Integrated Portal that you also log out from Tivoli Access Manager, you must edit `customizationProperties.xml`.

To configure single sign off for the Tivoli Integrated Portal Server and the Tivoli Access Manager computer:

1. In a text editor, open `install_dir/profiles/TIPProfile/config/cells/TIPCell/applications/isclite.ear/deployments/isclite/isclite.war/WEB-INF/customizationProperties.xml`.

Windows

For example: `C:\IBM\tivoli\tip1112\profiles\TIPProfile\config\cells\TIPCell\applications\isclite.ear\deployments\isclite\isclite.war\WEB-INF\customizationProperties.xml`

2. Edit the `TAMJunctionName` property, as follows:

```
<consoleproperties:console-property id="TAMJunctionName" value="tip"/>
<consoleproperties:console-property id="WebSealServerName" value=""/>
```

Where:

- `TAMJunctionName` is the junction name in Tivoli Access Manager that is configured to point at the Tivoli Integrated Portal Server.
- `WebSealServerName` is a Tivoli Access Manager WebSEAL server instance name. This property allows the Tivoli Integrated Portal Server process requests from declared WebSEAL hosts.

Results

When you log out from the Tivoli Integrated Portal, a Successful Logout message is displayed in your browser. This indicates that you logged out from both the Tivoli Integrated Portal and Tivoli Access Manager.

Setting form-based authentication for WebSEAL

Tivoli Access Manager provides form-based authentication as an optional alternative to the standard Basic Authentication mechanism.

For information on WebSEAL authentication and changing from basic mode to the form-based mode refer to Tivoli Access Manager documentation at http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc_6.1/am61_webservers_admin74.htm#chpt4_amwebpi_authent:

Setting up the Web GUI for productive usage

After you have optionally configured the user authentication for your Web GUI installation, and configured your encryption settings, you can configure the Web GUI for use in your environment, for example by defining additional data sources, setting up a load balancing environment, or creating launch-in-context integrations with other Tivoli products.

Changing data source configurations

If you want to retrieve events from multiple data sources or failover pairs, or to set up a dual server desktop (DSD) environment, you must configure the Web GUI to connect to these data sources.

If, during installation, you selected the ObjectServer as the user registry, the VMM plug-in contains the same configuration as for the ObjectServer specified as the data source.

For more information about how to configure multiple data sources or a DSD environment, see the sample XML files at the following location:

webgui_home_dir/etc/datasource/samples

Related reference

Appendix E, “Web GUI initialization file properties,” on page 633

Data source configuration file overview

The parameters controlling data source location, connection, failover, and cache cleanup are stored in the *ncwDataSourceDefinitions.xml* data source configuration file.

If you installed the Web GUI for the first time, by default this file contains the information you provided during installation. If you upgraded from IBM Tivoli Netcool/Webtop V2.2, the *ncwDataSourceDefinitions.xml* file is migrated to V7.3. If you upgraded from an version of IBM Tivoli Netcool/Webtop earlier than V2.2, this file contains entries derived from any data source properties discovered in the *webgui_home_dir/etc/server.init* file.

During an upgrade from all versions, the properties are migrated to be compatible with the Tivoli Netcool/OMNIBus V7.3 Web GUI.

Sample data source configuration file

The following example is a data source configuration file for a single failover pair.

Tip: For further samples, see the files at *webgui_home_dir/etc/datasources/samples*.

```
[1] <ncwDataSourceDefinitions>
[2]   <ncwDefaultDataSourceList>
[3]     <ncwDataSourceEntry name="NCOMS"/>
[4]   </ncwDefaultDataSourceList>
[5]   <ncwDataSourceDefinition type="singleServerOSDataSource" name="NCOMS">
[6]     <results-cache>
[7]       <chart maxAge="60" enabled="false" cleantime="120"/>
[8]       <config maxAge="3600"/>
[9]       <eventList maxAge="60" enabled="false" cleantime="120"/>
[10]      <eventSummary maxAge="10" enabled="true" cleantime="20"/>
[11]    <metric maxAge="10" enabled="true" cleantime="20"/>
[12]  </results-cache>
```

```

[13]     <ncwDataSourcePollingParameters>
[14]         <ncwFailOverPollingParameters backOffMultiplier="2"
basePollingTime="10"/>
[15]         <ncwHeartBeatParameters basePollingTime="15"/>
[16]     </ncwDataSourcePollingParameters>
[17]     <ncwConnectionParameters
[18]         <ncwStatementParameters
[19]             <ncwQueryTimeout baseTime="60"/>
[20]         </ncwStatementParameters>
[21]     </ncwConnectionParameters>
[22]     <ncwDataSourceCredentials password="" userName="root"
encrypted="false"/>
[23]     <ncwFailOverPairDefinition>
[24]         <ncwPrimaryServer>
[25]             <ncwOSConnection host="192.168.0.1" port="4100"/>
[26]         </ncwPrimaryServer>
[27]     </ncwFailOverPairDefinition>
[28] </ncwDataSourceDefinition>
[29] </ncwDataSourceDefinitions>

```

Explanation of sample data source configuration file

The following description explains how the code works line by line.

“Line 1”

“Lines 2-4”

“Line 5” on page 514

“Lines 6-12” on page 514

“Lines 13-16” on page 515

“Lines 17-21” on page 515

“Line 22” on page 516

“Lines 23-27” on page 516

“Line 28” on page 516

“Line 29” on page 516

Line 1

This line opens the top-level `<ncwDataSourceDefinitions>` element and initiates the file.

Lines 2-4

The `<ncwDefaultDataSourceList>` element contains a list of one or more `<ncwDataSourceEntry>` elements. The first `<ncwDataSourceEntry>` is the default data source, which provides the Web GUI server with a definitive point of configuration and display information when multiple data sources are used.

Note: The configuration file must contain at least one `<ncwDataSourceEntry>` element, and the data source named in this element must correspond to one defined by a `<ncwDataSourceDefinition>` element elsewhere in the configuration file.

The application looks to the data source cited in this element for the following information:

- The default data source for administrator login.
- The default ObjectServer entry in Administration page menus.
- The default data source for chart data.
- The default data source for SmartPage commands.

If communications with the default data source cannot be established, Web GUI uses the next `<ncwDataSourceEntry>` element in the list (if one is present) as the default data source for server activities. Server-validated user passwords and administrator passwords can differ between data sources.

Line 5

The `<ncwDataSourceDefinition>` element encapsulates all the configuration and communication parameters for an individual data source. The configuration file must contain at least one `<ncwDataSourceDefinition>` element.

The `name` attribute specifies a user-defined label. When different ObjectServers have similar or identical names, this attribute is used to distinguish between them. The label is also referred to by the `<ncwDataSourceEntry>` element. The `name` attribute must be a unique alphanumeric string that does not contain any spaces or special characters.

The `type` attribute specifies whether you want this data source to be a single ObjectServer and optional backup server, or a dual-server desktop (DSD) configuration, with an ObjectServer that has one or more display servers and optional backup server. For more information about the code required for a DSD configuration, see “Additional code for DSD configuration” on page 516.

Lines 6-12

The `<results-cache>` element specifies whether data source data caching is enabled. If you configure the Web GUI for use with display servers, this element is ignored, because data caching is not supported in dual-server desktop configurations. This element has the following child elements:

`<chart>`

Defines caching for chart results. This element has the following attributes:

enabled

Set this attribute to TRUE to enable caching.

maxAge

Specifies the expiry time, in seconds, for the cache.

cleantime

Specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the `maxAge` attribute is removed.

`<config>`

Defines caching for data source configuration data, for example event severity colors and conversions. This element has the following attribute:

maxAge

Specifies the expiry time, in seconds, for the cache.

`<eventList>`

Defines caching for results in the event lists. This element has the following attributes:

enabled

Set this attribute to TRUE to enable caching.

maxAge

Specifies the expiry time, in seconds, for the cache.

cleantime

Specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<eventSummary>

Defines caching for event summary results, such as maps and Event Dashboards. This element has the following attributes:

enabled

Set this attribute to TRUE to enable caching.

maxAge

Specifies the expiry time, in seconds, for the cache.

cleantime

Specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<metric>

Defines caching for metric results, that is, metric gauges. This element has the following attributes:

enabled

Set this attribute to TRUE to enable caching.

maxAge

Specifies the expiry time, in seconds, for the cache.

cleantime

Specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

Lines 13-16

The <ncwDataSourcePollingParameters> element contains the elements that control failover and data source heartbeat polling.

The basePollingTime attribute of the <ncwFailOverPollingParameters> element specifies the time interval at which the data source is polled if a failover occurs. The backOffMultiplier attribute contains the multiplier for the algorithm used by the Web GUI to calculate the polling backoff time during a failover. This example is not configured for failover (as shown in lines 14-18) so this property is ignored.

The basePollingTime attribute of the <ncwHeartBeatParameters> element on line 10 specifies the time interval for the Web GUI to poll an active data source.

Lines 17-21

The <ncwConnectionParameters> element specifies the connection parameters for the data source. The baseTime attribute of the <ncwQueryTimeout> element specifies the time, in seconds, before a query sent to the data source times out. If a query times out, the Web GUI attempts to reconnect to the data source.

Line 22

The `<ncwDataSourceCredentials>` element holds the login information required by the Web GUI to access the data source. If the encrypted attribute is set to TRUE, a password encrypted using the Tivoli Netcool/OMNIbus **nco_g_crypt** encrypt utility can be used.

By default, the `userName` attribute and the `password` attribute are not encrypted in the configuration file. For security, set the file permissions for the `ncwDataSourceDefinitions.xml` file to restrict access to the users who are responsible for administering the Web GUI server.

Lines 23-27

The `<ncwFailOverPairDefinition>` element contains an `<ncwPrimaryServer>` element, and, optionally an `<ncwBackUpServer>` element. These elements respectively provide the Web GUI with the IP address or host name, and port number of the primary data source and failover server. If a failover pair is configured, an `<ncwBackUpServer>` element is added to the `<ncwFailOverPairDefinition>` element. The `<ncwBackUpServer>` element has a child element `<ncwOSConnection>` that specifies the host name and port number of the failover server. For example:

```
<ncwBackUpServer>  
  <ncwOSConnection host="host" port="port"/>  
</ncwBackUpServer>
```

The fields contained within the `alerts.status` table and the authenticated users of the failover ObjectServer must exactly match those held on the primary ObjectServer. If the fields do not correspond, failover cannot take place. If the users do not correspond, users might not be able to log in. Before designating a failover server, ensure that the user names and field names are identical to those on the primary server, and that an equal number of ObjectServer fields are present.

The Web GUI regularly communicates with the backup ObjectServer so that in the event of failover it can immediately switch connections. This constant polling is called a hot-standby connection. When a failover event takes place, the Web GUI becomes aware that it is connected to the backup ObjectServer in a failover pair. The application automatically checks for the recovery of the primary ObjectServer in the failover pair and switches back after it has recovered.

Line 28

This line contains the element that closes the `<ncwDataSourceDefinition>` element. If you want to add additional data sources, then you must open a new `<ncwDataSourceDefinition>` element after this entry.

Line 29

This line closes the `<ncwDataSourceDefinitions>` element and concludes the file.

Additional code for DSD configuration

If an ObjectServer is configured for a DSD environment, the following code defines the display ObjectServer or ObjectServers. This code is inserted after the closing `</ncwFailOverPairDefinition>` element (line 26).


```
<ncwReadCloudDefinition>
    <ncwOSConnection host="host" port="port" />
</ncwReadCloudDefinition>
```

Use one <ncwOSConnection> element for each display ObjectServer. An ObjectServer cannot have more than one <ncwReadCloudDefinition> element.

Related concepts

"How IBM Tivoli Netcool/Webtop features are migrated to the Tivoli Netcool/OMNIbus Web GUI" on page 186

Related tasks

"Configuring multiple data sources"

"Configuring a dual-server desktop environment" on page 518

Related reference

"Data source configuration file data reference" on page 519

Configuring multiple data sources

To retrieve events from more than one data source or failover pair, you must add the additional data sources to the data source configuration file.

By default, the `ncwDataSourceDefinitions.xml` file contains the data source or failover pair that you specified during installation. After you have edited this file, you must restart the server for the changes to take effect.

If you want use multiple data sources or failover pairs, make sure that all data source contain consistent field definitions and consistent users, groups and permissions.

To configure the Web GUI for multiple data sources:

1. Open the `ncwDataSourceDefinitions.xml` file.
2. Locate the <ncwDefaultDataSourceList> element.
3. To add a data source, add a <ncwDataSourceEntry> element as a child of <ncwDefaultDataSourceList>.

Tip: For IPv6 networks, use host names instead of literal addresses. The first data source defined in the <ncwDefaultDataSourceList> element is the default data source.

A data source configuration file with two defined data sources is shown in the following example:

```
<ncwDefaultDataSourceList>
  <ncwDataSourceEntry name="NCOMS" />
  <ncwDataSourceEntry name="NILKA" />
</ncwDefaultDataSourceList>
```

Where *NCOMS* is the name of the data source defined during installation, and *NILKA* is the name of the additional data source.

4. To define the data source added in step 3:
 - a. Add an new <ncwDataSourceDefinition> element, plus child elements.
 - b. Set the type attribute of the <ncwDataSourceDefinition> element to "singleServerOSDataSource".
 - c. To define the additional data source as a failover pair, define the back up data source by adding the following code beneath the closing </ncwPrimaryServer> element:

```
<ncwBackUpServer>
  <ncwOSConnection host="host" port="port" />
</ncwBackUpServer>
```

Where *host* is the host name of the backup ObjectServer and *port* is the port number.

5. Save and close the file.
6. Restart the server.

Related tasks

"Restarting the server" on page 568

Related reference

"Data source configuration file data reference" on page 519

"Data source configuration file overview" on page 512

Configuring a dual-server desktop environment

Dual-server desktop (DSD) is a tiered event processing architecture. Web GUI installations that use a DSD architecture write to one or more display servers and to a single master ObjectServer simultaneously, but read event data only from the display servers. To set up a DSD environment, you must modify the data source configuration file.

The DSD architecture increases the performance of an ObjectServer that frequently experiences heavy loads. For example, if many ObjectServers send alerts to a central ObjectServer through unidirectional gateways, or if many users connect directly to the central ObjectServer either through the Web GUI or the native Tivoli Netcool/OMNIbus desktop. A DSD ObjectServer configuration reduces the workload of the central ObjectServer by deferring the load to display ObjectServers. From the perspective of a Web GUI client, there is no difference between being connected directly to the central ObjectServer or to a display ObjectServer. User actions on the Web GUI are sent to both central and display ObjectServers at the same time.

The display server for pages that contain Active Event Lists (AELs), Lightweight Event Lists (LELs), maps, or Table Views is static for the duration of the session, and is selected when a user logs in.

Restriction: Absolute data harmonization between two or more display servers is not possible. The greater the granularity and scale of load upon the display servers, the greater the potential for event disparity during the load-balancing cycle. However, this is highly unlikely to be observed. This does not apply to AELs and LELs, because only a single display server is used during an AEL or LEL session.

To set up a DSD environment:

1. Open the `ncwDataSourceDefinitions.xml` file.
2. Locate the `<ncwDefaultDataSourceList>` element.
3. Add all the required data sources as `<ncwDataSourceEntry>` elements as children of `<ncwDefaultDataSourceList>`.

Tip: For IPv6 networks, use host names instead of literal addresses. The first data source defined in the `<ncwDefaultDataSourceList>` element is the default data source.

For example:

```
<ncwDefaultDataSourceList>
  <ncwDataSourceEntry name="defaultdatasource1"/>
  <ncwDataSourceEntry name="datasource2"/>
  <ncwDataSourceEntry name="datasource3"/>
</ncwDefaultDataSourceList>
```

4. Save and close the file.
5. Define the data sources added in step “Configuring a dual-server desktop environment” on page 518 by adding `<ncwDataSourceDefinition>` elements, plus child elements, for each data source.
6. To define the additional data source as a failover pair, define the back up data source by adding the following code beneath the closing `</ncwPrimaryServer>` element:

```
<ncwBackUpServer>
  <ncwOSConnection host="host" port="port"/>
</ncwBackUpServer>
```

Where *host* is the host name of the backup ObjectServer and *port* is the port number.

7. To configure a data source for DSD, perform the following steps:
 - a. Set the type attribute of the `<ncwDataSourceDefinition>` element to “multipleServerOSDataSource”.
 - b. Define the display servers by adding the `<ncwReadCloudDefinition>` element beneath the closing `</ncwFailOverPairDefinition>` element, in which you define the host and port of the display servers. In the `<ncwReadCloudDefinition>` element, define each display server in an `<ncwOSConnection>` element. For example:

```
<ncwReadCloudDefinition>
  <ncwOSConnection host="192.168.0.9" port="4747"/>
  <ncwOSConnection host="192.168.0.10" port="4848"/>
  <ncwOSConnection host="192.168.0.11" port="4949"/>
</ncwReadCloudDefinition>
```

One `<ncwReadCloudDefinition>` element is permitted per data source; you cannot have multiple display server clouds communicating with a single master ObjectServer.

8. Save and close the file.
9. Restart the server.

Related tasks

“Restarting the server” on page 568

Related reference

“Data source configuration file data reference”

“Data source configuration file overview” on page 512

Data source configuration file data reference

Web GUI communication configuration files must conform in structure to the content described by a Document Type Definition (DTD).

XML overview:

The Extensible Markup Language (XML) is a standard, self-describing set of rules for structuring data so that it can be processed and exchanged across a variety of hardware types, operating systems, and applications.

A reasonable degree of XML knowledge on the part of the Web GUI administrator is assumed.

To write your own data source configuration files you must understand:

- The rules, logic, and components used by XML
- The concepts of *elements*, *attributes*, and *markup*
- How to create documents that are well-formed, and valid against an XML Document Type Definition (DTD)

DTD reference:

XML is hierarchical in structure and the DTD specifies whether each element permits *child elements*, that is, whether other elements can be used under an element within the hierarchy.

Attention: XML is case-sensitive. Elements, attributes, and attribute values used within an XML command file must be used exactly as they are shown in the DTD.

Data types and legends

The data types and legends that accompany the Web GUI DTD elements and attributes are as follows:

NM Indicates that the attribute types are names consisting of XML NMTOKEN character (letters, periods, numbers, underscores, dashes, and colons). NM often also indicates that the attribute contains a list of predefined choices.

CDATA

Indicates that the attribute contains unparsed character data.

IMP Indicates that the presence of the attribute is implied (optional).

REQ Indicates that the presence of the attribute is required.

Attributes and elements of the DTD

The XML elements and attributes defined in the Web GUI configuration DTD as follows:

<chart>

This element has the following attributes:

- maxAge (type: CDATA, presence: IMP)
- enabled (type: NM, presence: IMP)
- cleantime (type: CDATA, presence: IMP)

This element has no child elements.

<config>

This element has the following attributes:

- maxAge (type: CDATA, presence: IMP)

This element has no child elements.

<eventList>

This element has the following attributes:

- maxAge (type: CDATA, presence: IMP)
- enabled (type: NM, presence: IMP)
- cleantime (type: CDATA, presence: IMP)

This element has no child elements.

<eventSummary>

This element has the following attributes:

- maxAge (type: CDATA, presence: IMP)
- enabled (type: NM, presence: IMP)
- cleantime (type: CDATA, presence: IMP)

This element has no child elements.

<metric>

This element has the following attributes:

- maxAge (type: CDATA, presence: IMP)
- enabled (type: NM, presence: IMP)
- cleantime (type: CDATA, presence: IMP)

This element has no child elements.

<ncwBackUpServer>

This element has no attributes. This element has the following child elements:

- <ncwOSConnection>

ncwConnectionParameters

This element has no attributes. This element has the following child elements:

- <ncwstatementParameters> There are zero or one occurrences of this element.

<ncwDataSourceCredentials>

This element has the following attributes:

- userName (type: CCDATA, presence: IMP)
- password (type: CCDATA, presence: IMP)
- encrypted (type: NM, presence: IMP)
- algorithm (type: NM, presence: IMP)

This element has no child elements.

<ncwDataSourceDefinition>

This element has the following attributes:

- type (type: NM, presence: IMP)
- name (type: CDATA, presence: REQ)

This element has the following child elements:

- <results-cache>
- <ncwDataSourcePollingParameters>
- <ncwConnectionParameters> There are zero or one occurrences of this element.
- <ncwDataSourceCredentials>
- <ncwFailOverPairDefinition>

- <ncwReadCloudDefinition> There are zero or one occurrences of this element.

<ncwDataSourceDefinitions>

This is the root element and has no attributes. This element has the following child elements:

- <ncwDefaultDataSourceList>
- <ncwDataSourceDefinition> There is at least one occurrence of this element.

<ncwDataSourceEntry>

This element has the following attributes:

- name (type: CDATA, presence: REQ)

This element has no child elements.

<ncwDataSourcePollingParameters>

This element has no attributes. This element has the following child elements:

- <ncwFailOverPollingParameters>
- <ncwHeartBeatParameters>

<ncwDefaultDataSourceList>

This element has no attributes. This element has the following child elements:

- <ncwDataSourceEntry> There is at least one occurrence of this element.

<ncwFailOverPairDefinition>

This element has no attributes. This element has the following child elements:

- <ncwPrimaryServer>
- <ncwBackUpServer> There are zero or one occurrences of this element.

<ncwFailOverPollingParameters>

This element has the following attributes:

- backOffMultiplier (type: CDATA, presence: IMP)
- basePollingTime (type: CDATA, presence: IMP)

This element has no child elements.

<ncwHeartBeatParameters>

This element has the following attributes:

- basePollingTime (type: CDATA, presence: IMP)

This element has no child elements.

<ncwOSConnection>

This element has the following attributes:

- host (type: CDATA, presence: REQ)
- port (type: CDATA, presence: IMP)
- ssl (type: NM, presence: IMP)
- minPoolSize (type: CDATA, presence: IMP)
- maxPoolSize (type: CDATA, presence: IMP)

<ncwPrimaryServer>

This element has no attributes. This element has the following child elements:

- <ncwOSConnection>

<ncwQueryTimeout>

This element has the following attributes:

- baseTime (type: CDATA, presence: IMP)

This element has no child elements.

<ncwReadCloudDefinition>

This element has no attributes. This element has the following child elements:

- <ncwOSConnection> There is at least one occurrence of this element.

<ncwStatementParameters>

This element has no attributes. This element has the following child elements:

- <ncwQueryTimeout> There are zero or one occurrences of this element.

<results-cache>

This element has no attributes. This element has the following child elements:

- <chart>
- <config>
- <eventList>
- <eventSummary>
- <metric>

Element reference:

The elements used in the Web GUI configuration DTD often have one or more associated attributes, for which a value can be required.

The elements defined within the configuration DTD are as follows.

<chart>

This element is a child element of the <results-cache> element. This element specifies caching options for chart results. If caching is enabled, the maxAge attribute specifies the expiry time, in seconds, for the cache. The cleantime attribute specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<config>

This element is a child element of the <results-cache> element. This element specifies whether data caching is enabled. If caching is enabled, the maxAge attribute specifies the expiry time, in seconds, for the cache. For example:

```
<config maxAge="60" enabled="true">
```

<eventList>

This element is a child element of the <results-cache> element. This element specifies caching for results in the event lists. If caching is enabled, the maxAge attribute specifies the expiry time, in seconds, for the cache. The cleantime attribute specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<eventSummary>

This element is a child element of the <results-cache> element. This element specifies caching for event summary results, such as maps and Event Dashboards. If caching is enabled, the maxAge attribute specifies the expiry

time, in seconds, for the cache. The cleantime attribute specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<metric>

This element is a child element of the <results-cache> element. This element specifies caching for results in Gauges pages. If caching is enabled, the maxAge attribute specifies the expiry time, in seconds, for the cache. The cleantime attribute specifies the time interval, in seconds, at which cache entries are checked and removed. Cache data that exceeds the time imposed by the maxAge attribute is removed.

<ncwBackUpServer>

This element is a child element of <ncwDefaultDataSourceList> and contains the ncwOSConnection element specifying host and port of the failover ObjectServer. For example:

```
<ncwBackUpServer>
  <ncwOSConnection
    host="192.168.0.3"
    port="4141"
  />
</ncwBackUpServer>
```

<ncwConnectionParameters>

This element is a child element of <ncwDataSourceDefinition> and contains elements that control the connection to a data source.

<ncwDataSourceCredentials>

This element is a child element of <ncwDataSourceDefinition> and holds the login information required by the Web GUI to access the data source. If the encrypted attribute is set to true, a password encrypted using the Tivoli Netcool/OMNIBus **ncg_crypt** encryption utility can be used. For example:

```
<ncwDataSourceCredentials
  password=""
  userName="root"
  encrypted="false"
/>
```

<ncwDataSourceDefinition>

This element is a child element of the <ncwDataSourceDefinitions> element and contains the tags that define configuration and communication parameters for an individual data source.

<ncwDataSourceDefinitions>

This is the root element of the DTD.

<ncwDataSourceEntry>

This element is a child element of <ncwDefaultDataSourceList> and contains the names of the default data sources that communicate with the Web GUI. These entries are subsequently defined in the configuration file by corresponding <ncwDataSourceDefinition> tags. The first entry in the list is the default data source used by the Web GUI for client authentication. If this data source is not present, the next entry in the list is used as a default. For example:

```
<ncwDefaultDataSourceList>
  <ncwDataSourceEntry name="NCOMS"/>
  <ncwDataSourceEntry name="NILKA"/>
</ncwDefaultDataSourceList>
```


<ncwDataSourcePollingParameters>

This element is a child element of <ncwDataSourceDefinition> and contains the elements that control failover and data source heartbeat polling.

<ncwDefaultDataSourceList>

See <ncwDataSourceEntry>.

<ncwFailOverPairDefinition>

This element is a child element of <ncwDataSourceDefinition> and contains the tags that specify the primary and backup ObjectServers. The inclusion of a backup ObjectServer is optional, but only one is permitted per data source. For example:

```
<ncwFailOverPairDefinition>
  <ncwPrimaryServer>
    <ncwOSConnection
      host="192.168.0.7"
      port="4545"
    />
  </ncwPrimaryServer>
  <ncwBackUpServer>
    <ncwOSConnection
      host="192.168.0.8"
      port="4646"
    />
  </ncwBackUpServer>
</ncwFailOverPairDefinition>
```

<ncwFailOverPollingParameters>

This element specifies the time interval at which the data source is polled in the event of a failover. This element is used only when there is a failover server available, as defined by the <ncwBackUpServer> element. For example:

```
<ncwFailOverPollingParameters backOffMultiplier="2" basePollingTime="10"/>
```

<ncwHeartBeatParameters>

This element is a child element of <ncwDataSourcePollingParameters> and specifies the time interval, in seconds, for the Web GUI to poll an active data source. For example:

```
<ncwHeartBeatParameters basePollingTime="15"/>
```

<ncwOSConnection>

This element is a child element of both <ncwPrimaryServer> and <ncwBackUpServer> and specifies the communication criteria for a primary or failover data source. For example:

```
<ncwOSConnection host="192.168.0.3" port="4141"/>
```

<ncwPrimaryServer>

This element is a child element of <ncwDefaultDataSourceList> and contains the ncwOSConnection element specifying host and port of the primary ObjectServer. For example:

```
<ncwPrimaryServer>
  <ncwOSConnection
    host="192.168.0.3"
    port="4141"
  />
</ncwPrimaryServer>
```

<ncwQueryTimeout>

This element is a child element of <ncwStatementParameters> and defines the time out period, in seconds, for SQL statements sent to a data source. For example:

```
<ncwQueryTimeout baseTime="60" />
```

<ncwReadCloudDefinition>

This element is a child element of <ncwDataSourceDefinition> and holds the addresses of all the display servers you want to use with this master ObjectServer. One <ncwReadCloudDefinition> element permitted per data source. You cannot have multiple display server clouds communicating with a single master ObjectServer. For example:

```
<ncwReadCloudDefinition>
  <ncwOSConnection
    host="192.168.0.9"
    port="4747"
  />
  <ncwOSConnection
    host="192.168.0.10"
    port="4848"
  />
  <ncwOSConnection
    host="192.168.0.11"
    port="4949"
  />
</ncwReadCloudDefinition>
```

<ncwStatementParameters>

This element is a child element of <ncwConnectionParameters> and contains elements that control the exchange of SQL statements with a data source.

<results-cache>

The <results-cache> element is a child element of the <ncwDataSourceDefinition> element. It contains the child elements <chart>, <config>, <eventList>, <eventSummary>, and <metric>.

Attribute reference:

Use this information to understand the attributes used in the Web GUI configuration DTD. Some attributes are enumerated and the values of these attributes are constrained to a list of predefined text strings. When enumerated attributes are used within the XML command file, they must use one of the values shown in the list.

The following table describes each attribute defined within the configuration DTD. Default values (if any) are provided in the description.

Table 101. Configuration DTD attribute definitions

Attribute	Constrained values	Description
algorithm	DES AES	Specifies whether a DES or an AES algorithm is used.
backOffMultiplier	None	The multiplier for the backoff algorithm used to calculate the polling backoff time during a failover. The default value is 1.
basePollingTime	None	The seed time, in seconds, for the algorithm used to calculate the polling backoff time during a failover. The default value is 20 seconds for the <ncwFailoverPollingParameters> element or 15 seconds for the <ncwHeartbeatParameters> element.

Table 101. Configuration DTD attribute definitions (continued)

Attribute	Constrained values	Description
baseTime	None	<p>The timeout period, in seconds, for a query statement sent to the data source. If the Web GUI receives no response within this time, it attempts to reconnect to the data source.</p> <p>The default value is 30 seconds.</p>
cleantime	None	<p>The time interval, in seconds, the Web GUI server waits before checking for how long each user session has been inactive.</p> <p>When this check takes place, cache data that exceeds the time imposed by the maxAge attribute is removed.</p> <p>The default value is 120 seconds for the <chart> and <eventList> elements or 20 seconds for the <eventSummary> and <metric> elements.</p>
enabled	true false	<p>Specifies if page caching is turned on or off.</p> <p>The default value is true for the <ncwDataSourceDefinition>, <eventSummary>, and <metric> elements or false for the <chart> and <eventList> elements.</p>
encrypted	true false	<p>Specifies whether the user password is encrypted.</p> <p>The default value is false.</p>
host	None	<p>The host name or IP address of a specified data source.</p>
maxAge	None	<p>The cache expiry time limit in seconds.</p> <p>The default value is 10 seconds for the <eventSummary> and <metric> elements, 60 seconds for the <chart> and <eventList> elements, or 3600 seconds for the <config> element.</p>
maxPoolSize	Maximum value: 1024	<p>The maximum number of pooled connections to an ObjectServer data source that can exist at any one time.</p> <p>The default value is 10.</p>
minPoolSize	None	<p>The minimum number of pooled connections to an ObjectServer data source to maintain.</p> <p>The default value is 5.</p>

Table 101. Configuration DTD attribute definitions (continued)

Attribute	Constrained values	Description
name	None	<p>The name given to an ObjectServer data source displayed within the Web GUI during administrative activities.</p> <p>This value also links each data source definition that is listed at the start of the configuration file to its subsequent definition.</p>
password	None	<p>The password used to log in to the ObjectServer.</p> <p>The default is a blank password.</p>
port	None	<p>The port number of a specified data source.</p> <p>The default value is 8080.</p>
ssl	true false	<p>Specifies whether to use a SSL connection to an ObjectServer.</p> <p>The default value is false.</p>
type	singleServerOS DataSource multipleServerOS DataSource	<p>The type of data source configuration required for the specified data source. The required types are as follows:</p> <p>singleServerOSDataSource Use this type for a single primary data source configuration, or for a backup data source configuration.</p> <p>multipleServerOSDataSource Use this type for a dual-server desktop configuration.</p> <p>The default value is singleServerOSDataSource.</p>
userName	None	<p>The user name of the user connecting to the ObjectServer. The user must have root privileges on the ObjectServer.</p> <p>The default value is root.</p>

Setting environment variables for charts

On AIX and HP-UX operating systems, set the DISPLAY environment variable properly for Web GUI charts to display correctly.

To make sure that the charts are displayed correctly, set the DISPLAY environment variable to the host running the Windows X-server.

Configuring and maintaining single sign-on

How to set up and maintain the single sign-on (SSO) capability between the Web GUI and other Tivoli products.

Configuring SSO using ESS between multiple servers

How to configure single sign-on (SSO) between multiple servers.

Before you begin

Before configuring single-sign on between a number of servers, they all need to point to a central user registry, such as a Lightweight Directory Access Protocol (LDAP) server.

To configure single sign-on between a number of servers:

1. On the server running the Web GUI:
 - a. Configure SSO.
 - b. Restart the server.
 - c. Export the Lightweight Third Party Authentication (LTPA) keys from WebSphere.
2. On each of the other servers:
 - a. Copy the file of exported keys from the Web GUI server.
 - b. Configure SSO.
 - c. Import the LTPA keys into both WebSphere Application Server and ESS. Then restart the server

Related tasks

“Configuring single sign-on” on page 530

“Exporting the LTPA keys” on page 531

“Importing the LTPA keys” on page 531

“Restarting the server” on page 568

Maintaining the LTPA keys

If the LTPA keys change for WebSphere on the Web GUI server, export the keys and load them into the local ESS. In addition, load them into WebSphere and ESS on all other participating servers.

Since WebSphere and ESS have their own copies of the LTPA keys they need to be kept in synchronization. So should the keys change on the Web GUI server you need to import to that server's ESS component. In addition, import them into WebSphere and ESS for all other servers that cooperate in the single sign-on domain.

When the LTPA keys for WebSphere on the Web GUI change:

1. On the server running the Web GUI :
 - a. Export the LTPA keys from WebSphere.
 - b. Import the keys into ESS.
 - c. Restart the server.
2. On each of the other servers in the SSO domain:
 - a. Copy the files of exported keys from the Web GUI .
 - b. Import the LTPA keys into both WebSphere Application Server and ESS.
 - c. Restart the server.

Related tasks

“Exporting the LTPA keys” on page 531

“Importing the LTPA keys” on page 531

“Restarting the server” on page 568

Supporting procedures for single sign-on

Procedures used to set up and maintain single sign-on and LTPA keys between a number of servers.

Configuring single sign-on:

Use these instructions to establish single sign-on support and configure a federated repository.

Before you begin

Configuring SSO is a prerequisite to integrating products that are deployed on multiple servers. All Tivoli Integrated Portal Server instances must point to the central user registry (such as a Lightweight Directory Access Protocol server).

Attention: ITM single sign on (SSO) support is only available with ITM Version 6.2 Fix Pack 1 or higher.

To configure the WebSphere federated repositories functionality for LDAP:

1. Log in to the administrative console.
2. In the **Authentication** area, expand **Web security** and click **Single sign-on**.
3. Click the **Enabled** option if SSO is disabled.
4. Click **Requires SSL** if all of the requests are expected to use HTTPS.
5. Enter the fully-qualified domain names in the Domain name field where SSO is effective. If the domain name is not fully qualified, the Tivoli Integrated Portal Server does not set a domain name value for the **LtpaToken** cookie and SSO is valid only for the server that created the cookie. For SSO to work across Tivoli applications, their application servers must be installed in same domain (use the same domain name).
6. Optional: Enable the **Interoperability Mode** option if you want to support SSO connections in WebSphere Application Server version 5.1.1 or later to interoperate with previous versions of the application server.
7. Optional: Enable the **Web inbound security attribute propagation** option if you want information added during the login at a specific Tivoli Enterprise Portal Server to propagate to other application server instances.
8. After clicking **OK** to save your changes, stop and restart all the Tivoli Integrated Portal Server instances.

What to do next

Note: When you launch Tivoli Netcool/OMNIBus Web GUI, you must use a URL in the format protocol://host.domain:port /*. If you do not use a fully-qualified domain name, Tivoli Netcool/OMNIBus Web GUI cannot use SSO between Tivoli products.

Related concepts

“Single sign-on” on page 30

Exporting the LTPA keys:

How to export the LTPA keys from the WebSphere Application Server.

To export the LTPA keys from the WebSphere Application Server. to a file:

1. Log in to the Tivoli Integrated Portal server as an administrator.
2. In the navigation tree, click **Settings > WebSphere Administrative Console**.
3. Click **Launch WebSphere administrative console**.
4. Click **Security > Global security**.
5. In the **Authentication** area, click **LTPA**.
6. In the **Cross cell single-sign on** area, enter a password for the file of keys in **Password** and **Confirm password**.
7. Type a name for the file of keys in **Fully qualified key file name**.
8. Click **Export keys**.

The file of keys is created in *tip_home_dir/profiles/TIPProfile* and a confirmation message appears at the top of the administrative console.

9. You can now log out of the Websphere administrative console.

Importing the LTPA keys:

How to import the LTPA keys into WebSphere Application server and ESS. Each component maintains their own copy of the keys and so they must be synchronized.

Importing the keys into WebSphere:

To import LTPA keys into WebSphere:

1. If you have not already done so, copy the file of keys from the server where you exported them to the destination server.
Put the file in *tip_home_dir/profiles/TIPProfile*.
2. Log in to the Tivoli Integrated Portal server as an administrator.
3. In the navigation tree, click **Settings > WebSphere Administrative Console**.
4. Click **Launch WebSphere administrative console**.
5. Click **Security > Global security**.
6. In the **Authentication** area, click **LTPA**.
7. In the **Cross cell single-sign on** area, enter the password for the file of keys in **Password** and **Confirm password**.
8. Type a name for the file of keys in **Fully qualified key file name**.
9. Click **Import keys**.

The file of keys are imported into the WebSphere Application Server and a confirmation message appears at the top of the administrative console window.

10. You can now log out of the Websphere administrative console.

Importing the keys into ESS:

To import the LTPA keys into ESS:

1. In a command window, navigate to *tip_home_dir/profiles/TIPProfile/bin* and enter one of the following commands:

```
UNIX      Linux      ./wsadmin.sh
```

```
Windows   wsadmin.bat
```

2. When prompted, supply the username and password for the Tivoli Integrated Portal administrator (for example, *tipadmin* and *tippass*).
3. At the *wsadmin>* prompt enter the following command:

```
$AdminTask importESSLTPAKeys {-pathname /opt/IBM/tivoli/tip/profiles/  
TIPProfile/key_file_name -password key_file_password}
```

Replace:

key_file_name

with the name of the file of LTPA keys.

key_file_password

with the password for the file of LTPA keys.

4. Exit from *wsadmin* by entering:
quit
5. Restart the Tivoli Integrated Portal server.

Extending the functionality of the Web GUI

Tivoli Netcool/OMNIBus includes resources that can be used to extend the functionality of the Web GUI when Tivoli Netcool/OMNIBus is integrated with other products.

Enabling predictive eventing in the Web GUI

To configure the Web GUI to display predictive events generated in IBM Tivoli Monitoring, copy and run the *predictive_events_web_gui.xml* WAAPI command file, which creates the configuration artifacts required for predictive events.

Before you begin

The following prerequisites must be met:

- You must have configured Tivoli Netcool/OMNIBus as described in “Configuring predictive eventing in your integrated environment” on page 428.
- You must have performed at least the minimum configuration for the Web GUI Administration Application Program Interface (WAAPI) client, and be familiar with how the WAAPI client works.
- If you want to use the predictive eventing tools to open Tivoli Enterprise Portal from a predictive event without having to log in, you must have configured single sign-on between the Web GUI server and the Tivoli Enterprise Portal Server.

When run, the *predictive_events_web_gui.xml* command file creates the following resources for use with predictive events:

- Default global filter
- Default global view
- Tools
- Submenu for the Active Event List (AEL) that contains the tools

- Prompts
- .jsp file, images, and a stylesheet

After you have run the command file, you must add the submenu to an AEL menu, and, in the ShowDetailsInTEP tool, specify the host name and port number on which Tivoli Enterprise Portal is running. If you changed the default port number when you installed Tivoli Enterprise Portal, note the port number because it is needed for step 5.

For more information about the WAAPI client, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

To configure the Web GUI for predictive eventing:

1. On the Tivoli Netcool/OMNIBus server, copy the contents of the `$NCHOME/omnibus/extensions/itmpredictive` directory to the following location on the Web GUI server:
`webgui_home_dir/waapi/bin`
2. On the Web GUI server, change to the `webgui_home_dir/waapi/bin` directory.
3. To execute the command file, enter the following command:
`./runwaapi -file predictive_events_web_gui.xml`
4. Add the submenu to an AEL menu:
 - a. Click **Administration > Event Management Tools**, and click **Menu Configuration**.
 - b. From the **Available menus** list, select **alerts** and click **Modify**.
 - c. In the Menus Editor window, select **menu** from the **Available items** list.
 - d. Select the **Predictive Events** submenu and click **Add selected item**.
The submenu is added to the list in the **Current items** pane on the right side of the page.
 - e. Click **Save**.
5. Configure the ShowDetailsInTEP tool to open Tivoli Enterprise Portal:
 - a. Click **Administration > Event Management Tools > Tool Creation**.
 - b. Select **ShowDetailsInTEP**.
The tool is displayed with the following entry in the **URL** field:
`http://[teps_host]:15200/LICServletWeb/LICServlet`
 - c. Replace `teps_host` with the host name on which Tivoli Enterprise Portal is installed and optionally replace the port.
 - d. Click **Save**.

Results

You have now configured predictive eventing, and operators can use the predictive eventing tools in the AEL. For more information about how to use the predictive eventing tools, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Related concepts

“Single sign-on” on page 30

Related tasks

“Setting up the WAAPI client” on page 212

“Configuring single sign-on” on page 530

Enabling support for TADDM events in the Web GUI

You can add a menu, tools, and a filter for TADDM events to the Web GUI server to enable you to view further details about these events when displayed in the Active Event List.

Before you begin

The following prerequisites must be met:

- You must have set up integration between Tivoli Netcool/OMNIbus and TADDM, as described in “Configuring support for TADDM events in your integrated environment” on page 438.
- You must have configured the Web GUI Administration Application Program Interface (WAAPI) client with the appropriate property settings in the `webgui_home_dir/waapi/etc/waapi.init` properties file as described in “Setting up the WAAPI client” on page 212.

You can add the menu, tools, and filter by running a WAAPI command file, which is supplied in the Tivoli Netcool/OMNIbus installation. After running the command file, you must then add the menu as a submenu of the Active Event List **Alerts** menu.

For more information about the WAAPI client, see the *IBM Tivoli Netcool/OMNIbus Web GUI Administration and User's Guide*.

To add the menu and tools for TADDM events to the Web GUI server:

1. From a Tivoli Netcool/OMNIbus host, copy the contents of the `$NCHOME/omnibus/extensions/taddm` directory to the following location where the Web GUI server is installed:

```
webgui_home_dir/waapi/bin
```

2. From the Web GUI server, open a command window and enter the following WAAPI command to add the menu and tools for TADDM events:

```
webgui_home_dir/waapi/bin/runwaapi -file taddm_menutools_web_gui.xml
```

You can now add the new menu as a submenu of the **Alerts** menu, which is used in the Active Event List.

3. From the Web GUI, add a TADDM submenu to the **Alerts** menu, as follows:
 - a. Click **Administration > Event Management Tools > Menu Configuration**.
 - b. Select **alerts** from the list of menus, and click **Modify**.
 - c. From the **Available items** area, select **menu** from the drop-down list. The list of all menu items that can be added to the **Alerts** menu is shown.
 - d. Select the **TADDM** item and click **Add selected item** to move the item to the **Current items** area. You can use the arrow buttons to reposition the **TADDM** item, if required.
 - e. Click **Save** and click **OK** to confirm.

Results

The menu, tools, and filter are now available in the Active Event List for use with TADDM events. For more information about monitoring TADDM events, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Related tasks

“Setting up the WAAPI client” on page 212

Setting up and configuring a load balancing environment

Setting up a cluster of Web GUI servers involves creating a cluster of the underlying Tivoli Integrated Portal servers and then enabling Web GUI load balancing.

Note: Additional software components are required for load balancing: A DB2 database server and the IBM HTTP server. The following sections include instructions for obtaining both of these components.

The steps to set up a load balancing cluster of Web GUI servers are:

1. Ensure that all nodes that will form the cluster meet the load balancing requirements.
2. Download and configure the DB2 database server.
3. Download the IBM HTTP Server.
4. Set up the cluster on one node.
5. Add the remaining nodes to the cluster.
6. Set up server-to-server trust between all nodes in the cluster.
7. Verify the load balancing implementation.
8. Prepare the IBM HTTP Server for load balancing.
9. Set up clone IDs for each node in the cluster.
10. Generate the configuration plug-in for the IBM HTTP server.
11. Configure SSL from each node to the IBM HTTP server.
12. Start Web GUI load balancing on each node in the cluster.

Once the cluster is operational you can:

- Add further nodes to the cluster.
- Remove nodes from the cluster.
- Resynchronize any node with the rest of the cluster.
- Remove the entire cluster.

Related concepts

“Load balancing”

“Federal Information Processing Standard 140–2 (FIPS 140–2) support” on page 26

Related tasks

“Configuring SSL connections in FIPS 140–2 mode for the event feed from the ObjectServer” on page 500

“Encrypting passwords using FIPS 140–2 mode encryption” on page 499

“Enabling FIPS 140–2 mode for the Tivoli Integrated Portal Server” on page 497

“Adding a node to an existing cluster” on page 555

“Removing a node from a cluster” on page 555

Related reference

“FIPS 140–2 configuration checklist” on page 26

Load balancing

You can setup a load balancing cluster of portal nodes with identical configurations to evenly distribute user sessions. Load balancing is ideal for Tivoli Netcool/OMNIBus Web GUI installations with a large user population. When a node within a cluster fails, new user sessions are directed to other active nodes.

Work load is distributed by session, not by request. If a node in the cluster fails, users who are in session with that node must log back in to access the Tivoli Netcool/OMNIBus Web GUI. Any unsaved work is not recovered.

Synchronized data

After load balancing is set up, changes in the console that are stored in global repositories are synchronized to all of the nodes in the cluster using a common database. The following actions cause changes to the global repositories used by the console. Most of these changes are caused by actions in the **Settings** folder in the console navigation.

- Creating, restoring, editing, or deleting a page.
- Creating, restoring, editing, or deleting a view.
- Creating, editing, or deleting a preference profile or deploying preference profiles from the command line.
- Copying a portlet entity or deleting a portlet copy.
- Changing access to a portlet entity, page, external URL, or view.
- Creating, editing, or deleting a role.
- Changes to portlet preferences or defaults.
- Changes from the **Users and Groups** applications, including assigning users and groups to roles.

Note: Global repositories should never be updated manually.

During normal operation within a cluster, updates that require synchronization are first committed to the database. At the same time, the node that submits the update for the global repositories notifies all other nodes in the cluster about the change. As the nodes are notified, they get the updates from the database and commit the change to the local configuration.

If data fails to be committed on any given node, a warning message is logged into the log file. The node is prevented from making its own updates to the database. Restarting the Tivoli Integrated Portal Server instance on the node rectifies most

synchronization issues, if not, the node should be removed from the cluster for corrective action. See *Monitoring a load balancing cluster* for more information.

Note: If the database server restarts, all connections from it to the cluster are lost. It may take up to five minutes for connections to be restored, so that users can again perform update operations, for example, modifying or creating views or pages.

Manual synchronization and maintenance mode

Updates to deploy, redeploy, or remove console modules are not automatically synchronized within the cluster. These changes must be performed manually at each node. For deploy and redeploy operations, the console module package must be identical at each node.

When one of the deployment commands is started on the first node, the system enters *maintenance mode* and changes to the global repositories are locked. After you finish the deployment changes on each of the nodes, the system returns to an unlocked state. There is not any restriction to the order that modules are deployed, removed, or redeployed on each of the nodes.

While in maintenance mode, any attempts to make changes in the portal that affect the global repositories are prevented and an error message is returned. The only changes to global repositories that are allowed are changes to a user's personal portlet preferences. Any changes outside the control of the portal, for example, a form submission in a portlet to a remote application, are processed normally.

The following operations are also not synchronized within the cluster and must be performed manually at each node. These updates do not place the cluster in maintenance mode.

- Deploying, redeploying, and removing wires and transformations
- Customization changes to the console user interface (for example, custom images or style sheets) using `consoleProperties.xml`.

To reduce the chance that users could establish sessions with nodes that have different wire and transformation definitions or user interface customizations, schedule these changes to coincide with console module deployments.

Requirements

The following requirements must be met before load balancing can be enabled.

- Each node is configured to use a central user repository. This can be an ObjectServer or a Lightweight Directory Access Protocol (LDAP) repository.
- A front-end network dispatcher (for example, IBM HTTP Server) must be setup to handle and distribute all incoming session requests. See *Setting up intermediary services* for more information about this task.
- DB2 Version 9.7 must be installed within the network to synchronize the global repositories for the console cluster.
- Each node in the cluster must be enabled to use the same user repository in the same way. For example, they all use the same LDAP using the same user and group configuration.
- All console nodes in load balancing cluster must be installed in the same cell name. After console installation on each node, use the `-cellName` parameter on the `manageprofiles` command.

- All console nodes in load balancing cluster must have synchronized clocks.
- The Websphere application server and Tivoli Integrated Portal Server versions must have the same release level, including any fix packs. Fixes and upgrades for the runtime must be applied manually at each node.

Downloading and installing the DB2 server

To use the Web GUI in a load balancing environment, download and install a DB2 database, into which a Web GUI server pool can be installed.

The Web GUI load balancing environment requires DB2 Enterprise Edition Database V9.7 or later. The license for Tivoli Netcool/OMNIBus contains an entitlement to download, install, and deploy the DB2 database in a load balancing environment.

To download and install the DB2 Database:

1. Download the installation package for your operating system by following the instructions in the document available at the following Web address:

Operating system	Download document location
AIX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026995
HP-UX	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026996
HP-UX Integrity	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026999
Linux	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026997
Linux for System z	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24026998
Solaris	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027000
Windows	http://www-01.ibm.com/support/docview.wss?rs=3120&uid=swg24027001

2. Extract the contents of the installation package into a temporary location.
3. Install and configure the product by following the instructions on the *IBM DB2* information center at the following Web address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp>

What to do next

You must now create and configure the DB2 database for the load balancing environment. To install the Web GUI with the load balancing feature, you need the name of the DB2 database that you created. In addition, enable the database for XML.

Configuring the DB2 database for load balancing

After you have downloaded and installed the DB2 database server, use the scripts provided with the Web GUI installation media to create a database and database tables for Tivoli Integrated Portal. This database is required to install the load balancing feature.

Before you begin

You must have downloaded and installed the DB2 database server, and also downloaded and extracted the Web GUI installation package.

Two scripts are included in the Web GUI installation package to ensure that the DB2 databases are set up correctly for load balancing.

1. Copy the `cdimage/HAScripts` directory from the computer to which you extracted the Web GUI installation media to the computer on which the DB2 database is installed.
2. Log in to the DB2 database as the DB2 administrator.
3. **Windows** Start the DB2 command-line interface by entering `db2cmd` at the command prompt.
4. Change to the `HAScripts` directory.
5. Create a database for the Tivoli Integrated Portal by running the following command:

- **UNIX** **Linux** `./tipMakedb.sh database`
- **Windows** `tipMakedb.bat database`

Where *database* is the name of the Tivoli Integrated Portal database. The default value is `tipdb`.

6. Create the database tables by running the following command:
- **UNIX** **Linux** `./tipdbSchema.sh database`
 - **Windows** `tipdbSchema.bat database`

Where *database* is the name of the Tivoli Integrated Portal database.

What to do next

You can now install the Web GUI with the load balancing feature.

Downloading the HTTP server

To use the Web GUI in a load balancing environment download and, later in the configuration procedure, install the IBM HTTP Server for WebSphere Application Server. A load balancing environment requires the IBM HTTP Server to dispatch requests from Web GUI clients among the nodes in cluster.

To download the IBM HTTP Server:

1. Download the server package appropriate for your operating system from the IBM Passport Advantage site. The part numbers for the IBM HTTP Server are:

Option	Description
Operating system	Part number
AIX 32-bit	1G2NML
AIX 64-bit	1G2NML

Option	Description
HP-UX	C1G2UML
HP-UX Integrity	C1G2XML
Linux 32-bit	C1G33ML
Linux 64-bit	C1G3CML
Linux for System z 32-bit	C1G3RML
Linux for System z 64-bit	C1G3KML
Solaris 32-bit	C1G3FML
Solaris 64-bit (SPARC)	C1G3IML
Solaris 64-bit (AMD)	C1G3LML
Windows 32-bit	C1G2HML
Windows 64-bit	C1G2KML

2. Unpack the download file into a suitable temporary directory.

Setting up a load balancing cluster

You can configure a Tivoli Integrated Portal Server instance to use a database as a file repository instead of a local directory.

Before you begin

Tivoli Integrated Portal is installed on a machine using the cell name designated for all console nodes within the cluster. You have installed and setup a network dispatcher (for example, IBM HTTP Server), DB2, and an LDAP as explained in “Requirements” on page 537.

1. On the machine where DB2 is installed, create a DB2 database (see Creating databases). Make sure you enable the database for XML.
2. Check that you have the JDBC driver for DB2 on the computer where Tivoli Integrated Portal is installed. The JDBC driver should be available at:
tip_home_dir/universalDriver/lib.
3. From a command prompt, change to the *tip_home_dir/profiles/TIPProfile/bin/ha* directory and edit the settings in *tipha.properties*.

Property name	Description
DBHost	The hostname or IP address of the machine where the DB2 database is installed. Example: tipdb.cn.ibm.com
DBPort	Port number of the DB2 server. Example: 50000 (default)
DBName	The name of the database that you created. Example: tipdb
DBProviderClass	Class name of the DB2 provider. Example: com.ibm.db2.jcc.DB2Driver (default)
DBProviderName	Name of the DB2 provider. Example: TIP_Universal_JDBC_Driver (default)
DBDatasource	JNDI name of the datasource. Example: jdbc/tipds Note: For the Web GUI, do not change the value of this property.

Property name	Description
DBDataSourceName	Name of the datasource used for load balancing. Example: tipds Note: For the Web GUI, do not change the value of this property.
DBHelperClassName	DB2 Helper class name. Example: com.ibm.websphere.rsadapter.DB2UniversalDataStoreHelper (default)
DBDsImplClassName	DB2 datasource implementation class name. Example: com.ibm.db2.jcc.DB2ConnectionPoolDataSource (default)
DBDriverVarName	WebSphere environment variable name for DB2 JDBC driver class path. Example: TIP_JDBC_DRIVER_PATH
DBJDBCDriverPath	Location of DB2 JDBC driver libraries (for example, db2jcc.jar). Example: C:/IBM/tivoli/tipv2/universalDriver/lib
DBDriverType	JDBC driver type. Example: 4 (default)
DBType	Database type. Example: DB2 (default)
JaasAliasName	JAAS alias name used to store database username and password. Example: TIPAlias (default)
JaasAliasDesc	Description for JAAS alias name. Example: JAAS Alias used for load balancing
LocalHost	The hostname or IP address of the machine on which the console is running. LocalHost and LocalPort uniquely identify the node in the cluster. Example: tip01.cn.ibm.com
LocalPort	Administrative console secure port. LocalHost and LocalPort uniquely identify the node in the cluster. Example: 16311
WasRoot	The full system path to where the application server and console images were extracted during installation. Example: C:/IBM/tivoli/tipv2
ProfileName	The profile name that was specified on the manageprofiles command after installation. If no profile name was specified, the default is used. Example: TIPProfile (default)
CellName	The cell name that was specified on the manageprofiles command after installation. If no cell name was specified, the default is used. Example: TIPCell (default) This parameter is optional for a single node console installation. For a load balancing cluster, however, it is required to ensure all nodes use the same cell name.
NodeName	The application server node name. Example: TIPNode (default)
ServerName	The WebSphere Application Server instance name. Example: server1 (default)
IscAppName	The Tivoli Integrated Portal Server enterprise application name. The Tivoli Integrated Portal Server enterprise application is installed in directory the following directory: \${WAS_ROOT}\profiles\\${ProfileName}\installedApps\ \${CellName}\\${IscAppName}.ear Example: isc (default)

Property name	Description
LoggerLevel	The level of logging required. The default is OFF. Example: FINER
HAEnabled	Indicates if load balancing is enabled. Attention: Do not edit this value manually.

4. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:

- **Windows** stopServer.bat server1
- **UNIX** **Linux** stopServer.sh server1

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.

5. Make sure your database is empty and the server is not started. Problems may occur if you try to setup load balancing on a non-empty database or active server.
6. From a command prompt, change to the *tip_home_dir*/profiles/TIPProfile/bin/ha directory and issue this command:

- **Windows** ..\ws_ant.bat -f install.ant configHA
-Dusername=DB2_username -Dpassword=DB2_password
- **Linux** **UNIX** ../ws_ant.sh -f install.ant configHA
-Dusername=DB2_username -Dpassword=DB2_password

7. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:

- **Windows** startServer.bat server1
- **UNIX** **Linux** startServer.sh server1

Results

The load balancing cluster is created and the console node is joined to the cluster as the first node.

What to do next

Add (or join) additional nodes to the cluster.

Joining a node to a load balancing cluster

You can configure a Tivoli Integrated Portal Server to join an existing load balancing cluster.

Before you begin

1. Make sure you have successfully enabled load balancing following the steps in "Setting up a load balancing cluster" on page 540.
2. Tivoli Integrated Portal should be installed to the node using the same cell name that is designated for the cluster.
3. All console modules deployed to the cluster must be already deployed to the node that you intend to join.
4. You should deploy any wires or transformations used by the nodes in the cluster.

5. If the cluster is using any customization changes in `consoleProperties.xml` you must copy these changes and this file to the same location on the node that you intend to join.
6. The node must be configured to the same LDAP with the same user and group definitions as all other nodes in the cluster.

The following parameters are used on the join option when a node is added:

- **-Dusername** - specify the DB2 administrator's username
 - **-Dpassword** - specify the DB2 administrator's password
1. Check that you have the JDBC driver for DB2 on the computer where Tivoli Integrated Portal is installed. The JDBC driver should be available at:
`tip_home_dir/universalDriver/lib`.
 2. From a command prompt, change to the `tip_home_dir/profiles/TIPProfile/bin/ha` directory and edit the settings in `tipha.properties`.

Property name	Description
DBHost	The hostname or IP address of the machine where the DB2 database is installed. Example: <code>tipdb.cn.ibm.com</code>
DBPort	Port number of the DB2 server. Example: <code>50000</code> (default)
DBName	The name of the database that you created. Example: <code>tipdb</code>
DBProviderClass	Class name of the DB2 provider. Example: <code>com.ibm.db2.jcc.DB2Driver</code> (default)
DBProviderName	Name of the DB2 provider. Example: <code>TIP_Universal_JDBC_Driver</code> (default)
DBDatasource	JNDI name of the datasource. Example: <code>jdbc/tipds</code> Note: For the Web GUI, do not change the value of this property.
DBDatasourceName	Name of the datasource used for load balancing. Example: <code>tipds</code> Note: For the Web GUI, do not change the value of this property.
DBHelperClassName	DB2 Helper class name. Example: <code>com.ibm.websphere.rsadapter.DB2UniversalDataStoreHelper</code> (default)
DBDsImplClassName	DB2 datasource implementation class name. Example: <code>com.ibm.db2.jcc.DB2ConnectionPoolDataSource</code> (default)
DBDriverVarName	WebSphere environment variable name for DB2 JDBC driver class path. Example: <code>TIP_JDBC_DRIVER_PATH</code>
DBJDBCdriverPath	Location of DB2 JDBC driver libraries (for example, <code>db2jcc.jar</code>). Example: <code>C:/IBM/tivoli/tipv2/universalDriver/lib</code>
DBDriverType	JDBC driver type. Example: <code>4</code> (default)
DBType	Database type. Example: <code>DB2</code> (default)
JaasAliasName	JAAS alias name used to store database username and password. Example: <code>TIPAlias</code> (default)
JaasAliasDesc	Description for JAAS alias name. Example: <code>JAAS Alias used for load balancing</code>

Property name	Description
LocalHost	The hostname or IP address of the machine on which the console is running. LocalHost and LocalPort uniquely identify the node in the cluster. Example: tip01.cn.ibm.com
LocalPort	Administrative console secure port. LocalHost and LocalPort uniquely identify the node in the cluster. Example: 16311
WasRoot	The full system path to where the application server and console images were extracted during installation. Example: C:/IBM/tivoli/tipv2
ProfileName	The profile name that was specified on the manageprofiles command after installation. If no profile name was specified, the default is used. Example: TIPProfile (default)
CellName	The cell name that was specified on the manageprofiles command after installation. If no cell name was specified, the default is used. Example: TIPCell (default) This parameter is optional for a single node console installation. For a load balancing cluster, however, it is required to ensure all nodes use the same cell name.
NodeName	The application server node name. Example: TIPNode (default)
ServerName	The WebSphere Application Server instance name. Example: server1 (default)
IscAppName	The Tivoli Integrated Portal Server enterprise application name. The Tivoli Integrated Portal Server enterprise application is installed in directory the following directory: \${WAS_ROOT}\profiles\\${ProfileName}\installedApps\ \${CellName}\\${IscAppName}.ear Example: isc (default)
LoggerLevel	The level of logging required. The default is OFF. Example: FINER
HAEnabled	Indicates if load balancing is enabled. Attention: Do not edit this value manually.

3. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:

- **Windows** stopServer.bat server1
- **UNIX** **Linux** stopServer.sh server1

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.

4. Make sure the Tivoli Integrated Portal Server is not started.
5. At a command prompt, change to the *tip_home_dir*/profiles/TIPProfile/bin/ha directory and issue this command
 - **Windows** ..\ws_ant.bat -f install.ant configHA -Dusername=DB2_username -Dpassword=DB2_password
 - **Linux** **UNIX** ../ws_ant.sh -f install.ant configHA -Dusername=DB2_username -Dpassword=DB2_password

6. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:

- Windows `startServer.bat server1`
- UNIX Linux `startServer.sh server1`

Results

The console node is joined to the cluster.

What to do next

Add another node to the cluster, or if you have completed adding nodes, enable server to server trust for each node to every other node in the cluster.

Depending on the network dispatcher (for example, IBM HTTP Server) that you use, you might have further updates to get session requests routed to the new node. Refer to the documentation applicable to your network dispatcher for more information.

Enabling server-to-server trust

Use this procedure to enable load balanced nodes to connect to each other and send notifications.

These steps are required to enable load balancing between the participating nodes. Complete these steps on each node.

1. In a text editor, open the `ssl.client.props` file from the *tip_home_dir*/profiles/TIPProfile/properties directory.
2. Uncomment the section that starts with **com.ibm.ssl.alias=AnotherSSLSettings** so that it looks like this:

```
com.ibm.ssl.alias=AnotherSSLSettings
com.ibm.ssl.protocol=SSL_TLS
com.ibm.ssl.securityLevel=HIGH
com.ibm.ssl.trustManager=IbmX509
com.ibm.ssl.keyManager=IbmX509
com.ibm.ssl.contextProvider=IBMJSE2
com.ibm.ssl.enableSignerExchangePrompt=true
#com.ibm.ssl.keyStoreClientAlias=default
#com.ibm.ssl.customTrustManagers=
#com.ibm.ssl.customKeyManager=
#com.ibm.ssl.dynamicSelectionInfo=
#com.ibm.ssl.enabledCipherSuites=
```

3. Uncomment the section that starts with **com.ibm.ssl.trustStoreName=AnotherTrustStore** so that it looks like this:

```
com.ibm.ssl.trustStoreName=AnotherTrustStore
com.ibm.ssl.trustStore=${user.root}/etc/trust.p12
com.ibm.ssl.trustStorePassword={xor}CDo9Hgw=
com.ibm.ssl.trustStoreType=PKCS12
com.ibm.ssl.trustStoreProvider=IBMJCE
com.ibm.ssl.trustStoreFileBased=true
```

4. Update the location of the trust store that the signer should be added to in the `com.ibm.ssl.trustStore` property of `AnotherTrustStore` by replacing the default value **com.ibm.ssl.trustStore=\${user.root}/etc/trust.p12** with the correct path for your trust store. Example:

```
com.ibm.ssl.trustStore=${user.root}/config/cells/TIPCell/nodes/TIPNode02
/trust.p12
```

After the update, the section must look like this:

```
com.ibm.ssl.trustStoreName=AnotherTrustStore
com.ibm.ssl.trustStore=${user.root}/config/cells/TIPCell/nodes/TIPNode/trust.p12
com.ibm.ssl.trustStorePassword={xor}CDo9Hgw=
com.ibm.ssl.trustStoreType=PKCS12
com.ibm.ssl.trustStoreProvider=IBMJCE
com.ibm.ssl.trustStoreFileBased=true
```

5. Save your changes to `ssl.client.props`.
6. Stop and restart the Tivoli Integrated Portal Server:
 - a. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `stopServer.bat server1`
 - **UNIX** **Linux** `stopServer.sh server1`

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.

- b. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `startServer.bat server1`
 - **UNIX** **Linux** `startServer.sh server1`
7. Complete all of the steps so far on each node before you continue with the rest of the steps.
8. Run the following command on each node for each *myremotehost* (that is, for every node that you want to enable trust with) in the cluster:

```
Windows tip_home_dir\profiles\TIPProfile\bin\retrieveSigners.bat
NodeDefaultTrustStore AnotherTrustStore -host myremotehost -port
remote_SOAP_port
```

```
Linux UNIX tip_home_dir/profiles/TIPProfile/bin/
retrieveSigners.sh NodeDefaultTrustStore AnotherTrustStore -host
myremotehost -port remote_SOAP_port
```

where *myremotehost* is the name of the computer to enable trust with; *remote_SOAP_port* is the SOAP connector port number (16313 is the default). If you have installed with non-default ports, check `tip_home_dir/properties/TIPPortDef.properties` for the value of `SOAP_CONNECTOR_ADDRESS` and use that.

9. Stop and restart the Tivoli Integrated Portal Server:
 - a. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `stopServer.bat server1`
 - **UNIX** **Linux** `stopServer.sh server1`

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.

- b. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `startServer.bat server1`
 - **UNIX** **Linux** `startServer.sh server1`

Example

In this example, the load balancing cluster is comprised of two Microsoft Windows nodes named *myserver1* and *myserver2*. The command entered on *myserver1*:

```
retrieveSigners.bat NodeDefaultTrustStore AnotherTrustStore -host myserver2  
-port 16313
```

The command entered on *myserver2*:

```
retrieveSigners.bat NodeDefaultTrustStore AnotherTrustStore -host myserver1  
-port 16313
```

Verifying a load balancing implementation

Use the information in this topic to verify that your Tivoli Integrated Portal load balancing setup is working correctly once you have added all nodes to the cluster and enabled server-to-server trust.

This task allows you to confirm the following functions are working correctly:

- The database used for your load balancing cluster is properly created and initialized.
- Every node in the cluster uses the database as its repository instead of its own local file system.
- Server-to-server trust is properly enabled between nodes in the cluster.

To verify your load balancing configuration:

1. Ensure that each Tivoli Integrated Portal Server instance on every node in the cluster is running.
2. In a browser, log into one node, create a new View and save your changes.
3. Log into the remaining nodes and verify that the newly created view is available in each one.

Preparing the HTTP server for load balancing

Install the IBM HTTP Server and configure the Web server plug-in for passing requests to the Tivoli Integrated Portal Server that are part of the load balancing configuration.

Before you begin

The IBM HTTP Server uses a Web server plug-in to forward HTTP requests to the Tivoli Integrated Portal Server. You can configure the HTTP server and the Web server plug-in to act as the load balancing server, that is, pass requests (HTTP or HTTPS) to one of any number of nodes. The load balancing methods supported by the plug-in are *round robin* and *random*:

- With a round robin configuration, when a browser connects to the HTTP server, it is directed to one of the configured nodes. When another browser connects, it is directed to a different node.
- With the random setting, each browser is connected randomly to a node. Once a connection is established between a browser and a particular node, that connection remains until the user logs out or the browser is closed.

The HTTP server is necessary for directing traffic from browsers to the applications that run in the Tivoli Netcool/OMNIBus Web GUI environment. The server is installed between the portal and the Tivoli Integrated Portal Server, and is outside the firewall.

The Web server plug-in uses the `plugin-cfg.xml` configuration file to determine whether a request is for the application server.

Complete this procedure to configure the Web server plug-in for load balancing for each node.

1. Install IBM HTTP Server ensuring that you include the IBM HTTP Server Plug-in for IBM WebSphere Application Server option. For more information, see http://publib.boulder.ibm.com/infocenter/wasinfo/fep/topic/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_installihs.html.
2. Create a new CMS-type key database. For more information see http://publib.boulder.ibm.com/infocenter/wasinfo/fep/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_createkeydb.html.
3. Create a self-signed certificate to allow SSL connections between nodes. For more information, see http://publib.boulder.ibm.com/infocenter/wasinfo/fep/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_certselfsigned.html.

4. To enable SSL communications for the IBM HTTP Server, in a text editor, open `HTTP_server_install_dir/conf/httpd.conf`. Locate the line `# End of example SSL configuration` and add the following lines, ensuring that the `KeyFile` line references the key database file created in step 2 and save your changes.

```
LoadModule ibm_ssl_module modules/mod_ibm_ssl.so
<IfModule mod_ibm_ssl.c>
    Listen 443
    <VirtualHost *:443>
        SSLEnable
    </VirtualHost>
</IfModule>
SSLDisable
KeyFile "C:/Program Files/IBM/HTTPServer/bin/test.kdb"
```

For more information, refer to the first example at http://publib.boulder.ibm.com/infocenter/wasinfo/fep/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_setupssl.html.

5. Restart the IBM HTTP Server. For more information, see http://publib.boulder.ibm.com/infocenter/wasinfo/fep/topic/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_startihs.html.
6. On the IBM HTTP Server computer, to verify that SSL is enabled ensure that you can access `https://localhost`.
7. Stop and restart the Tivoli Integrated Portal Server:
 - a. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `stopServer.bat server1`
 - **UNIX** **Linux** `stopServer.sh server1`
 - Note:** On UNIX and Linux systems, you are prompted to provide an administrator username and password.
 - b. In the `tip_home_dir/profiles/TIPProfile/bin` directory, depending on your operating system, enter one of the following commands:
 - **Windows** `startServer.bat server1`
 - **UNIX** **Linux** `startServer.sh server1`
8. Start the HTTP server:
 - a. Change to the directory where it is installed.




- b. Run this command: `bin/apachectl start` Note you must restart the server after changing the `plugin-cfg.xml` file.

What to do next

Enter the URL for the HTTP Server in a browser `http://HTTP_server_host/HTTP_server_port` and it will be forwarded to one of the nodes.

Note: The default load balancing method is random, whereby each browser is connected randomly to a node.

Related reference

-  IBM HTTP Server V7.0 Information Center
-  IBM DB2 Database for Linux, UNIX, and Windows Information Center
-  Web server plug-in tuning tips

Setting clone IDs for nodes

Assign a clone ID for all nodes in the cluster.

Complete this procedure to set clone IDs for all nodes in the cluster. You must carry out these steps on each node.

1. In a text editor, open the `server.xml` file from the `tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/nodes/TIPNode/servers/server1` directory
2. In `server.xml`, locate the entry `<components xmi:type="applicationserver.webcontainer:WebContainer"`
3. Within the `components` element, add the following entry:

```
<properties xmi:id="WebContainer_1183077764084" name="HttpSessionCloneId"
  value="12345" required="false"/>
```

Where:

`value` is the clone ID for the node, for example, `value="12345"`. The clone ID must be unique to each node. An example of an updated `components` element is provided here:

```
<components xmi:type="applicationserver.webcontainer:WebContainer"
  xmi:id="WebContainer_1183077764084" enableServletCaching="false"
  disablePooling="false">
  <stateManagement xmi:id="StateManageable_1183077764087"
    initialState="START"/>
  <services xmi:type="applicationserver.webcontainer:SessionManager"
    xmi:id="SessionManager_1183077764084" enable="true" enableUrlRewriting="false"
    enableCookies="true" enableSSLTracking="false"
    enableProtocolSwitchRewriting="false"
    sessionPersistenceMode="NONE" enableSecurityIntegration="false"
    allowSerializedSessionAccess="false" maxWaitTime="5"
    accessSessionOnTimeout="true">
    <defaultCookieSettings xmi:id="Cookie_1183077764084" domain=""
      maximumAge="-1" secure="false"/>
    <sessionDatabasePersistence
      xmi:id="SessionDatabasePersistence_1183077764084"
      datasourceJNDIName="jdbc/
        Sessions" userId="db2admin" password="{xor}0z1tPjsyNjE="
      db2RowSize="ROW_SIZE_4KB" tableSpaceName=""/>
    <tuningParams xmi:id="TuningParams_1183077764084"
      usingMultiRowSchema="false" maxInMemorySessionCount="1000"
      allowOverflow="true" scheduleInvalidation="false"
      writeFrequency="TIME_BASED_WRITE" writeInterval="10"
      writeContents="ONLY_UPDATED_ATTRIBUTES" invalidationTimeout="30">
      <invalidationSchedule xmi:id="InvalidationSchedule_1183077764084"
        firstHour="14" secondHour="2"/>
    </tuningParams>
  </services>
</components>
```

```

        </tuningParams>
    </services>
    <properties xmi:id="WebContainer_1183077764084" name="HttpSessionCloneId"
value="12345" required="false"/>
</components>

```

4. Save the changes you made to `server.xml`.

Generating the plugin-cfg.xml file

Run `GenPluginCfg.bat` to generate the `plugin-cfg.xml` file and save it in `tip_home_dir/profiles/TIPProfile/config/cells`.

Complete this procedure to generate the `plug-cfg.xml` file. You must carry out these steps on each node.

1. On a node, change to `tip_home_dir/profiles/TIPProfile/bin/` and run the following command:

- Windows `GenPluginCfg.bat`
- Linux UNIX `GenPluginCfg.sh`

This command generates a file called `plugin-cfg.xml` and saves it to the `tip_home_dir/profiles/TIPProfile/config/cells` directory.

2. On the IBM HTTP Server, in the following directory, replace the existing `plugin-cfg.xml` with the version generated in step 1:

`HTTP_web_server_install_dir/plugins/config/webserver1`

The following steps establish the new `/ibm/*` URI (Uniform Resource Identifier), which is where the plug-in will redirect requests:

- a. On the IBM HTTP Server, change to the directory where the Web server definition file is (such as `cd plugins/config/webserver1`).
- b. Open the `plugin-cfg.xml` file in a text editor, and in reference to the sample content extract provided below, edit the file to provide details of your IBM HTTP Server and all Tivoli Integrated Portal Server instances.

HTTP SERVER PATH is the path to where the HTTP server is installed.

HTTP SERVER PORT is the port for the HTTP server.

SERVER1 is the fully qualified name of the computer where the application server is installed and started.

SERVER2 is the fully qualified name of the computer where another application server is installed and started.

CLONE_ID is the is the unique clone ID assigned to a particular node (server) in the cluster.

- c. In the `ServerCluster` section, the values for the keyring and stashfile properties should be **HTTP SERVER PATH** `/plug-ins/etc/plugin-key.kdb` and **HTTP SERVER PATH** `/plug-ins/etc/plugin-key.sth` respectively.
- d. Continue to add Server entries for any other nodes, following the same pattern. Add a new entry under `PrimaryServers` for each additional server.
- e. Add `CloneID` and `LoadBalanceWeight` attributes for every Server entry.

Attention: The HTTP and HTTPS port values for all nodes should be the same.

```

<Config ASDisableNagle="false" IISDisableNagle="false"
IgnoreDNSFailures="false" RefreshInterval="60"
ResponseChunkSize="64" AcceptAllContent="false"
IISPluginPriority="High" FIPSEnable="false"
AppServerPortPreference="HostHeader" VHostMatchingCompat="false"
ChunkedResponse="false">
  <Log LogLevel="Trace" Name="HTTP SERVER PATH/Plugins/logs/webserver1/

```

```

http_plugin.log"/>
<Property Name="ESIEnable" Value="true" />
<Property Name="ESIMaxCacheSize" Value="1024" />
<Property Name="ESIInvalidationMonitor" Value="false" />
<Property Name="ESIEnableToPassCookies" Value="false" />
<Property Name="PluginInstallRoot" Value="HTTP SERVER PATH/Plugins" />
<VirtualHostGroup Name="default_host">
  <VirtualHost Name="*:16310" />
  <VirtualHost Name="*:80" />
  <VirtualHost Name="*:16311" />
  <VirtualHost Name="*:5060" />
  <VirtualHost Name="*:5061" />
  <VirtualHost Name="*:443" />
  <VirtualHost Name="*:HTTP SERVER PORT" />
</VirtualHostGroup>
<ServerCluster CloneSeparatorChange="false" GetDWLMTable="false"
IgnoreAffinityRequests="true" LoadBalance="Round Robin"
Name="server1_Cluster" PostBufferSize="64" PostSizeLimit="-1"
RemoveSpecialHeaders="true" RetryInterval="60">
  <Server Name="TIPNode1_server1"
ConnectTimeout="0" CloneID="CLONE_ID" ExtendedHandshake="false"
ServerIOTimeout="0" LoadBalanceWeight="100" MaxConnections="-1"
WaitForContinue="false">
    <Transport Hostname="SERVER1" Port="16310"
Protocol="http"/>
    <Transport Hostname="SERVER1" Port="16311"
Protocol="https">
      <Property name="keyring" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.kdb"/>
      <Property name="stashfile" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.sth"/>
    </Transport>
  </Server>
  <Server Name="TIPNode1_server2"
ConnectTimeout="0" CloneID="CLONE_ID" ExtendedHandshake="false"
ServerIOTimeout="0" LoadBalanceWeight="100" MaxConnections="-1"
WaitForContinue="false">
    <Transport Hostname="SERVER2" Port="16310"
Protocol="http"/>
    <Transport Hostname="SERVER2" Port="16311"
Protocol="https">
      <Property name="keyring" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.kdb"/>
      <Property name="stashfile" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.sth"/>
    </Transport>
  </Server>
</PrimaryServers>
</ServerCluster>
<UriGroup Name="server1_Cluster_URIs">
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/ivt/*" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/*" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/*.jsp" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/*.jsv" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/*.jsw" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/j_security_check" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/ibm_security_logout" />

```

```

    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/console/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/help/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/action/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCWire/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/isc/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCHA/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/tip_ISCAdminPortlet/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCAdminPortlets/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/mum/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPChangePasswd/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPEXportImport/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/tivoli/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/proxy/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/TIPWebWidget/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/dbfile/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPChartPortlet/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/TIPUtilPortlets/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/WIMPortlet/*" />
    <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/SysMgmtCommonTaskGroups/*" />
</UriGroup>
<Route ServerCluster="server1 Cluster" UriGroup="server1_Cluster_URIs"
VirtualHostGroup="default_host" />
<RequestMetrics armEnabled="false" newBehavior="false" rmEnabled="false"
traceLevel="HOPS">
    <filters enable="false" type="URI">
        <filterValues enable="false" value="/snoop" />
        <filterValues enable="false" value="/hitcount" />
    </filters>
    <filters enable="false" type="SOURCE_IP">
        <filterValues enable="false" value="255.255.255.255" />
        <filterValues enable="false" value="254.254.254.254" />
    </filters>
    <filters enable="false" type="JMS">
        <filterValues enable="false" value="destination=aaa" />
    </filters>
    <filters enable="false" type="WEB_SERVICES">
        <filterValues enable="false" value="wsdlPort=aaa:op=bbb:nameSpace=ccc" />
    </filters>
</RequestMetrics>
</Config>

```

Configuring SSL from each node to the IBM HTTP Server

For load balancing implementations, you must configure SSL between the IBM HTTP Server plug-in and each node in the cluster.

Before you begin

This task assumes that you have already installed and configured the IBM HTTP Server for load balancing.

For each node in the cluster, follow these instructions to configure the node to communicate over a secure (SSL) channel with the IBM HTTP Server.

1. Log in to the Tivoli Netcool/OMNIBus Web GUI.
2. In the navigation pane, click **Settings > Websphere Administrative Console** and click **Launch Websphere administrative console**.
3. Follow these steps to extract signer certificate from the trust store:
 - a. In the WebSphere Application Server administrative console navigation pane, click **Security > SSL certificate and key management**.
 - b. In the Related Items area, click the **Key stores and certificates** link and in the table click the **NodeDefaultTrustStore** link.
 - c. In the Additional Properties area, click the **Signer certificates** link and in the table that is displayed, select the root entry check box.
 - d. Click **Extract** and in the page that is displayed, in the **File name** field, enter a certificate file name (*certificate.arm*), for example, *c:\tivpc064ha1.arm*.
 - e. From the **Data Type** list select the **Base64-encoded ASCII data** option and click **OK**.
 - f. Locate the extracted signer certificate and copy it to the computer running the IBM HTTP Server.

Note: These steps are particular to Tivoli Integrated Portal, for general WebSphere Application Server details and further information, see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.base.doc/info/aes/ae/tsec_sslextractsigncert.html

4. On the computer running the IBM HTTP Server, follow these steps to import the extracted signer certificate into the key database:
 - a. Start the key management utility (iKeyman), if it is not already running, from *HTTP_SERVER_PATH/bin*:
 - **UNIX** **Linux** At the command line, enter `./ikeyman.sh`
 - **Windows** At the command line, enter `ikeyman.exe`
 - b. Open the CMS key database file that is specified in *plugin-cfg.xml*, for example, *HTTP_SERVER_PATH/plugin-ins/etc/plugin-in-key.kdb*.
 - c. Provide the password (default is WebAS) for the key database and click **OK**.
 - d. From the **Key database content**, select **Signer Certificates**.
 - e. Click **Add** and select the signer certificate that you copied from the node to the computer running the IBM HTTP Server and click **OK**.
 - f. Select the **Stash password to a file** check box and click **OK** to save the key database file.

Note: For more information on certificates in WebSphere Application Server, see: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_ikeyssca.html

5. Repeat these steps for each node in the cluster.

6. For the changes to take effect, stop and restart all nodes in the cluster and also restart the computer running the IBM HTTP Server.
 - a. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:
 - **Windows** stopServer.bat server1
 - **UNIX** **Linux** stopServer.sh server1

Note: On UNIX and Linux systems, you are prompted to provide an administrator username and password.
 - b. In the *tip_home_dir*/profiles/TIPProfile/bin directory, depending on your operating system, enter one of the following commands:
 - **Windows** startServer.bat server1
 - **UNIX** **Linux** startServer.sh server1
 - c. Restart the IBM HTTP Server. For more information, see http://publib.boulder.ibm.com/infocenter/wasinfo/fep/topic/com.ibm.websphere.ihs.doc/info/ihs/ihs/tihs_startihs.html.

What to do next

You should now be able to access the load balanced cluster through https://http_server_hostname/ibm/console (assuming that the default context root (/ibm/console) was defined in at the time of installation).

Starting Web GUI load balancing operation

Once you have added a node you start Web GUI load balancing. When creating a cluster, carry out this operation on each of the cluster's nodes.

To start Web GUI load balancing on a node:

1. Open *webgui_home_dir*/etc/server.init in a text editor.
2. Locate the load balancing properties that begin with the **cluster.mode** property.
3. Change the values of the properties as follows:

Table 102. Setting the load balancing properties

Property	Value
cluster.mode	on
cluster.hostname	The name or TCP/IP address of the host that is running the new node. Example: server1
cluster.port	The number of the SSL port that the Web GUI server uses. Example: 16311

4. Find the property **timedtasks.enabled** and set its value to true.
5. Modify or set up the schedules for timed tasks as required.

Note: Define the same set of timed tasks with identical schedules on all nodes in the cluster.

6. Save the file.
7. Restart the server.

The node joins the cluster and reads its configuration data from the database.

Related tasks

“Restarting the server” on page 568

Adding a node to an existing cluster

You can add additional nodes to an existing cluster. Each node must first be set up in the same way as all the nodes already in the cluster. Then you add the node and configure it to work in the cluster.

Before you begin

Install the Web GUI on the new node, and configure it in exactly the same way as the existing nodes in the cluster, with the exception of setting the `server.init` properties that enable operation in a cluster. These properties have the prefix `cluster.`

The procedure to add a node consists of the following steps:

1. “Joining a node to a load balancing cluster” on page 542
2. “Enabling server-to-server trust” on page 545
3. “Setting clone IDs for nodes” on page 549
4. “Generating the `plugin-cfg.xml` file” on page 550
5. “Configuring SSL from each node to the IBM HTTP Server” on page 553
6. “Starting Web GUI load balancing operation” on page 554

Related reference

Appendix E, “Web GUI initialization file properties,” on page 633

Removing a node from a cluster

Removing a node that is no longer required in a cluster is a 3-stage procedure: stopping the Web GUI load balancing operations on the node, removing the node from the cluster, restarting the node.

Removing the Web GUI load balancing information:

Before removing a node, remove its load balancing information from the cluster's configuration database and reset the load balancing properties in `server.init`.

Before you begin

Make sure that no users are logged in to the node.

To remove the load balancing information:

1. Run the following WAAPI command file from the node you want to remove:
`webgui_home_dir/waapi/etc/samples/cluster_removalnode.xml`
This removes the node's details from the configuration database of the cluster.
2. Open `webgui_home_dir/etc/server.init` in a text editor.
3. Locate the property **`cluster.mode`** and set its value to `off`.
4. Save the file.

Removing the node:

Follow these steps to remove a node from the load balancing cluster.

The following parameters are used on the disjoin option when a node is removed.

- **-Dusername** - specify the DB2 administrator's username
 - **-Dpassword** - specify the DB2 administrator's password
1. From a command prompt, change to the *tip_home_dir/profiles/TIPProfile/bin/ha* directory and issue this command:
 - **Windows** `..\ws_ant.bat -f uninstall.ant disjoin -Dusername=DB2_username -Dpassword=DB2password`
 - **Linux** **UNIX** `../ws_ant.sh -f uninstall.ant disjoin -Dusername=DB2_username -Dpassword=DB2password`
 2. Update the network dispatcher (for example, IBM HTTP Server) to remove the node from the configuration.

Removing a remote node:

This command should be used only in the rare occasions where physical access to the node is not available or a serious hardware or software failure has occurred. If the node is remotely disjoined but continues to function, some problems with synchronization might arise that can lead to problems with data consistency and synchronization.

1. From a command prompt, change to the *tip_home_dir/profiles/TIPProfile/bin/ha* directory and issue this command:
 - **Windows** `..\ws_ant.bat -f uninstall.ant remote-disjoin -DremoteHost=remote_host -DremotePort=9044 -Dusername=DB2_username -Dpassword=DB2_password`
 - **Linux** **UNIX** `../ws_ant.sh -f uninstall.ant remote-disjoin -DremoteHost=remote_host -DremotePort=9044 -Dusername=DB2_username -Dpassword=DB2_password`
2. Update the network dispatcher (for example, IBM HTTP Server) to remove the node from the configuration.

Restarting the server as a stand-alone system:

Restart the removed node to use it as a stand-alone system. This implements the changes in the load balancing properties that you made when removing the node so making it a stand-alone system once again.

Related tasks

“Restarting the server” on page 568

Configuring launch-in-context integrations with Tivoli products

You can configure the Web GUI to launch into compatible Tivoli products.

The following types of integrations can be configured. In both cases, the configuration options differ depending on whether the product is based on Tivoli Integrated Portal.

Launch-out integrations

Another Tivoli product is launched from the Web GUI.

Launch-in integrations

The Web GUI is launched from another Tivoli product.

This information describes configurations for the Web GUI. For information about integration configurations for other Tivoli products, see the information center for that product.

Related concepts

“Integration with other Tivoli products” on page 31

Integration prerequisites

Before you can configure an integration between the Web GUI and another Tivoli product, you must take note of a number of prerequisites.

These prerequisites are as follows:

- An LDAP server must be configured as the user registry.
- To avoid having to reenter your user credentials when you launch from the Web GUI to another Tivoli product, single sign-on must be configured. The computers on which each product is running must be configured to be part of the same Websphere Application Server single sign-on domain. For integrations between products that run in the Tivoli Integrated Portal framework, all Tivoli Integrated Portal products must use a common user registry.
- The Web GUI must support integration with the Tivoli product and version that you want to configure.

Related concepts

“Integration with other Tivoli products” on page 31

“Single sign-on” on page 30

Related tasks

“Configuring single sign-on” on page 530

Mapping of user roles between products

For integrations between the Web GUI and products based on Tivoli Integrated Portal, the user roles of some products map directly to Web GUI roles.

If an integrating product is not based on Tivoli Integrated Portal, you must make sure the required users are created in both the Web GUI and the integrating product, and that in both products the users are assigned the required roles.

If the roles of a product based on Tivoli Integrated Portal do not map to Web GUI roles, you must make sure that the required users are assigned both the Web GUI roles, and the corresponding roles from the integrating product.

The following table describes the products, and the roles of those products, that map to specific Web GUI roles.

Table 103. Mapping of Tivoli Integrated Portal products to Web GUI roles

Integrating product	Role in integrating product	Corresponding Web GUI role
IBM Tivoli Network Manager IP Edition	ncp_networkview	ncw_user
IBM Tivoli Network Manager IP Edition	ncp_hopview	ncw_user
IBM Tivoli Network Manager IP Edition	ncp_mibbrowser	ncw_user
IBM Tivoli Network Manager IP Edition	ncp_structurebrowser	ncw_user

Note: In addition to the ncw_user role, a Web GUI user must also have the ncw_admin role, netcool_rw role, or the netcool_ro role assigned; these roles control whether the user has administrative access, read-write access or read-only access.

For more information about Web GUI roles, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

Configuring Web GUI launch-out integrations

You can launch out of the Web GUI to another Tivoli product in the following ways: by creating a script tool that is launched from the Active Event List, by configuring the Event Dashboard to run a script that launches a Tivoli product, or, for products that are based on Tivoli Integrated Portal, by creating and subscribing to events in the Tivoli Integrated Portal action framework.

Configuring Web GUI launch-out integrations for the Active Event List:

To launch out of the Active Event List (AEL), you can create a tool that, when run, launches the URL of another Tivoli product. For integrations with products that are based on Tivoli Integrated Portal, you can use the Tivoli Integrated Portal action framework to broadcast events between portlets.

Configuring Web GUI launch-out integrations using tools:

To launch another Tivoli product from the Active Event List (AEL), create a script tool that launches the URL of the product when users run the script against events in the AEL.

Before you begin

You need to know the URL of the Tivoli product that you want to launch from the AEL. For more information about how to build this URL, see the information center for the specific product. To find the information centre for a Tivoli product, see the *Tivoli Documentation Central* Web site: <http://www.ibm.com/tivoli/documentation>.

You can use this method to launch into supported products deployed with the Tivoli Integrated Portal framework and supported products that use other GUI frameworks, such as TP Ae or Java Swing. After you have created the tool, you must add it to an AEL menu.

To create a tool that launches from the AEL into other Tivoli products:

1. In the navigation pane, click **Administration > Event Management Tools**.
2. In the Tool Creation page, click **Create Tool**.
3. Select **CGI/URL** from the **Type** list.
4. Type a name for the tool in the **Name** field.
 By default, the following characters cannot be used in tool names:
`$! £ % ^ & * () + = ~ ` ~ # @ ' : ; < > { } [] ? / \ \ | , "`
 By default, the following characters cannot be used as the initial character of tool names:
`/ \ \ * ? " < > | & .`
 These invalid characters are defined in the following file:
`webgui_home_dir/etc/illegalChar.prop`
5. In the **URL** field, enter the fully-qualified URL of the application in the following format:
`protocol://hostname:port/path/?parameters`
 Where the valid values for each variable in the URL are as follows:

protocol
 The Web protocol to use. Valid values are http and https.

hostname
 The host name for the Tivoli product to which you are launching.

port
 The port number for the Tivoli product to which you are launching.

path
 The location of the requested resource.

parameters
 The parameters for the URL.
6. Complete the other fields as follows:
 - Method**
 Select **GET**.
 - Open in**
 Select **New window**.
 - Execute for each selected row**
 Select this check box to run the tool against all selected rows individually in the AEL. Clear the check box if you want the tool to run against only the first row in the selection.
 - Window for each selected row**
 Select this check box to open a separate window for each selected row in the AEL.
7. Define access for tools based on the groups that a user belongs to and the class of an event against which the tool is deployed:
 - Group** Select the user group that you want to access the tool and click **>**. To give all groups access to the selected tool, click **>>**. Users must be members of a selected group to use the tool.
 - Class** Select the class of event (defined by the Class field in the ObjectServer) that you want to access the tool and click **>**. To give all classes access to the selected tool, click **>>**.
8. Click **Save**.
9. Add the new tool to an AEL menu:

- a. Click **Administration > Event Management Tools > Menu Configuration**.
- b. From the **Available menus** list, select the menu to which you want to add the tool and click **Modify**.
- c. Select **tool** from the **Available items** list.
- d. Select the new tool and click **Add selected item**.
- e. Click **Save**.

Results

The tool is added to the selected AEL menu.

What to do next

Check that the URL is built correctly and launches the Tivoli product specified in the tool by testing it on an event in the AEL. To open the default AEL applet, click **Availability > Events > Active Event List (AEL)**. You have the following options:

- To view the tool, from the menu bar, click **Tool**.
- To run the tool against an event, right click a row in the AEL and select the tool from the list.

If the Web GUI and the product that is being launched are not configured for single sign-on, a login window is displayed. Before you can view the event information, you must provide a user name and password.

Related reference

"Sample scripts for launch-out integrations" on page 564

Configuring Web GUI launch-out integrations using the Tivoli Integrated Portal action framework:

For integrations between products based on Tivoli Integrated Portal, use the Tivoli Integrated Portal action framework to define events that can be launched by a tool in the Active Event List (AEL).

Restriction: This task is applicable only to products that run on Tivoli Integrated Portal.

The Tivoli Integrated Portal action framework defines communications between portlets. In the integrating Tivoli product, an event must be defined which can be used by the Web GUI in a tool launched from the AEL.

To create actions:

1. On the Web GUI server, open the `tip_home_dir/profiles/TIPProfile/installedApps/TIPCell/isc.ear/OMNIBusWebGUI.war/WEB-INF/ibm-portal-event.xml` file.
2. In this file, define the broadcasting event. For example:

```
<!-- Event Definition -->
<events:event-definition>
  <events:name xmlns:x="http://ibm.com/namespace">
    x:eventname
  </events:name>
</events:event-definition>
```

Where *eventname* is the name of the broadcasting event and *namespace* is the name space shorthand for the Web GUI.

Tip: The combination of *namespace* and *eventname* must be unique.

3. On the server of the integrating Tivoli product, open the `ibm-portal-event.xml` file and set up a subscription to the event in the portlet that you want to be launched from the AEL:

- a. Locate the section that pertains to the portlet that you want to subscribe to the event.
- b. Define the subscription.

For example:

```
<!-- Portlet Subscriptions -->
<events:portlet-definition-ref portletDefinitionRef="portletdefinition">
  <events:supported-subscribed-event>
    <events:name xmlns:x="http://ibm.com/namespace">x:EventName
    </events:name>
  </events:supported-subscribed-event>
</events:portlet-definition-ref>
```

Where *eventname* is the name of the event defined in step 2 on page 560, *namespace* is the name space shorthand for the Web GUI, and *portletdefinition* is the portlet definition of the page to which you want to launch from the AEL.

4. To create a tool that broadcasts the event created in steps 2 on page 560 and 3 from the AEL:
 - a. In the Web GUI navigation, click **Administration > Event Management Tools > Tool Creation**.
 - b. On the Tool Creation Page, click **Create Tool**.
 - c. In the **Name** field, type a unique name for the tool.

By default, the following characters cannot be used in tool names:

\$! % ^ & * () + = - ` ~ # @ ' : ; < > { } [] ? / \ \ | , "

By default, the following characters cannot be used as the initial character of tool names:

/ \ \ * ? " < > | & .

These invalid characters are defined in the following file:

`webgui_home_dir/etc/illegalChar.prop`

- d. In the **Script Commands** field, type the JavaScript command that broadcasts the event. For example:



```
{ $appletparam.portletNamespace } sendPortletEvent
({ 'name': 'http://ibm.com/namespace#eventname',
  'parameter': { parametervalue } });
```

Where *namespace* is the name of the broadcasting event and *eventname* is the name space shorthand for the Web GUI.

- e. Select **Execute for each selected row**.
 - f. Define access privileges for the tool based on the groups that a user belongs to and the class of an event against which the tool is deployed:

Group Select the user group that you want to access the tool and click >. To give all groups access to the selected tool, click >>. Users must be members of a selected group to use the tool.

Class Select the class of event (defined by the Class field in the ObjectServer) that you want to access the tool and click >. To give all classes access to the selected tool, click >>.
 - g. Click **Save**.
5. Configure the AEL to launch the tool created in step 4:

- a. Open the required AEL portlet. To open the default AEL, click **Availability > Events > Active Event List (AEL)**.
- b. Edit your own portlet preferences, or set a portlet default for all users:
 - To edit your own portlet preferences, click **Edit Preferences** .
 - To edit the portlet defaults for all users of this portlet, click **Edit Options > Edit Defaults** .
- c. In the **Event List Single Click Action** field, select the tool that you created in step 4 on page 561.
- d. Click **OK**.

Results

Check that the URL is built correctly and launches the Tivoli product specified in the tool by testing it on an event in the AEL. You have the following options:

- To view the tool, from the menu bar, click **Tool**.
- To run the tool against an event, right click a row in the AEL and select the tool from the list.

If the Web GUI and the product that is being launched are not configured for single sign-on, a login window is displayed. Before you can view the event information, you must provide a user name and password.

Configuring launch-out integrations for the Event Dashboard:

You can configure the Event Dashboard to run a script that launches another Tivoli product. The script is run when a user clicks a monitor box on the Event Dashboard.

Before you begin

If you want the script to launch a URL, you need to know the URL from the Tivoli product that you want to launch from the Event Dashboard. For integrations between the Web GUI and products not based on Tivoli Integrated Portal, you must provide a URL. For more information about how to build this URL, see the information center for the specific product. To find the information centre for a Tivoli product, see the *Tivoli Documentation Central* Web site: <http://www.ibm.com/tivoli/documentation>.

For integrations between the Web GUI products based on Tivoli Integrated Portal, you can also use the Tivoli Integrated Portal action framework to define events that can be launched by the script. To do so, complete steps 1 and 2 on page 563.

To configure launch-out integrations for the Event Dashboard:

1. Optional: On the Web GUI server, define the broadcasting event in the following file: `tip_home_dir/profiles/TIPProfile/installedApps/TIPCell/isc.ear/OMNIBusWebGUI.war/WEB-INF/ibm-portal-event.xml`. For example:


```
<!-- Event Definition -->
<events:event-definition>
  <events:name xmlns:x="http://ibm.com/namespace">
    x:eventname
  </events:name>
</events:event-definition>
```



Where *eventname* is the name of the broadcasting event and *namespace* is the name space shorthand for the Web GUI.

Tip: The combination of *namespace* and *eventname* must be unique.

2. Optional: On the server of the integrating Tivoli product, open the `ibm-portal-event.xml` file and set up a subscription to the event in the portlet that you want to be launched from the Event Dashboard:
 - a. Locate the section that pertains to the portlet that you want to subscribe to the event.
 - b. Define the subscription. For example:

```
<!-- Portlet Subscriptions -->
<events:portlet-definition-ref portletDefinitionRef="portletdefinition">
  <events:supported-subscribed-event>
    <events:name xmlns:x="http://ibm.com/namespace">x:eventname
    </events:name>
  </events:supported-subscribed-event>
</events:portlet-definition-ref>
```

Where *eventname* is the name of the event defined in step 1 on page 562, *namespace* is the name space shorthand for the Web GUI, and *portletdefinition* is the portlet definition of the page to which you want to launch from the Event Dashboard.

3. Open an Event Dashboard portlet. To open the default Event Dashboard, click **Availability > Events > Event Dashboard**.
4. Edit your portlet preferences, or the portlet defaults for all users:
 - To edit your own portlet preferences, click **Edit** .
 - To edit the portlet defaults, click **Edit Options**  > **Edit Defaults**.
5. In the Edit Event Dashboard Portlet Preferences window, from the **Single Click** list select **Script**.
6. In the text field, write the required JavaScript. See “Sample scripts” for more information.
7. Click **OK**.

Sample scripts

Use these examples to help you write a script for the Event Dashboard in step 5.

The following example shows how to write a script that broadcasts an event in the Tivoli Integrated Portal action framework, as defined in step 1 on page 562 and step 2:

```
{${appletparam.portletNamespace}}sendPortletEvent
({'name':'http://ibm.com/namespace#eventname',
 'parameter':{'parametervalue'}});
```

Where *namespace* is the name of the broadcasting event and *eventname*, is the name space shorthand for the Web GUI.

The following example shows how to write a script that launches a static URL; for example to launch a Tivoli product that is not based on Tivoli Integrated Portal:

```
window.open("protocol://hostname:portnumber/contextroot/?querystring");
```

Where the valid values for each variable in the URL are as follows:

protocol

The Web protocol to use. Valid values are http and https.

hostname

The host name for the Tivoli product to which you are launching.

port

The port number for the Tivoli product to which you are launching.

path

The location of the requested resource.

parameters

The parameters for the URL.

What to do next

Test the script by clicking a monitor box on the Event Dashboard. If the Web GUI and the product that is being launched are not configured for single sign-on, a login window is displayed. Before you can view the event information, you must provide a user name and password.

Related reference

“Sample scripts for launch-out integrations”

Sample scripts for launch-out integrations:

Use these samples to help you build scripts for launch-out integrations in Active Event List (AEL) tools and in the Event Dashboard.

To launch into IBM Tivoli Service Request Manager

The following sample shows how to launch into Tivoli Service Request Manager, using the value of the TTNNumber field for the event:

```
window.open("protocol://host.domain:port/maximo/ui/maximo.jsp?event=loadapp
&value=incident&additionalevent=sqlwhere
&additionaleventvalue=ticketid%3D%27{@TTNumber}%27
```

To launch into IBM Tivoli Application Dependency Discovery Manager (TADDM)

The following sample shows how to launch into TADDM, using the values of the URL and identifier of the event:

```
window.open("{@URL}/cdm/servlet/LICServlet?
graph=physicalinfrastructure&guid={@Identifier}&console=java");
```

To launch into IBM Tivoli Monitoring

The following example shows how to launch into IBM Tivoli Monitoring:

```
var str={@TECHostName};
var unquoted = str.replace('/',g, "");
window.open("protocol://host.domain:port///cnp/kdh/lib/cnp.html
?hostname=" + unquoted);
```

To launch a Web site

The following sample shows how to launch a predetermined Web site, depending on the severity of the event:


```

if ({@Severity} > 2) {
    window.open("http://www.ibm.com");
} else {
    window.open("http://www.google.com");
}

```

Related tasks

“Configuring launch-out integrations for the Event Dashboard” on page 562

“Configuring Web GUI launch-out integrations using tools” on page 558

Configuring Web GUI launch-in integrations

You can launch into the Web GUI from another Tivoli product in the following ways: by building a URL that opens a Web GUI application or, for products that are based on Tivoli Integrated Portal, by using the Tivoli Integrated Portal to define and subscribe to events.

Configuring Web GUI launch-in integrations for Tivoli Integrated Portal products:

For products that are based on Tivoli Integrated Portal, use the Tivoli Integrated Portal action framework to define an event in the Web GUI , and define a tool in the launching product that broadcasts the event.

These instructions describe the configuration steps only for the Web GUI. For the configuration of the launching Tivoli product, see the information center for that product. To find the information centre for a Tivoli product, see the *Tivoli Documentation Central* Web site: <http://www.ibm.com/tivoli/documentation>.

To configure the Web GUI for a launch-in integration by another Tivoli product:

1. In the `ibm-portal-event.xml` file of the launching Tivoli product, define the event. For example:

```

<!-- Event Definition -->
<events:event-definition>
    <events:name xmlns:x="http://ibm.com/namespace">
        x:eventname
    </events:name>
</events:event-definition>

```

Where *namespace* is the name space shorthand of the launching product and *eventname* is the name of the event.

Tip: The combination of *namespace* and *eventname* must be unique.

2. On the Web GUI server, open the `tip_home_dir/profiles/TIPProfile/installedApps/TIPCell/isc.ear/OMNIBusWebGUI.war/WEB-INF/ibm-portal-event.xml` file.
3. Subscribe to the event created in step 1. For example:

```

<!-- Portlet Subscriptions -->
<events:portlet-definition-ref portletDefinitionRef="portletdefinition">
    <events:supported-subscribed-event>
        <events:name xmlns:x="http://ibm.com/namespace">
            x:eventname
        </events:supported-subscribed-event>
        </events:name>
    </events:portlet-definition-ref>

```

Where *portletdefinition* is the portlet definition for the Web GUI application that you want to subscribe to the event, *namespace* is the name space shorthand of the launching product, and *eventname* is the name of the event created in step 1.

- Tip:** The portlet definition for the AEL is `item.portletDef.AEL`.
4. Save and close the file.

What to do next

In the launching Tivoli product, you must perform the following tasks:

1. Define a tool to broadcast the event.
2. In the launching product, configure a single-click action to launch the tool.

Configuring Web GUI launch-in integrations for non-Tivoli Integrated Portal products:

For products that are not based on Tivoli Integrated Portal, to launch into the Web GUI, build a URL that points to a Web GUI application and launch that URL from your product.

To build the URL:

- Use the URL parameters of any Web GUI application. These URLs have the following format:

protocol://server.domain:port/ibm/console/webtop/path?querystring

Where the valid values for each variable in the URL are as follows:

protocol

The Web protocol to use. Valid values are `http` and `https`.

server

The server on which the Web GUI is hosted.

domain

The domain of the server.

port

The port number for the Web GUI server.

path

The location of the requested resource.

querystring

Contains name-value pair parameters that are delimited by separators. The format for a name-value pair is `name=value`. Use an equals sign (=) to separate names and values, and use an ampersand (&) to separate name-value pairs.

Important: You must use the fully-qualified host name of the Web GUI server in URLs.

For more information about the Web GUI HTTP GET parameters, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

- Use the URL of any `.html` file created by using SmartPage commands. These URLs have the following format:

protocol://server.domain:port/ibm/console/webtop/filename.html

Where *filename* is the name of the `.html` file. For more information about SmartPage commands, see the *IBM Tivoli Netcool/OMNIBus Web GUI Administration and User's Guide*.

What to do next

Use the functions provided by the launching product to launch the Web GUI URL. If the Web GUI and the product that is being launched are not configured for single sign-on, a login window is displayed. Before you can view the event information, you must provide a user name and password.

Examples of how to specify the URL:

Use these examples to help you build a URL for launching the Web GUI from another Tivoli product.

The examples are as follows:

Example for launching the Active Event List (AEL) with a predefined filter and view, and a single data source

```
http://home.webgui.com:16315/ibm/console/webtop/  
AELView?filtertype=global&filtername=Last10Mins  
&view=DefaultNoSevAckCols&datasource=NCOMS
```

Example for launching the AEL with a predefined filter and view, and two data sources

```
http://home.webgui.com:16315/ibm/console/webtop/  
AELView?filtertype=global&filtername=Last10Mins  
&view=DefaultNoSevAckCols&datasource=NCOMS,NILKA
```

Example for launching the AEL with a transient filter and a view, and a single data source

```
http://home.webgui.com:16315/ibm/console/  
AELView?sql="Node+++'PE_1'+AND+FirstOccurrence+>=+1240000000  
+AND+LastOccurrence+<+1240000000+%2B+(7+%2B+1)+*+86400"  
&transientname="CriticalThreshold"  
&view=PredictiveEventsView&datasource=NCOMS
```

Example for launching the Lightweight Event List (LEL) with a filter and view

```
http://home.webgui.com:16315/ibm/console/webtop/lwse1/  
lwse1.jsp?filtertype=global&filtername=Last10Mins  
&view=DefaultNoSevAckCols&datasource=NCOMS
```

Example for launching the Table View with a filter and view

```
http://home.webgui.com:16315/ibm/console/webtop/TableView/?map=E-  
Commerce&filtertype=global&filtername=Last10Mins  
&view=DefaultNoSevAckCols&datasource=NCOMS&maxrows=35
```

Example for launching a map page

```
http://home.webgui.com:16315/ibm/console/webtop/Example_Geographic
```

Setting user access to the Inline Frame portlet

If you plan to create content, for example maps, on Inline Frame portlets, you must grant access to the portlet to all non-administrative users.

By default, read-writer users and read-only users cannot access the Inline Frame portlet. Only administrative users, that is, users with the ncw_admin role, can access the portlet.

To give read-writer and read-only users access to the Inline Frame portlet:

1. In the navigation pane, click **Settings > Portlet Management**.
2. In the Portlet Management page, click **Uncategorized Portlets > Inline Frame**.

3. Click **Roles with Access to this Portlet** and click **Add**.
4. From the list, select the roles that you want to give access to the portlet:
 - For read-write users, select **ncw_user** and **netcool_rw**.
 - For read-only users, select **ncw_user** and **netcool_ro**.
5. Click **Add** and click **Save**.

Enabling multiple logins

Configure the Tivoli Integrated Portal to allow multiple users to log in using the same user ID and password.

1. Log in as an administrative user.
2. Navigate to:
`tip_home_dir/profiles/TIPProfile/config/cells/TIPCell/applications/isc.ear/deployments/isc/isc-lite.war/WEB-INF/`
3. Edit `consoleProperties.xml`.
4. Locate the property with a `id` attribute of **ENABLE_CONCURRENT_LOGIN** and set its value to `true`.
5. Save the file and exit from the text editor.
6. Restart the server.

Related tasks

“Restarting the server”

Installing, configuring, and using Tivoli Common Reporting

Tivoli Common Reporting V2.1 is provided as an optional part of the Tivoli Netcool/OMNIbus V7.3.1 installation package. Tivoli Common Reporting is distributed as a compressed file that is available on CD, or that you can download from the IBM Passport Advantage Online Web site.

The boxed product package contains the CD that you can use to install Tivoli Common Reporting on your operating system.

For more information about installing, configuring, and using Tivoli Common Reporting, see the *Tivoli Common Reporting Information Center* at http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.tivoli.tcr.doc_21/ctcr_prodooverview.html.

Related tasks

“Obtaining the installation package” on page 174

Restarting the server

After customization and configuration activities you might need to restart the Web GUI server.

Restart the server after or while carrying out any of the following actions on your Web GUI server:

- Modifications to any of the following files:
 - `server.init`
 - `ncwDataSourceDefinitions.xml`
 - `virtualhosts.xml`
 - `deployment.xml`
 - `security.xml`

- winconfig.xml
- Any properties file in the *tip_home_dir*/tip/properties directory
- Setting up a load balancing cluster
- Adding a node to a load balancing cluster
- Adding or changing user registries
- Backing up and restoring the Web GUI
- Copying configurations from another Web GUI server
- Configuring encryption
- Configuring single sign-on
- Configuring LDAP or Active Directory and their connections

If you do not use the timed tasks facility in the `server.init` file, you also need to restart the server after changing any files in the following directories in *webgui_home_dir*/etc:

- configstore
- cgi-bin
- charts
- charts/definitions
- templates and all the directories it holds

To restart the server:

1. On the command-line interface, change to the *tip_home_dir*/profiles/TIPProfile/bin.
2. Stop the server:

- `Linux` `UNIX` `stopServer.sh server1`

Attention: Linux and Unix systems prompt you to supply the user name and password of the administrative user.

- `Windows` `stopServer.bat server1`

Wait a moment for the server to completely shut down.

3. Start the server:

- `Linux` `UNIX` `startServer.sh server1`
- `Windows` `startServer.bat server1`

Chapter 27. Example Tivoli Netcool/OMNIbus installation scenarios (basic, failover, and desktop architectures)

Some example Tivoli Netcool/OMNIbus installation scenarios for the basic, failover, and desktop architectures are described here. Each example architecture builds on the previous one.

Example Tivoli Netcool/OMNIbus basic architecture

The example Tivoli Netcool/OMNIbus basic architecture uses a single Syslog probe to monitor an application that writes debug messages to the syslog daemon on its host computer.

The Syslog probe forwards events to the ObjectServer running on a second host computer. Users view the events using a Windows desktop on a third host. The ObjectServer and probe run under process control.

This example adds some fields to the alerts.status table after the system has been in operation.

Deploying the basic architecture

The Tivoli Netcool/OMNIbus basic architecture comprises the following components: ObjectServer, process agent, Syslog probe, and event list.

The ObjectServer (NETCOOLPRI) and process agent run on a Solaris computer with the host name nchost01. The event list runs on a Windows computer with the host name ncdesktop. The installation monitors an application that writes debug messages to the syslog daemon on a Solaris computer with the host name targethost.

The Tivoli Netcool/OMNIbus basic architecture is shown in the following figure.

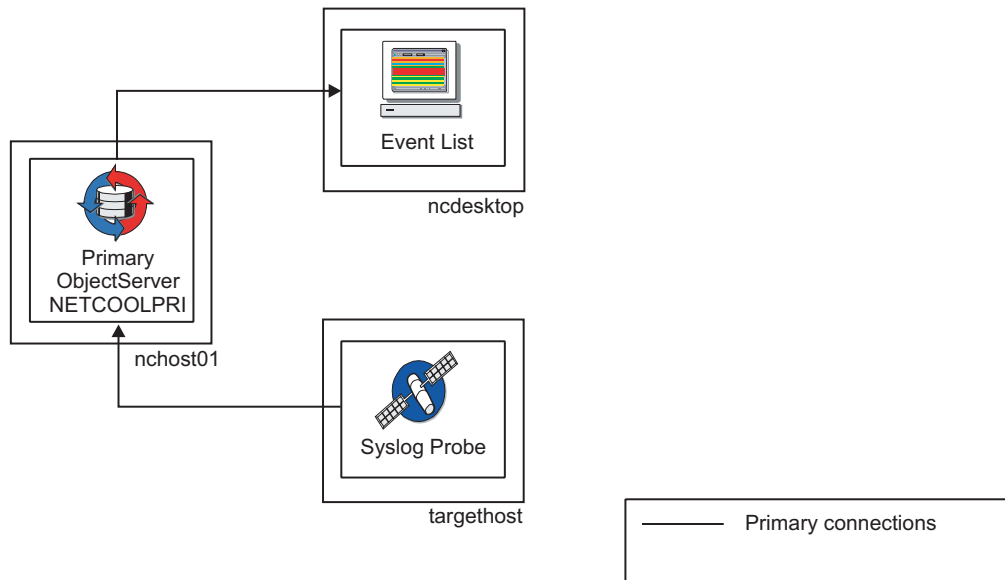


Figure 19. Example Tivoli Netcool/OMNibus basic architecture

Prerequisites for the basic architecture

A UNIX user account called `netcool` must exist on each host. This user must be a member of the UNIX group `ncadmin` in order to use the process control (`nco_pa_*`) utilities.

The following environment variables must be set for the `netcool` user:

- `$NCHOME - /opt/IBM/tivoli/netcool`
- `$PATH - $NCHOME/omnibus/bin:$PATH`

This deployment assumes that default directories are used.

Step 1: Installing the ObjectServer and process agent

Use this step to install the ObjectServer and the process agent.

On the `nchost01` computer, proceed as follows:

1. Download the Tivoli Netcool/OMNibus installation bundle for Solaris from Passport Advantage, and extract the files.
2. Run the installation program `./install.bin`.
3. Agree to the license conditions and accept the default installation location.
4. Select and install **Process Agent, Servers, Desktop, Confpack, and Probe Support**.

The ObjectServer and process agent are installed on the same Solaris computer in the `$NCHOME` location. Additional features that are required for configuring the system are also installed.

Step 2: Installing the probes

To install probes on the nchost01 computer, download the relevant probes from Passport Advantage. The accompanying README.txt file, which is available for each probe, describes how to install the probe.

Tip: Include the Simnet probe (nco_p_simnet) in your download selection. This probe sends simulated events to the ObjectServer, and is required for test purposes.

The probes are installed in the \$NCHOME/omnibus/probes location.

Step 3: Installing the event list

Use this step to install the event list.

On the Windows computer (ncdesktop), proceed as follows:

1. Download the Tivoli Netcool/OMNIBus installation bundle for Windows from Passport Advantage, and extract the files.
2. Double-click the install.exe file to start the installation.
3. Agree to the license conditions and accept the default installation location.
4. Select **Desktop** as the only installable feature.

When you complete the installation, the desktop tools, including the event list, are installed on the Windows computer.

Step 4: Configuring communications

Use this step to configure communications between the Tivoli Netcool/OMNIBus components.

On the ObjectServer computer (nchost01), proceed as follows:

1. Run the following command to open the Server Editor window:
\$NCHOME/omnibus/bin/nco_xigen
2. Configure the ObjectServer and process agent communications settings by specifying the example Server Editor settings, as shown in the following table. Click **Add** after each set of entries. (The SSL value of 0 is available by default, and is shown as a blank value in the display area of the Server Editor.)

Table 104. Server Editor settings - basic architecture

Server	Hostname	Port	SSL
NCOOS_PA	nchost01	4200	0
NCOPR_PA	targethost	4300	0
NETCOOLPRI	nchost01	4100	0

3. Apply your changes and close the Server Editor. This creates the file \$NCHOME/etc/interfaces.solaris2, containing your communications information.

Step 5: Creating the ObjectServer

Use this step to create and start an ObjectServer called NETCOOLPRI.

On the nchost01 computer, proceed as follows:

1. Create the ObjectServer by running the following command:
`$NCHOME/omnibus/bin/nco_dbinit -server NETCOOLPRI`
2. Start the ObjectServer by running the following command:
`$NCHOME/omnibus/bin/nco_objserv -name NETCOOLPRI`

You now have a running ObjectServer named NETCOOLPRI.

Step 6: Testing the system

Use this step to test the ObjectServer and event list by running a Simnet probe. The Simnet probe sends simulated events to the ObjectServer.

Proceed as follows:

1. On the nchost01 computer, start the Simnet probe by running the following command:
`$NCHOME/omnibus/probes/nco_p_simnet -server NETCOOLPRI`
2. Start the event list on the Windows computer ncdesktop:
`%NCHOME%\omnibus\desktop\NCOEvent.exe`
Log on to NETCOOLPRI as the root user with the corresponding password. You will see simulated events in the event list.

Note: The default root user password is an empty string. This is not the system root user and password.

You have now shown that the ObjectServer can process events and send them to the event list.

Step 7: Installing and configuring the Syslog probe and the Syslog daemon

This step consists of a number of substeps, which you must perform on the targethost computer.

1. "Configuring the Syslog daemon"
2. "Installing and configuring the Syslog probe" on page 575
3. "Testing the Syslog probe" on page 575

Configuring the Syslog daemon

Use this step to configure the Syslog daemon to log debug messages from the target application. Proceed as follows:

1. On the targethost computer, enter the following command:
`touch /var/log/ncolog`
2. Edit the /etc/syslog.conf file. Add the following line:
`*.debug /var/log/ncolog`

The separator between the selector and the action *must* be a tab character for the entry to be accepted by syslog.

Note: This line must not be the first line of the `/etc/syslog.conf` file. If it is, it will activate a bug in the `syslogd` daemon, where it attempts a check on the first file in the first entry in the `/etc/syslog.conf` file, and this will make the `syslog` system unstable. Also note that some implementations of `syslogd` are limited to 20 valid entries in the `/etc/syslog.conf` file.

3. Restart the `syslog` daemon. Find the process identifier of the `syslog` daemon and issue a `kill -HUP` command to that process. For example:

```
targethost# ps-ef | grep syslogd
root 169 1 0 Dec 12 ? 0:47 /usr/sbin/syslogd
root 26429 25748 0 16:13:13 pts/13 0:00 grep syslogd
targethost# kill -HUP 169
```

This causes the `Syslog` daemon to re-read the `/etc/syslog.conf` file.

4. Check that the `syslog` daemon is sending messages to the `/var/log/ncolog` file, by using the command:

```
logger -p debug testing
more /var/log/ncolog
```

The following message appears at the end of the log file:

```
timestamp targethost netcool: testing
```

The `syslog` daemon is now configured to log debug messages from the target application.

Installing and configuring the Syslog probe

To install and configure the `Syslog` probe:

1. On the `targethost` computer, download the Tivoli Netcool/OMNIBus installation bundle and the `Syslog` probe (`nco_p_syslog`) for Solaris from Passport Advantage.
2. Install Tivoli Netcool/OMNIBus and select **Administrator**, **Process Agent**, and **Probe Support** as installable features.
3. Install the probe as documented in the `README.txt` file, provided for each probe. This file shows how to install the probe in the various installation modes.

The probe is installed in the `$NCHOME/omnibus/probes` location.

4. Copy the `$NCHOME/etc/interfaces.solaris2` file from the computer running the ObjectServer (`nchost01`) to the `$NCHOME/etc` directory on the `targethost` computer.
5. Edit the `$NCHOME/omnibus/probes/solaris2/syslog.props` file. Copy and paste the **Manager**, **Server**, and **LogFile** properties to the end of the file. This enables you to keep a commented default configuration within the file.
6. Uncomment and edit the pasted **Manager**, **Server**, and **LogFile** properties. Use the following values:

```
Manager : 'Syslog@targethost'
Server  : 'NETCOOLPRI'
LogFile : '/var/log/ncolog'
```

7. Start the probe, using the command:

```
$NCHOME/omnibus/probes/nco_p_syslog &
```

You have finished installing and configuring the `Syslog` probe.

Testing the Syslog probe

To ensure the `Syslog` probe is working properly:

1. On the targethost computer, check that the Syslog probe is reading messages from the /var/log/ncoolog file, using the command:
`logger -p debug "testing the probe"`
2. Log in to the ObjectServer using the SQL interactive interface, **nco_sql**. Use the following command:
`$NCHOME/omnibus/bin/nco_sql -server NETCOOLPRI -user root`
3. Enter the ObjectServer root password at the prompt.
4. Determine whether an alert with a summary of testing the probe is present in the alerts.status table. To do this, enter:
`1> select * from alerts.status where Summary like 'testing the probe';`
`2> go`
5. If the Syslog probe has read the event and forwarded it to the ObjectServer, the last line of text output reads as follows:

(1 row affected)
6. You can also confirm this by checking the event list.

You have finished testing the Syslog probe.

Step 8: Configuring process control

Configure process control on the computers running the primary ObjectServer (nchost01) and Syslog probe (targethost).

Computer running the ObjectServer

Use this procedure to configure a process agent called NCOOS_PA on nchost01. Proceed as follows:

1. On nchost01, edit the process agent configuration file \$NCHOME/omnibus/etc/nco_pa.conf. The complete configuration file for the process agent NCOOS_PA is as follows:

```
nco_process 'ObjectServer'
{
  Command '$NCHOME/omnibus/bin/nco_objserv -name NETCOOLPRI -pa NCOOS_PA' run as 0
  Host='nchost01'
  Managed=true
  RestartMsg='The ObjectServer has been restarted'
  AlertMsg='The ObjectServer has gone down'
  RetryCount=0
  ProcessType=PaPA_AWARE
}
nco_service 'Omnibus'
{
  ServiceType=Master
  ServiceStart=Auto
  process 'ObjectServer' NONE
}
nco_routing
{
  host 'nchost01' 'NCOOS_PA'
}
```

For the ObjectServer process defined in the first section of the nco_pa.conf file, the first lines define the command line used to start the process and the host it is on. The Managed item is set to true so that process control restarts the ObjectServer process if it stops for any reason.

The Omnibus service contains the ObjectServer process. The ServiceType entry Master specifies that the Omnibus service should be the first service to start. The

ServiceStart entry Auto specifies that the Omnibus service should start automatically after the process control daemon starts.

The nco_routing section notifies each process agent of the location of the other process agents.

2. Stop the ObjectServer. You must also stop the probe that is running on the targethost computer.
3. Start the process control daemon, using the command:
`$NCHOME/omnibus/bin/nco_pad -name NCOOS_PA`
4. To check that the ObjectServer is running, enter:
`ps -ef | grep nco_objserv`

The ObjectServer is now running under process control.

Computer running the Syslog probe

To configure a process agent called NCOPR_PA on targethost:

1. On targethost, edit the `$NCHOME/omnibus/etc/nco_pa.conf` file. The complete configuration file for the process agent NCOPR_PA is shown below.

```
nco_process 'SyslogProbe'
{
  Command '$NCHOME/omnibus/probes/nco_p_syslog' run as 0
  Host='targethost'
  Managed=true
  RestartMsg='The Syslog Probe has been restarted'
  AlertMsg='The Syslog Probe has gone down'
  RetryCount=0
  ProcessType=PaPA_AWARE
}
nco_service 'Probes'
{
  ServiceType=Master
  ServiceStart=Auto
  process 'SyslogProbe' NONE
}
nco_routing
{
  host 'targethost' 'NCOPR_PA'
}
```

For the SyslogProbe process defined in the first section of the above file, the first lines define the command line used to start the process and the host it is on. The Managed item is set to true so that process control will restart the process if it stops for any reason.

The Probes service contains the SyslogProbe process. The ServiceType entry Master specifies that the SyslogProbe service should be the first service to start. The ServiceStart entry Auto specifies that the SyslogProbe service should start automatically after the process control daemon starts.

The nco_routing section notifies each process agent of the location of the other process agents.

2. Start the process control daemon, using the command:
`$NCHOME/omnibus/bin/nco_pad -name NCOPR_PA`
3. To check that the probe is running, enter:
`ps -ef | grep nco_p_syslog`

The Syslog probe is now running under process control.

Step 9: Adding columns to the ObjectServer

Use this step to add columns to the ObjectServer NETCOOLPRI.

Proceed as follows:

1. On the targethost computer, stop the probe, using the command:
`$NCHOME/omnibus/bin/nco_pa_stop -server NCOPR_PA -service Probes`
2. Using Netcool/OMNIBus Administrator or **nco_sql**, add the following fields to the alerts.status table:
`CustomerID int,
CustomerContact varchar(1024),
ReferenceCode varchar(128),`
If you use **nco_sql** to add the fields, use the ALTER TABLE command. For information about using Netcool/OMNIBus Administrator and the ALTER TABLE command, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.
3. On the targethost computer, restart the probe, using the command:
`$NCHOME/omnibus/bin/nco_pa_start -server NCOPR_PA -service Probes`
The probe uses a recovery file that records the last log file entry it read. It reads this and then reads the /var/log/ncolog file from the next entry.
4. Check that the probe is running correctly under process control, using the command:
`$NCHOME/omnibus/bin/nco_pa_status -server NCOPR_PA -user netcool`
5. Check that the Syslog probe is reading messages from the /var/log/ncolog file, using the command:
`logger -p debug "testing the new tables"`
6. Restart the event list on the Windows computer ncdesktop and log on to NETCOOLPRI. The event list will contain an event with the summary testing the new tables.

The configuration of the Tivoli Netcool/OMNIBus basic architecture is complete.

Next steps

After you have completed the steps for deploying the basic architecture, you can optionally install the Web GUI.

Important: If you want to proceed by deploying the basic failover architecture, do not install the Web GUI at this point. Instead, install the Web GUI after you have completed the steps for deploying the basic failover architecture.

Proceed as follows:

1. Download the Web GUI installation bundle from Passport Advantage and extract the files.
2. Run the installation program:
 - **UNIX** **Linux** `./install.sh`
 - **Windows** `install.exe`
3. Agree to the license conditions and accept the default installation location.
4. On the Choose Install Set panel, select **Default installation**.
5. On the ObjectServer Information panel, specify the following information about the ObjectServer that you created in “Step 5: Creating the ObjectServer” on page 574:

User ID

Type the user name of the ObjectServer administrator.

Password

Type the password of the ObjectServer administrator.

Confirm Password

Retype the password of the ObjectServer administrator.

Name Type NETCOOLPRI.

Primary Hostname

Type nchost01.

Primary Port

Type 4100.

Example Tivoli Netcool/OMNIbus basic failover architecture

You can add a failover (backup) ObjectServer to the example basic architecture. Alert data from the primary ObjectServer is replicated in the backup ObjectServer through a bidirectional ObjectServer Gateway. If a connection to the primary ObjectServer fails, the clients attempt to connect to the backup ObjectServer.

The instructions in this example assume that you have an existing basic Tivoli Netcool/OMNIbus architecture.

Deploying the basic failover architecture

The Tivoli Netcool/OMNIbus basic failover architecture comprises all components from the basic architecture, and the following additional components: backup ObjectServer and bidirectional ObjectServer Gateway.

The backup ObjectServer and bidirectional ObjectServer Gateway run on a single Solaris computer with the host name nchost02.

The Tivoli Netcool/OMNIbus basic failover architecture is shown in the following figure.

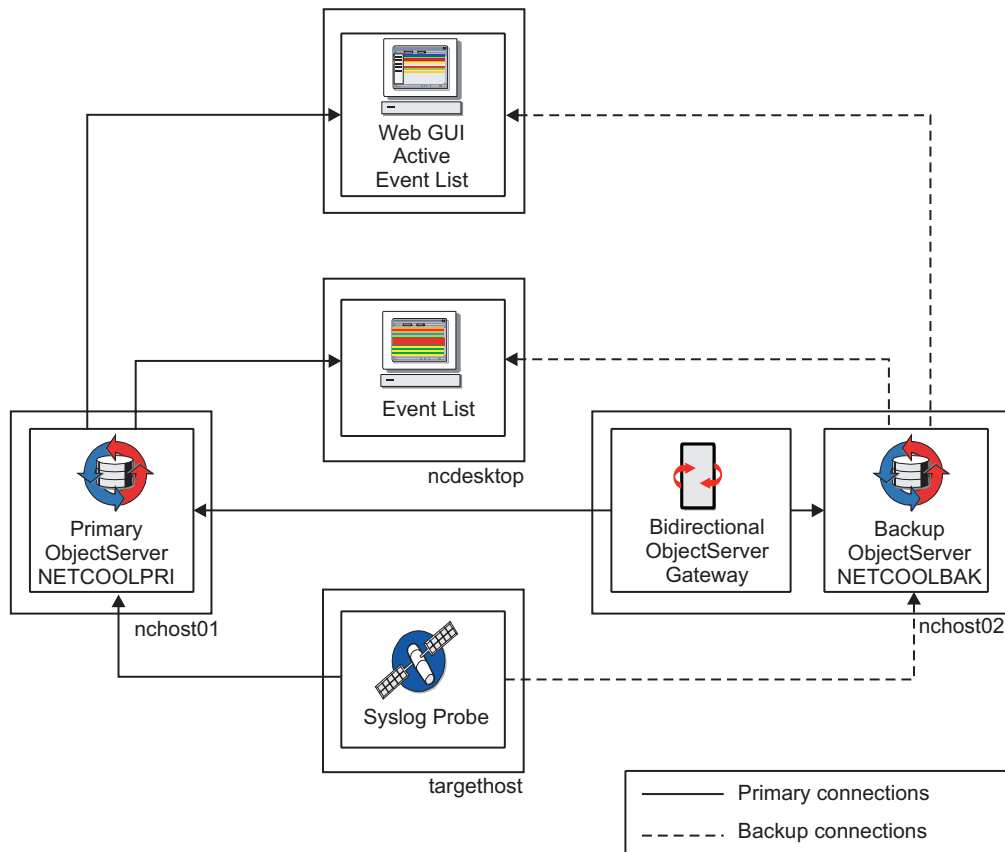


Figure 20. Example Tivoli Netcool/OMNIBus basic failover architecture

Related reference

“Deploying the basic architecture” on page 571

Prerequisites for the basic failover architecture

A UNIX user account called `netcool` must exist on each host. This user must be a member of the UNIX group `ncadmin` in order to use the process control (`nco_pa_*`) utilities.

The following environment variables must be set for the `netcool` user:

- `$NCHOME` - `/opt/IBM/tivoli/netcool`
- `$PATH` - `$NCHOME/omnibus/bin:$PATH`

This deployment assumes that default directories are used.

Step1: Installing the basic architecture

Before you can add any failover components to your installation, you must first install the Tivoli Netcool/OMNIBus basic architecture.

Related reference

“Example Tivoli Netcool/OMNIBus basic architecture” on page 571

Step 2: Installing the backup ObjectServer and ObjectServer Gateway

This step installs the backup ObjectServer and ObjectServer Gateway.

On the backup computer (nchost02), proceed as follows:

1. Download the Tivoli Netcool/OMNIBus installation bundle for Solaris from Passport Advantage and extract the files. Alternatively, copy the extracted files from the primary ObjectServer computer (nchost01).
2. Run the installation program `./install.bin`. Select and install **Gateways**, **Process Agent**, **Servers**, and **Confpack**.

When complete, the ObjectServer, ObjectServer Gateway, process agent, and other required features are installed on nchost02.

Step 3: Configuring communications

Use this step to configure communications between Tivoli Netcool/OMNIBus components.

Proceed as follows:

1. On the computer on which the *primary* ObjectServer is running (nchost01), run the following command to open the Server Editor window:
`$NCHOME/omnibus/bin/nco_xigen`
2. Configure the additional communications settings. The complete set of Server Editor entries is shown in the following table.

Table 105. Server Editor settings - basic failover architecture

Server	Hostname	Port	SSL
FAIL_GATE	nchost02	4500	0
NCOBK_PA	nchost02	4600	0
NCOOS_PA	nchost01	4200	0
NCOPR_PA	targethost	4300	0
NETCOOLBAK	nchost02	4400	0
NETCOOLPRI	nchost01	4100	0
NETCOOLVIR	nchost01	4100	0
Backup 1:	nchost02	4400	0

3. Apply your changes and close the Server Editor. This creates the file `$NCHOME/etc/interfaces.solaris2`, containing your communications information.
4. Copy the file `$NCHOME/etc/interfaces.solaris2` to the `$NCHOME/etc` directory on the backup computer (nchost02) and the Syslog probe computer (targethost).
5. For the Windows computer (ncdesktop), use the Server Editor to enter the additional communications information.

The primary, backup, probe, and desktop computers now have the same communications information.

Step 4: Creating and configuring the backup ObjectServer

Use this step to create an ObjectServer called NETCOOLBAK.

On the backup computer (nchost02), proceed as follows:

1. Create the ObjectServer by running the following command:
`$NCHOME/omnibus/bin/nco_dbinit -server NETCOOLBAK`
2. Use the **nco_confpack** utility to export a configuration package from the NETCOOLPRI ObjectServer.
3. Use the **nco_confpack** utility to import the configuration package into the NETCOOLBAK ObjectServer.
4. Edit the `$NCHOME/omnibus/etc/NETCOOLBAK.props` file. Change the **BackupObjectServer** property to True:
`BackupObjectServer: True`

The NETCOOLBAK ObjectServer is now configured as a backup ObjectServer. It is started later using process control.

Related concepts

Chapter 11, "Importing and exporting ObjectServer configurations," on page 301

Step 5: Configuring the bidirectional ObjectServer Gateway

Use this step to configure the bidirectional ObjectServer Gateway mapping.

On the backup computer (nchost02), proceed as follows:

1. Create the directory `$NCHOME/omnibus/gates/FAIL_GATE`.
2. Copy all of the files in `$NCHOME/omnibus/gates/objserv_bi` to the `$NCHOME/omnibus/gates/FAIL_GATE` directory.
3. Rename the `$NCHOME/omnibus/gates/FAIL_GATE/objserv_bi.map` file to `FAIL_GATE.map`.
4. Edit the `FAIL_GATE.map` file.

A partial default bidirectional ObjectServer Gateway mapping is shown below, with some additional custom fields (in bold) that you must add to match the NETCOOLPRI and NETCOOLBAK ObjectServers:

```
CREATE MAPPING StatusMap
(
  'Identifier' = '@Identifier' ON INSERT ONLY,
  'Node' = '@Node' ON INSERT ONLY,
  'NodeAlias' = '@NodeAlias' ON INSERT ONLY,
  ...
  ...
  'CustomerID' = '@CustomerID' ON INSERT ONLY,
  'CustomerContact' = '@CustomerContact' ON INSERT ONLY,
  'ReferenceCode' = '@ReferenceCode' ON INSERT ONLY,
  'ServerName' = '@ServerName' ON INSERT ONLY,
  'ServerSerial' = '@ServerSerial' ON INSERT ONLY
);
```

5. Rename the `$NCHOME/omnibus/gates/FAIL_GATE/objserv_bi.props` file to `FAIL_GATE.props`.
6. Edit the following entries in the `FAIL_GATE.props` file:

```
# Common Netcool/OMNIBus Properties.
MessageLog : '$OMNIHOME/log/FAIL_GATE.log'

# Common Gateway Properties.
Gate.MapFile : '$OMNIHOME/gates/FAIL_GATE/FAIL_GATE.map'
Gate.StartupCmdFile : '$OMNIHOME/gates/FAIL_GATE/objserv_bi.startup.cmd'
```

```
# Bidirectional ObjectServer Gateway Properties.
Gate.ObjectServerA.Server : 'NETCOOLPRI'
Gate.ObjectServerA.Username : 'root'
Gate.ObjectServerA.Password : ''
Gate.ObjectServerA.TblReplicateDefFile:
'$OMNIHOME/gates/FAIL_GATE/objserv_bi.objectservera.tblrep.def'

Gate.ObjectServerB.Server : 'NETCOOLBAK'
Gate.ObjectServerB.Username : 'root'
Gate.ObjectServerB.Password : ''
Gate.ObjectServerB.TblReplicateDefFile:
'$OMNIHOME/gates/FAIL_GATE/objserv_bi.objectserverb.tblrep.def'
```

7. Copy the `$NCHOME/omnibus/gates/FAIL_GATE/FAIL_GATE.props` file to `$NCHOME/omnibus/etc/FAIL_GATE.props`.

The bidirectional ObjectServer Gateway is now configured to exchange alert data between the NETCOOLPRI and NETCOOLBAK ObjectServers.

Step 6: Configuring the Syslog probe

Use this step to configure the Syslog probe to fail over to the backup ObjectServer.

Proceed as follows:

1. Log in to the computer on which the Syslog probe is running (targethost).
2. Stop the Syslog probe, using the command:
`$NCHOME/omnibus/bin/nco_pa_stop -server NCOPR_PA -service Probes`
3. Edit the `$NCHOME/omnibus/probes/solaris2/syslog.props` file. Copy and paste the **ServerBackup** and **PollServer** properties to the end of the file. This enables you to keep a commented default configuration in the file.
4. Uncomment and edit the pasted **ServerBackup** and **PollServer** properties. Use the following values:

```
NetworkTimeout : 30
PollServer : 30
ServerBackup : 'NETCOOLBAK'
```

Note: Setting the **NetworkTimeout** property to 30 seconds should be appropriate for most networks; however, you might need to increase this value if the probe is continually disconnecting from the ObjectServer. This property overrides the operating system standard TCP timeout (which for Solaris is between 7 and 12 minutes).

5. Restart the probe, using the command:
`$NCHOME/omnibus/bin/nco_pa_start -server NCOPR_PA -service Probes`

The Syslog probe is now running again under process control and is using the updated properties.

Step 7: Configuring process control on the backup computer

Use this step to configure a process agent called NCOBK_PA on nchost02.

For detailed information about process control, see the *IBM Tivoli Netcool/OMNIBus Administration Guide*.

Proceed as follows:

1. Log in to the computer running the backup ObjectServer (nchost02).
2. Edit the `$NCHOME/omnibus/etc/nco_pa.conf` file.
3. Add the process, service, and routing information for the backup ObjectServer, as shown below:

```

nco_process 'Bak_ObjectServer'
{
  Command '$NCHOME/omnibus/bin/nco_objserv -name NETCOOLBAK -pa NCOBK_PA' run as 0
  Host='nchost02'
  Managed=true
  RestartMsg='The backup ObjectServer has been restarted'
  AlertMsg='The backup ObjectServer has gone down'
  RetryCount=0
  ProcessType=PaPA_AWARE
}
nco_process 'Bi_Gate'
{
  Command '$NCHOME/omnibus/bin/nco_g_objserv_bi -name FAIL_GATE' run as 0
  Host='nchost02'
  Managed=true
  RestartMsg='The bidirectional gateway has been restarted'
  AlertMsg='The bidirectional gateway has gone down'
  RetryCount=5
  ProcessType=PaPA_AWARE
}
nco_service 'Bak_OS'
{
  ServiceType=Master
  ServiceStart=Auto
  process 'Bak_ObjectServer' NONE
  process 'Bi_Gate' 'Bak_ObjectServer'
}
nco_routing
{
  host 'nchost02' 'NCOBK_PA'
}

```

4. Start the process control daemon, using the command:
`$NCHOME/omnibus/bin/nco_pad -name NCOBK_PA`
5. To test that the backup ObjectServer and bidirectional ObjectServer Gateway are running under process control, enter:
`$NCHOME/omnibus/bin/nco_pa_status -server NCOBK_PA -user netcool`
6. Enter the password at the prompt.

The backup ObjectServer and bidirectional ObjectServer Gateway should be reported as running under process control.

The configuration of the Tivoli Netcool/OMNIBus basic failover architecture is complete.

Next steps

After you have completed the steps for deploying the failover architecture, you can install the Web GUI.

Proceed as follows:

1. Download the Web GUI installation bundle from Passport Advantage and extract the files.
2. Run the installation program:
 - `UNIX Linux ./install.sh`
 - `Windows install.exe`
3. Agree to the license conditions and accept the default installation location.
4. On the Install Set panel, select **Default installation**.
5. On the ObjectServer Information panel, specify the following information about the ObjectServer that you created in Basic architecture step 5: Creating the ObjectServer:

User ID

Type the user name of the ObjectServer administrator.

Password

Type the password of the ObjectServer administrator.

Confirm Password

Retype the password of the ObjectServer administrator.

Name Type NETCOOLPRI.

Primary Hostname

Type nchost01.

Primary Port

Type 4100.

6. On the same panel, select **Enable Secondary Server for Failover** and then specify the following information about the ObjectServer that you created in “Step 4: Creating and configuring the backup ObjectServer” on page 582.

User ID

Type the user name of the ObjectServer administrator.

Password

Type the password of the ObjectServer administrator.

Confirm Password

Retype the password of the ObjectServer administrator.

Name Type NETCOOLBAK.

Primary Hostname

Type nchost02.

Primary Port

Type 4400.

Example Tivoli Netcool/OMNIBus desktop ObjectServer architecture

You can add a desktop ObjectServer to the example basic failover architecture. A desktop ObjectServer increases the performance of a standard ObjectServer that frequently experiences heavy loads from users' desktops.

Related concepts

Chapter 12, “Setting up desktop ObjectServers,” on page 327

Related reference

“Example Tivoli Netcool/OMNIBus basic failover architecture” on page 579

Deploying the desktop ObjectServer architecture

The Tivoli Netcool/OMNIBus desktop ObjectServer architecture comprises all components from the basic failover architecture and the following additional components: desktop ObjectServer and unidirectional ObjectServer Gateway.

In the following example, the desktop ObjectServer and unidirectional ObjectServer Gateway are installed on a single Solaris computer with the host name ncdesk.

The Tivoli Netcool/OMNIBus desktop ObjectServer architecture is shown in the following figure.

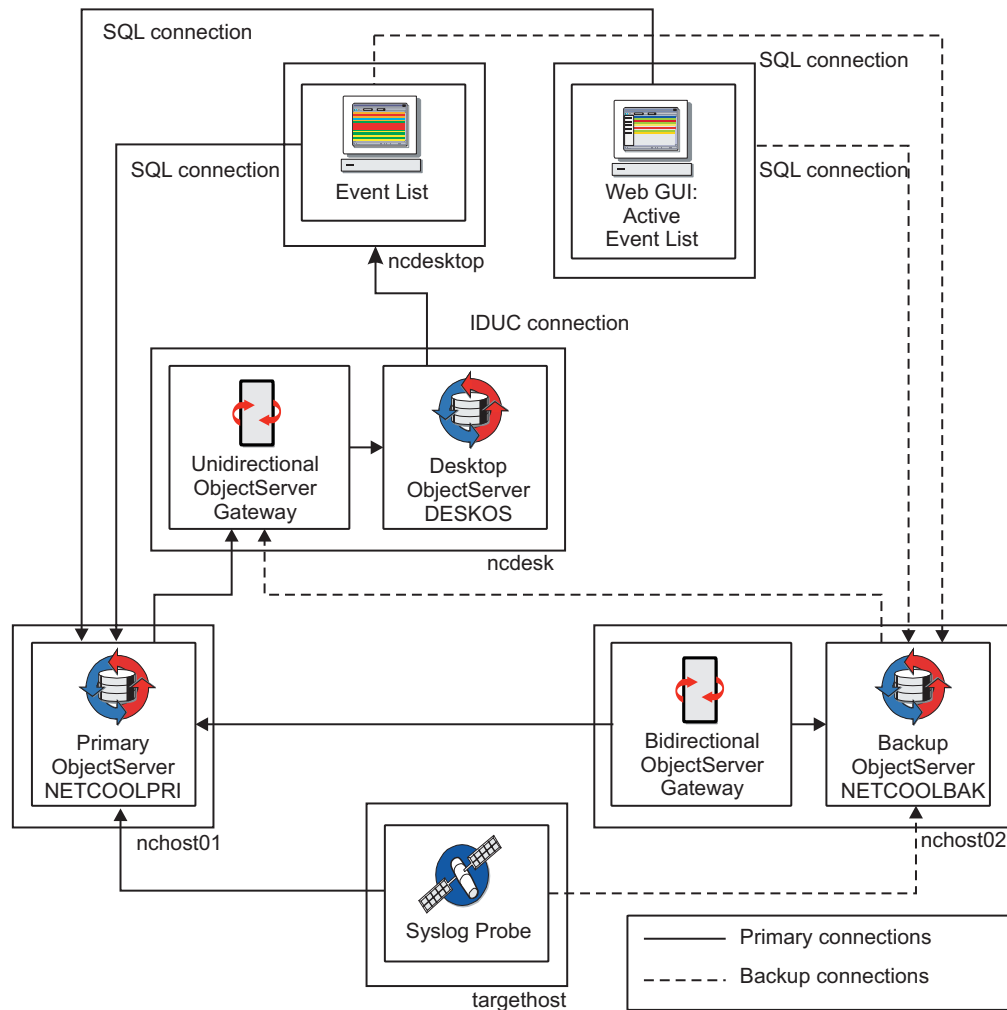


Figure 21. Example Tivoli Netcool/OMNIBus desktop ObjectServer architecture

Related reference

“Deploying the basic failover architecture” on page 579

Prerequisites for the desktop ObjectServer architecture

A UNIX user account called `netcool` must exist on each host. This user must be a member of the UNIX group `ncadmin` in order to use the process control (`nco_pa_*`) utilities.

The following environment variables must be set for the `netcool` user:

- `$NCHOME` - `/opt/IBM/tivoli/netcool`
- `$PATH` - `$NCHOME/omnibus/bin:$PATH`

This deployment assumes that default directories are used.

Step1: Installing the basic failover architectures

Before you can add any desktop ObjectServer components to your installation, you must first install the basic Tivoli Netcool/OMNIBus failover architecture.

Related reference

“Example Tivoli Netcool/OMNIBus basic failover architecture” on page 579

Step 2: Installing the desktop ObjectServer and unidirectional gateway

Use this step to install the desktop ObjectServer and unidirectional ObjectServer Gateway.

On the desktop ObjectServer computer (ncdesk), proceed as follows:

1. Download the Tivoli Netcool/OMNIBus installation bundle for Solaris from Passport Advantage and extract the files. Alternatively, copy the extracted files from the primary ObjectServer computer (nchost01).
2. Run the installation program `./install.bin`. Select and install **Gateways**, **Process Agent**, **Servers**, and **Confpack**.

The ObjectServer, unidirectional ObjectServer Gateway, and process control files are installed on the same Solaris computer.

Step 3: Configuring component communications

Use this step to configure communications between Tivoli Netcool/OMNIBus components.

Proceed as follows:

1. On the computer *on which the primary ObjectServer is running* (nchost01), run the following command to open the Server Editor window:
`$NCHOME/omnibus/bin/nco_xigen`
2. Configure the additional communications settings. The complete set of Server Editor entries is shown in the following table.

Table 106. Server Editor settings - desktop ObjectServer architecture

Server	Hostname	Port	SSL
DESKOS	ncdesk	4700	0
DSD_GATE	ncdesk	4800	0
FAIL_GATE	nchost02	4500	0
NCOBK_PA	nchost02	4600	0
NCOOS_PA	nchost01	4200	0
NCOPR_PA	targethost	4300	0
NCOSDESK_PA	ncdesk	4900	0
NETCOOLBAK	nchost02	4400	0
NETCOOLPRI	nchost01	4100	0
NETCOOLVIR	nchost01	4100	0
Backup 1:	nchost02	4400	0

3. Apply your changes and close the Server Editor. This creates the file `interfaces.solaris2`, which contains your communications information.

4. Copy the file `$NCHOME/etc/interfaces.solaris2` to the `$NCHOME/etc` directory on `nhost02`, `ncdesk`, and `targethost`.
5. For the Windows computer (`ncdesktop`), use the Server Editor to enter the correct communications information.

All computers in your Tivoli Netcool/OMNIBus installation now have the same communications information.

Step 4: Creating and configuring the desktop ObjectServer

Use this step to create and configure a desktop ObjectServer called `DESKOS`.

Proceed as follows:

1. On the desktop ObjectServer computer (`ncdesk`), enter the following command:
`$NCHOME/omnibus/bin/nco_dbinit -desktopserver -server DESKOS`
 The desktop ObjectServer `DESKOS` is created.
2. Use the **nco_confpack** utility to export a configuration package from the `NETCOOLPRI` ObjectServer. On `nhost01`, enter the following command:
`$NCHOME/omnibus/bin/nco_confpack -export -package /tmp/netcoolpri.jar -server NETCOOLPRI -user root`
3. Copy the `/tmp/netcoolpri.jar` file to the `/tmp` directory on the desktop ObjectServer computer (`ncdesk`).
4. Start the ObjectServer by using the following command:
`$NCHOME/omnibus/bin/nco_objserv -name DESKOS`
5. Use the **nco_confpack** utility to import the configuration package into the `DESKOS` ObjectServer. On `ncdesk`, enter the following command:
`$NCHOME/omnibus/bin/nco_confpack -import -package /tmp/netcoolpri.jar -server DESKOS -user root`
6. Using Netcool/OMNIBus Administrator or **nco_sql**, insert a new row into the `master.national` table with the following values:
 - In the `KeyField` field, enter a key value of `0`.
 - In the `MasterServer` field, enter the name of the master ObjectServer (`NETCOOLPRI`).
 - In the `DualWrite` field, enter `1`.

If you use **nco_sql**, the insert command is:

```
insert into master.national
values (0,'NETCOOLPRI',1);
go
```

The desktop ObjectServer `DESKOS` is now configured and running.

Step 5: Configuring the unidirectional ObjectServer Gateway

Use this step to configure the unidirectional ObjectServer Gateway.

From the desktop ObjectServer computer (`ncdesk`), proceed as follows:

1. Create the directory `$NCHOME/omnibus/gates/DSD_GATE`.
2. Copy all the files in `$NCHOME/omnibus/gates/objserv_uni` to `$NCHOME/omnibus/gates/DSD_GATE`.
3. Rename the `$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.props` file to `DSD_GATE.props`.
4. Edit the following entries in the `DSD_GATE.props` file:


```
# Common Netcool/OMNIBus Properties.
MessageLog : '$OMNIBUSHOME/log/DSD_GATE.log'

# Common Gateway Properties.
Gate.MapFile : '$OMNIBUSHOME/gates/DSD_GATE/DSD_GATE.map'
Gate.StartupCmdFile : '$OMNIBUSHOME/gates/DSD_GATE/objserv_uni.startup.cmd'

# Unidirectional ObjectServer Gateway Properties.
Gate.Reader.Server : 'NETCOOLPRI'
Gate.Reader.Username : 'root'
Gate.Reader.Password : ''
Gate.Reader.FailbackEnabled : TRUE
Gate.Reader.Tbl.ReplicateDefFile:
'$OMNIBUSHOME/gates/DSD_GATE/objserv_uni.reader.tblrep.def'
Gate.Writer.Server : 'DESKOS'
Gate.Writer.Username : 'root'
Gate.Writer.Password : ''
```

5. Copy the \$NCHOME/omnibus/gates/DSD_GATE/DSD_GATE.props properties file to \$NCHOME/omnibus/etc/DSD.props.
6. Rename the \$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.map file to \$NCHOME/omnibus/gates/DSD_GATE/DSD_GATE.map.
7. Edit the DSD_GATE.map file. The fields and field order in the mapping must match the alerts.status table *exactly* in the primary and backup ObjectServers, including the additional custom fields.

You must also add the MasterSerial entry to the end of the StatusMap section. This provides unique identification of the events in the master ObjectServer.

A partial default unidirectional ObjectServer Gateway mapping is shown below, with the MasterSerial entry and the additional custom fields in bold:

```
CREATE MAPPING StatusMap
(
  'Identifier' = '@Identifier' ON INSERT ONLY,
  'Node' = '@Node' ON INSERT ONLY,
  'NodeAlias' = '@NodeAlias' ON INSERT ONLY,
  ...
  ...
  'CustomerID' = '@CustomerID' ON INSERT ONLY,
  'CustomerContact' = '@CustomerContact' ON INSERT ONLY,
  'ReferenceCode' = '@ReferenceCode' ON INSERT ONLY,
  ...
  ...
  'ServerName' = '@ServerName' ON INSERT ONLY,
  'ServerSerial' = '@ServerSerial' ON INSERT ONLY,
  'MasterSerial' = '@Serial' ON INSERT ONLY
);
```

8. The DSD_GATE.map file also defines how to map the replicated data for additional tables from the master ObjectServer to the desktop ObjectServer. It contains several commented mapping entries like the following lines:

```
# CREATE MAPPING SecurityUsersMap
# (
#   'UserID' = '@UserID' ON INSERT ONLY,
#   'UserName' = '@UserName',
#   'SystemUser' = '@SystemUser',
#   'FullName' = '@FullName',
#   'Passwd' = '@Passwd',
#   'UsePAM' = '@UsePAM',
#   'Enabled' = '@Enabled'
# );
```

Uncomment all mapping entries in the file.

9. Edit the file \$NCHOME/omnibus/gates/DSD_GATE/objserv_uni.reader.tblrep.def. This file defines the tables to replicate from the master ObjectServer to the desktop ObjectServer. It contains several commented table replication entries like the following lines:

```
# REPLICATE ALL FROM TABLE 'security.users'
# USING MAP 'SecurityUsersMap'
# INTO 'transfer.users';
```

Uncomment all table replication entries in the file.

10. Add an entry for the gateway to the Server Editor.

Your unidirectional ObjectServer Gateway is now configured and can send events from the master ObjectServer (NETCOOLPRI) to the desktop ObjectServer (DESKOS).

Step 6: Configuring process control on the desktop ObjectServer computer

Use this procedure to configure a process agent called NCOSDESK_PA on ncdesk.

Proceed as follows:

1. Log in to the computer running the desktop ObjectServer (ncdesk).
2. Edit the \$NCHOME/omnibus/etc/nco_pa.conf file.
3. Add the process, service, and routing information for the desktop ObjectServer, as follows:

```
nco_process 'Desk_ObjectServer'
{
  Command '$NCHOME/omnibus/bin/nco_objserv -name DESKOS -pa NCOSDESK_PA' run as 0
  Host='ncdesk'
  Managed=true
  RestartMsg='The desktop ObjectServer has been restarted'
  AlertMsg='The desktop ObjectServer has gone down'
  RetryCount=0
  ProcessType=PaPA_AWARE
}
nco_process 'Uni_Gate'
{
  Command '$NCHOME/omnibus/bin/nco_g_objserv_uni -name DSD_GATE run as 0
  Host='ncdesk'
  Managed=true
  RestartMsg='The unidirectional gateway has been restarted'
  AlertMsg='The unidirectional gateway has gone down'
  RetryCount=5
  ProcessType=PaPA_AWARE
}
nco_service 'Desk_OS'
{
  ServiceType=Master
  ServiceStart=Auto
  process 'Desk_ObjectServer' NONE
  process 'Uni_Gate' 'Desk_ObjectServer'
}
nco_routing
{
  host 'ncdesk' 'NCOSDESK_PA'
}
```

4. Stop the desktop ObjectServer.
5. Start the process control daemon, using the command:
\$NCHOME/omnibus/bin/nco_pad -name NCOSDESK_PA
6. To test that the desktop ObjectServer and unidirectional ObjectServer Gateway are running under process control, enter:
\$NCHOME/omnibus/bin/nco_pa_status -server NCOSDESK_PA -user netcool
7. Enter the password at the prompt.

The desktop ObjectServer and unidirectional ObjectServer Gateway should be reported as running under process control.

The configuration of the Tivoli Netcool/OMNIbus desktop ObjectServer architecture is complete.

Next steps

After you have deployed the desktop ObjectServer architecture, you can configure the Web GUI dual-server desktop architecture.

Proceed as follows:

1. On the Web GUI, server, open the `ncwDataSourceDefinitions.xml` file.
2. Define the DESKOS ObjectServer as a display server of the NETCOOLPRI ObjectServer:
 - a. Locate the `<ncwDataSourceDefinition>` element for NETCOOLPRI.
 - b. Set the type attribute of the `<ncwDataSourceDefinition>` element to `multipleServerOSDataSource`.
 - c. Add the `<ncwReadCloudDefinition>` element beneath the closing `</ncwFailOverPairDefinition>` element, in which you define the host and port of the DESKOS display server. In the `<ncwReadCloudDefinition>` element, define DESKOS in an `<ncwOSConnection>` element. For example:

```
<ncwReadCloudDefinition>
    <ncwOSConnection host="ncdesk" port="4700"/>
</ncwReadCloudDefinition>
```

3. Save and close the file.
4. Stop the server by entering the following command:
 - **UNIX** **Linux** `install_dir/bin/stopServer.sh server1 -username TIP_administrator_username -password password`
 - **Windows** `install_dir/bin/stopServer.bat`
5. Restart the server by entering the following command:
 - **UNIX** **Linux** `install_dir/bin/startServer.sh server1`
 - **Windows** `install_dir/bin/startServer.bat server1`

Chapter 28. Example installation scenario for the non-Web and Web GUI components of Tivoli Netcool/OMNIbus (Windows)

This installation scenario describes how to perform a basic installation and configuration of the non-Web components and the Web GUI component of Tivoli Netcool/OMNIbus within a Windows test environment.

Tivoli Netcool/OMNIbus tracks alert information in a high-performance, in-memory ObjectServer database, and presents information of interest to specific users through filters and views that can be configured individually. The non-Web components of Tivoli Netcool/OMNIbus include ObjectServers, probes, gateways, desktop tools, and administration tools.

The Web GUI component provides a Web-based interface for processing network events from one or more ObjectServers, and uses a client-server architecture. The Web GUI server runs inside the Tivoli Integrated Portal, which provides single sign-on, consolidated user management, and a single point of access for different Tivoli applications. Clients connect to the Tivoli Integrated Portal to access the Web GUI.

The information in this scenario acts as a quick guide to installing and running the product. The information walks you through the steps required to:

- Install and configure an ObjectServer for event management
- Install and configure a probe for sending events to the ObjectServer
- Install and configure the Web GUI for monitoring events in the ObjectServer

Setting up the test environment

In its simplest configuration, the main components that are required for setting up the test environment are the base features required for setting up an ObjectServer, a Syslogd probe for acquiring event data, and the Web GUI for monitoring the events generated.

The following figure depicts this setup. The ObjectServer (MY0BJ) and Web GUI server run on a Windows server with the host name myserverhost. The Syslogd probe runs on a Windows client with the host name probehost, and forwards events to the ObjectServer. Events that are sent to the ObjectServer can be viewed in the Web GUI Active Event List.

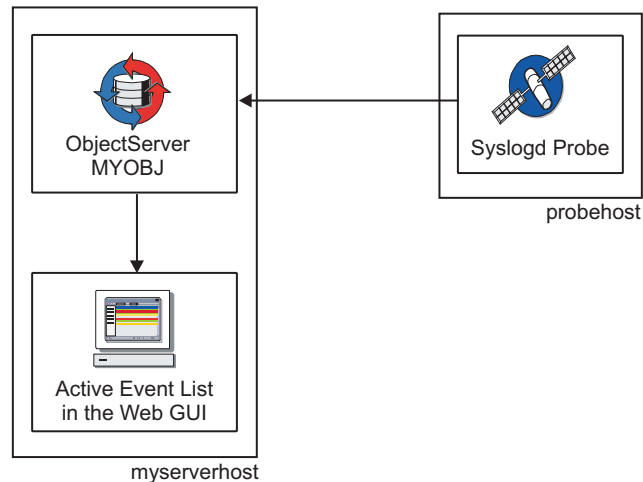


Figure 22. Architecture of test environment

The high-level steps for setting up this simple test environment are as follows:

1. Install the non-Web components of Tivoli Netcool/OMNIBus and set up an ObjectServer:
 - a. Install the base features for setting up an ObjectServer.
 - b. Create an ObjectServer.
 - c. Define and generate the interface connections between the ObjectServer and any connecting applications.
 - d. Set up the ObjectServer to run as a Windows service.
2. Install and configure the Web GUI component to receive events from the ObjectServer:
 - a. Install the Web GUI and define interface connections to the ObjectServer.
 - b. Log in to the Web GUI server.
 - c. Create a new user and configure access permissions to enable the user to view events in the Active Event List.
3. Install and configure the probe to acquire event data:
 - a. Install the **Probe Support** feature and then install the probe.
 - b. Configure interface connections to the ObjectServer and set up the probe to run as a Windows service.

Scope, assumptions, and installation prerequisites

Although Tivoli Netcool/OMNIBus is supported on various operating systems, this information describes a Windows installation and configuration for quick testing.

Assumptions are as follows:

- The following product and operating system versions are used:

Product or component (and version)	Operating system version
Tivoli Netcool/OMNIBus V7.3.1 (non-Web and Web GUI)	Windows Server 2003
Tivoli Netcool/OMNIBus Syslogd probe for Windows (latest version)	Windows XP Professional

- You are installing the non-Web and Web GUI components on the same server, and you are installing the probe on a client.
- You have downloaded the installation packages for the test system and extracted the contents of each installation package into a temporary location on the Windows server.
- The default installation directories are used.
- Your operating system has all the recommended patches, including the latest patch levels, installed.
- The Windows computers do not have any existing Tivoli Netcool/OMNIbus installation, or earlier versions of the Web GUI (that is, Netcool/Webtop).

The non-Web components, Web GUI component, and the probe are distributed as separate installation packages that you can download from the IBM Passport Advantage Web site. You can find information about the Windows download packages here:

- IBM Tivoli Netcool/OMNIbus V7.3.1 (non-Web package and Web GUI package): <http://www.ibm.com/support/docview.wss?rs=3120&uid=swg24023996>
- Tivoli Netcool/OMNIbus Probe for Syslogd: <http://www.ibm.com/support/docview.wss?rs=3120&uid=swg21410573>

Installation prerequisites are as follows:

- You must have Administrator privileges on the Windows computers.
- You must have write access permissions to the installation directories.
- For use with the Web GUI, the computer must have either version 5.0 or 6.0 Update 2 of the Sun Java Virtual Machine plug-in installed.

Alternative installation and configuration methods are available to set up and run Tivoli Netcool/OMNIbus and probes. For the purposes of this test scenario, the installation wizard will be used to install Tivoli Netcool/OMNIbus, and the various product components will be configured to run as Windows services.

Installing Tivoli Netcool/OMNIbus and setting up the ObjectServer

You must install the non-Web components of Tivoli Netcool/OMNIbus, set up an ObjectServer, and then start it before installing the Web GUI.

Installing Tivoli Netcool/OMNIbus

To install Tivoli Netcool/OMNIbus on the Windows server:

1. From Windows Explorer, navigate to the directory where you extracted the contents of the downloaded Tivoli Netcool/OMNIbus package.
2. Double-click the `install.exe` file to run the installation program.
3. From the drop-down list at the bottom of the splash screen, select a language and click **OK**.
4. From the Introduction page of the installation wizard, click **Next** to proceed to the Software Licence Agreement page.

Tip: While configuring your installation options, you can click **Previous** to revisit previous pages of the wizard and make changes if required.

5. Read the license agreement and the non-IBM terms, and then accept both the IBM and non-IBM terms. You can also click **Print** to obtain a hardcopy of the license agreement.

6. Click **Next** and wait while the wizard installs the Deployment Engine on the computer.
The installation location defaults to C:\Program Files\IBM\Common\acsi.
7. From the Select Destination Folder page, click **Next** to accept the default installation location for Tivoli Netcool/OMNIBus. This location is C:\IBM\tivoli\netcool.
8. From the Choose Install Set page, you can choose **Typical** to install all the Tivoli Netcool/OMNIBus features, or choose **Custom** to select a subset of the features. Choose **Typical** and click **Next**.

Tip: You can choose **Custom** by clicking the associated icon, and click **Next** to view the selection of features. For this test installation scenario, you would, at a minimum, require the Desktop, Servers, Confpack, Administrator, and Local Help System features.

9. From the Pre-Installation Summary page, review the installation settings and then click **Install** to start the installation. The Installing Netcool/OMNIBus page shows the progress of the installation. On completion, the Installation Complete page is displayed. This page confirms that the installation was successful and informs you that the system needs to be restarted (either now or later) to complete the installation.
10. Accept the option to restart now, and then click **Done** to close the wizard and reboot.

Results

The Tivoli Netcool/OMNIBus installation adds the following shortcuts to the Windows **Start** menu:

- **Start > All Programs > Netcool Conductor**
- **Start > All Programs > NETCOOL Suite**

What to do next

Each Tivoli Netcool/OMNIBus installation must have at least one ObjectServer to store and manage alert information. You can now create an ObjectServer by running the database initialization utility (**nco_dbinit**).

Creating an ObjectServer

To create an ObjectServer called MYOBJ for your test environment:

1. From a command prompt window, go to the C:\IBM\tivoli\netcool\omnibus\bin directory:
`cd C:\IBM\tivoli\netcool\omnibus\bin`
2. Run the following command:
`nco_dbinit -server MYOBJ`

The **nco_dbinit** utility creates the following objects for the MYOBJ ObjectServer:

- A properties file called MYOBJ.props in the location C:\IBM\tivoli\netcool\omnibus\etc
- Default database tables and data
- Default users called root and nobody, along with default groups and roles to help you manage permissions

The root user is created as an administrative user and is allocated an empty string as a password. The nobody user is disabled and cannot be used to access the ObjectServer.

Tip: Leave the command prompt window open for later use.

What to do next

You must now use the Server Editor to add communication details for the ObjectServer on the Windows server. Two Server Editor entries are required for the ObjectServer: a *listener* entry that responds to client requests and a *client* entry that local clients can use to connect to the server.

Any computer (such as the probe computer) that needs to connect to the ObjectServer will also need these communication details.

Configuring server communication information

To add the communication details for the ObjectServer on the Windows server:

1. Click **Start > All Programs > NETCOOL Suite > System Utilities > Servers Editor**.

The Server Editor window opens. The server list at the top of this window shows the list of default server entries. Any other server settings that you define in this window will be added to this list.

2. Complete this window as follows. (Instructions are provided only as relevant for the test environment.)

Listener

Initially leave this check box clear.

SSL Leave this check box clear. For the purposes of this test, encrypted connections from clients that use SSL are not being used.

Name Type the name of the ObjectServer that you created earlier - that is, MYOBJ.

Port Type a valid, unused port number in this field - for example, 4321. This is the port on which the ObjectServer will listen for connections.

Host Specify the host name of the current computer on which you installed the ObjectServer. (This name should be visible in the drop-down list.)

Add Click this button to add the MYOBJ server details to the server list above the entry fields. These details correspond to the client entry.

You must now add the listener entry:

Listener

Select this check box.

Add Click this button to add a Listeners subentry to the MYOBJ entry in the server list.

3. Click **OK** to save your changes and close the Server Editor window.

Results

The Server Editor saves the communications settings in the connections data file, which is located in C:\IBM\tivoli\netcool\ini\sql.ini.

What to do next

You must now set up the ObjectServer to run as a Windows service by running the ObjectServer executable file with one or more command-line options.

Setting up the ObjectServer as a Windows service

To set up the MYOBJ ObjectServer as a Windows service:

1. From the command prompt window, run the following command to install the ObjectServer service:

```
nco_objserv /install /cmdline "-name MYOBJ"
```

Tip: If you had already closed the previous command window from which you ran the **nco_dbinit** utility, you must change to the C:\IBM\tivoli\netcool\omnibus\bin location before running the **nco_objserv** command.

A message is displayed confirming that the ObjectServer service was successfully installed.

2. Verify that the service was created as follows:
 - a. Open the Windows Control Panel.
 - b. Double-click **Administrative Tools** and then **Services**.
 - c. From the Services window, locate the **Netcool/OMNIBus Object Server** service and double-click this entry to open the Properties window.
 - d. From the **General** tab, ensure that the value in the **Startup type** field is set to **Automatic**.
 - e. Click **OK** to close the Properties window.
3. Reboot the computer. The ObjectServer service will automatically start on system startup.

Results

The ObjectServer configuration is complete.

Installing and configuring the Web GUI

An ObjectServer must be running at the time you install the Web GUI. The installation process attempts to connect to the ObjectServer.

Before you begin

You must have the following information at hand for establishing the connection:

- The Tivoli Netcool/OMNIBus administrator name and password - in this case, root, with a blank password
- The ObjectServer name, host, and port - in this case, MYOBJ, the fully-qualified computer name, and 4321

Installing the Web GUI

On the same computer where you installed the non-Web components of Tivoli Netcool/OMNIBus:

1. From Windows Explorer, navigate to the directory where you extracted the contents of the downloaded Web GUI package.
2. Double-click the `install.exe` file to run the installation program.
3. Select the language to use for the installation procedure, then click **OK** to continue to the Introduction page.
4. Click **Next** to proceed to the Software License Agreement page.
5. Read and accept the license conditions, and click **Next** to proceed. Wait while the Deployment Engine is installed. (The Deployment Engine files that were installed with the non-Web components are detected, so the installation completes in a very short time.)
6. Accept the default installation directory shown. This path is `C:\IBM\tivoli`. Then click **Next**.
7. Click **Default** for the type of installation, and then click **Next**.
8. Complete the fields of the WebSphere Information page and click **Next**:

User ID

The default user ID for a Tivoli Integrated Portal administrative user is `tipadmin`. Leave this value unchanged.

Password

Type a password for the `tipadmin` user to be created.

Tip: You will use the user ID and password specified here to log on to the Tivoli Integrated Portal console from where you can access the Web GUI. Therefore, you need to remember these values.

Confirm Password

Retype the password for confirmation.

Port Number

The Tivoli Integrated Portal Server requires a sequence of 14 port numbers, starting with the one entered here. A default value of 16310 is supplied. If you know that any of ports 16310 through 16323 is already in use, type a different number. Otherwise, the installer will select the ports automatically.

9. From the Default User Registry Selection page, select **ObjectServer** to indicate that new users and groups will be registered on the ObjectServer.
10. Click **Next**.
11. Complete the fields of the ObjectServer Information page with details for the MYOBJ ObjectServer, and then click **Next**:

User Id

Leave the default value of `root` unchanged. This user ID for the ObjectServer administrator will be used to log on to MYOBJ.

Password

Leave this field blank. (Remember that the default `root` user for MYOBJ was created with a blank password.)

Note: In a production environment, it is important to define passwords to keep your system secure.

Confirm Password

Leave this field blank to match the previous entry.

Name Type MYOBJ as the ObjectServer name. The default value is NCOMS.

Primary Hostname

Type the fully-qualified name or the static IP address of the computer where the ObjectServer is installed. Examples of entry formats are: *MyServer.MySubdomain.MyDomain.com* and 9.51.111.121.

Primary Port

The default port number is 4100. Type the port number that you specified earlier when configuring server communication information for the ObjectServer - that is, 4321.

Enable Secondary Server for Failover

Leave this check box clear.

12. From the Pre-Installation Summary page, review the selection summary and then click **Install** to start the installation. The Installing page is displayed, along with the name of each component as it is being installed.
13. When the Install Complete page is displayed, read any messages that appear; then click **Done** to close the installation wizard.

Results

After you click **Done**, the default browser is started and the URL for the Tivoli Integrated Portal Server and its secure port number is displayed in the address box.

At this stage, you might see a security alert with a message stating that there is a problem with the security certificate. This alert indicates that the browser application is verifying the security certificate of the Tivoli Integrated Portal Server. Although this warning appears, the certificate is valid and you can accept it by clicking **Yes**.

What to do next

You can now log on to the Tivoli Integrated Portal console. (Note that the Tivoli Integrated Portal Server starts automatically after it has been installed and whenever the computer is started.)

Logging on to the Tivoli Integrated Portal console

To log on to the console from the login page:

1. Specify your user credentials as follows:

User ID

Enter tipadmin. This was the default administrator ID specified during the installation.

Password

Enter the password that you specified for the tipadmin user.

2. Click **Log in**.

Results

After your user credentials have been verified, the Welcome page for the Tivoli Netcool/OMNIBus Web GUI is displayed in the Tivoli Integrated Portal console

window. This window has a navigation pane on the left with a set of nodes for accessing the functions that you want to perform. The work area on the right typically displays the current page that you are working on, and contains one or more Web applications or portlets, each in its own portlet window with a title bar.

Note: While you are logged on to the console, avoid clicking the browser **Back** button to go to the previous Web page because you will be logged out automatically. Click **Forward** and you will see that you are logged out and must resubmit your credentials to log on again.

What to do next

After installation, the Tivoli Integrated Portal administrator must typically create one or more Web GUI administrative users with permissions to modify Web GUI settings. The Tivoli Integrated Portal administrator can also create additional users with varying access permissions. Roles and groups are associated with each Web GUI user. Roles are used to assign permissions to users, and groups can be used to logically categorize users with common functional goals.

In your test environment, you will create one user, assign roles to the user, and then log on as this user to view events generated in the Active Event List.

Creating a user and configuring access permissions for the Active Event List

To create a user and assign access permissions:

1. While logged on as the `tipadmin` user, click **Users and Groups > Manage Users** in the left navigation pane of the console window.
2. From the work area, click **Create**.
3. Enter the following details for the new user. Example entries are given here. The fields marked with an asterisk (*) on the screen are mandatory.

User ID

Type `webtestuser` as the unique identifier for the user.

First name

Type `Ann` as the first name.

Last name

Type `Other` as the last name.

E-mail Leave this optional field blank.

Password

Type `netcool` as the password.

Confirm password

Retype the password for confirmation.

4. Click **Create** to create the user and save the details in the ObjectServer.
5. When the confirmation message is displayed that the user was successfully created, click **Close**. The user details are displayed within a table in the work area. You can now assign roles to this user.
6. In the left navigation pane, the **Users and Groups** node should be in an expanded state. Click the nested **User Roles** entry.
7. From the **User Roles** work area on the right, click **Search**. A list of users appears in the grid.
8. Click the user ID for `webtestuser`. A list of the available roles appears.

9. Set the check boxes for the following roles:
 - **ncw_user**: This role grants the user access to Web GUI event management functions.
 - **ncw_admin**: This role grants the user access to Web GUI administrative functions.
 - **netcool_rw**: This role grants the user read-write access to event management functions, including access to Active Event List tools and the ability to change event data.
10. Click **Save** to assign these roles to the webtestuser user.
11. Click the **Logout** hyperlink above the work area, and adjacent to the IBM logo.
12. Log on again using the webtestuser user ID and password.

What to do next

Now that Tivoli Netcool/OMNIBus and the Web GUI are up and running, you can install the probe on a client computer and configure it to send events to the ObjectServer for viewing in the Active Event List.

The process for installing the probe is two-fold:

1. Probes require the **Probe Support** feature of Tivoli Netcool/OMNIBus to be installed, so you will need to install Tivoli Netcool/OMNIBus on the computer designated for the probe.
2. When the Tivoli Netcool/OMNIBus installation completes, you must install the probe.

Installing the probe

To install the Syslogd probe on the designated computer:

1. Copy the extracted Tivoli Netcool/OMNIBus non-Web package and probe installation package to a temporary location on this computer.
2. Install Tivoli Netcool/OMNIBus in a similar manner described in “Installing Tivoli Netcool/OMNIBus” on page 595. At a minimum, ensure that the Desktop, Confpack, Administrator, and Probe Support features are selected for installation.
3. Navigate to the extracted probe files and locate the README.txt file. Follow the instructions for installing a probe on V7.3.1, or later. On completion, the executable file, properties file, and rules file for the Syslogd probe are available in the following directory:

`C:\IBM\tivoli\netcool\omnibus\probes\win32`

What to do next

You must now configure the probe and set it to run as a Windows service. Before the probe can be run as a service, it must register itself with the Service Control Manager on Windows. A command is available for this process.

Configuring the probe and setting it to run as a Windows service

To configure the Syslogd probe and set it up as a service:

1. Click **Start > All Programs > NETCOOL Suite > System Utilities > Servers Editor**.
2. From the Server Editor window, define a listener and client entry for the MYOBJ ObjectServer as described in “Configuring server communication information” on page 597.
3. Save your entries and close the window.
4. Open the probe properties file in text mode. The file location is C:\IBM\tivoli\netcool\omnibus\probes\win32\syslogd.props.
5. Copy and paste the **Server** property to the end of the file. This enables you to keep a commented default configuration within the file as a reference.
6. Uncomment and edit the pasted **Server** property as follows:
Server : 'MYOBJ'
7. Save and close the properties file.
8. From a command prompt window, go to the C:\IBM\tivoli\netcool\omnibus\probes\win32 location. Then run:
nco_p_syslogd.exe -install
A message is displayed to confirm that the NCO Syslogd Probe service was successfully installed.
9. From the Windows Control Panel, use the Services window to start the probe service. The display name for the probe is **NCO Syslogd Probe**.

What to do next

Now that the probe is up and running, you can monitor events from the probe on the computer where Tivoli Netcool/OMNIBus and the Web GUI are installed.

Monitoring events in the Active Event List

While logged on as the webtestuser user, you can access the Active Event List from the Tivoli Integrated Portal console by clicking **Availability > Events > Active Event List (AEL)** in the navigation pane.

Full details on monitoring events in the Active Event List are provided in the Tivoli Netcool/OMNIBus documentation set.

Tip: If you want to run the Web GUI on subsequent occasions, remember that the Tivoli Integrated Portal Server must be running before you can connect to it from your browser. You must then log in to the console to start a work session. The steps are as follows:

1. The Tivoli Integrated Portal Server starts automatically after it has been installed and whenever the computer is started. This is because the server is set up as a Windows service with Automatic startup. If you, however, need to manually restart the server, you can start the service as follows:
 - a. From the Windows Control Panel, double-click **Administrative Tools** and then **Services**.
 - b. From the Services window, locate the **Tivoli Integrated Portal - TIPProfile_Port_16310** service. Then right-click the entry and click **Start**.
2. From a browser window, log on to the console by entering the following URL:
<https://localhost:16316/ibm/console>

Instead of `localhost`, you can also specify the fully-qualified host name (in the format *host_name.domain_name*) or the IP address of the Tivoli Integrated Portal Server.

Next steps

The test environment described in this scenario is intended to help you familiarize yourself with the way in which the non-Web and Web GUI components of Tivoli Netcool/OMNIbus can work together at the simplest level. The components are highly configurable; review the documentation in depth to help you determine which configuration is best suited to your requirements.

Appendix A. IBM Support Assistant

The IBM Support Assistant (ISA) is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. ISA provides quick access to support-related information along with serviceability tools for problem determination.

How can IBM Support Assistant help?

IBM Support Assistant will help you get the information you need quickly. ISA provides this quick access through its concurrent Search tool that spans across the bulk of IBM documentation and returns the results categorized by source for easy review.

ISA also provides a product information feature that has key product information links that are essential to self-help. These include:

- Product support pages
- Product home pages
- Product troubleshooting guides
- Product education roadmaps and the IBM Education Assistant
- Product recommended updates
- Product newsgroups and forums

ISA has a Tool workbench that provides you with the problem determination tools that IBM Software Support uses to resolve issues.

Included in ISA, is a Service feature with an automated system and symptom based collector. The system collector gathers general information from your operating system, registry, and other relevant applications. The system-based collection provides the unique ability to collect specific information relating to a particular problem that you are having.

You can also use ISA to enter your entitlement information once and have it saved for future sessions. This enables you to create a problem report for IBM and attach the collector file at the same time.

Supported version for Tivoli Netcool/OMNIBus

A Tivoli Netcool/OMNIBus product plug-in is available for you to install within the ISA framework to gather information that can be used to diagnose and resolve problems.

The minimum requirement for Tivoli Netcool/OMNIBus is IBM Support Assistant V4.1.1.

To use the IBM Support Assistant with Tivoli Netcool/OMNIBus, you must install both the IBM Support Assistant and the Tivoli Netcool/OMNIBus product plug-in after you install Tivoli Netcool/OMNIBus. These two components are not provided on the Tivoli Netcool/OMNIBus installation media, but you can download them.

Downloading IBM Support Assistant

If you do not have IBM Support Assistant installed, you must download the compressed archive file for the IBM Support Assistant Workbench, and extract the files. The download location of ISA is: <https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=isa>. You can download ISA for any operating system that is supported by Tivoli Netcool/OMNIbus, providing an ISA archive file exists for that operating system. You need to log in using your IBM Web identity; if you do not already have one, complete the free registration process to obtain a web identity. The archive contains an installation program that you must use to install ISA, and the Installation and Troubleshooting guide, which provides the relevant instructions.

If you already have an earlier version of ISA installed, or if the downloaded version for your operating system was earlier than V4.1.1, you must update to V4.1.1. Start the application and use its Updater component to locate the **IBM Support Assistant** and **IBM Support Assistant Language Pack** upgrades to V4.1.1. Select and install these features.

Installing the product plug-in for Tivoli Netcool/OMNIbus

After you install or upgrade ISA, you must use its Updater component to locate and install the product plug-in for the current version of Tivoli Netcool/OMNIbus. Select **Update > Find New > Product Add-ons**. The Tivoli Netcool/OMNIbus plug-in is available as a new plug-in under the Tivoli brand.

Using the ISA data collection tool with Tivoli Netcool/OMNIbus

The ISA data collecting tool can collect diagnostic information about your Tivoli Netcool/OMNIbus issues.

Restriction: This information is applicable to the non-Web components of Tivoli Netcool/OMNIbus only.

Requisites for running this tool are as follows:

- You must have read, write, and execute permissions to all the plug-in subdirectories within the ISA installation directory.
- On Windows, you must have Administrator privileges.
- You must have the permission to run all the applications in the Tivoli Netcool/OMNIbus installation.
- There must be an ObjectServer running in production mode, from which the data can be retrieved. Otherwise, some steps are skipped; for example, information about your locale and your Sybase version cannot be collected.
- You must know the ObjectServer credentials so that you can log in to the ObjectServer as a Tivoli Netcool/OMNIbus Administrator.
- You can send the data collection results to <ftp://ftp.emea.ibm.com/> automatically. (Alternatively, you can send the results to another IBM server. You must have the FTP server name, user name, password, and remote directory. Contact your Level 2 support for this information.)

If necessary, you might need to log in Electronic Service Request (ESR) to submit a problem report.

To collect data about a Tivoli Netcool/OMNIbus component, perform the following steps on the local computer where the component is installed:

1. Start the ISA.
2. Run the System Collector to collect the system information about the computer.
3. Run the ISA data collecting tool.

For example if you are encountering issues with the process agent (**nco_pad**) on a Solaris computer, start the ISA on that computer, run the System Collector, and then run the data collecting tool for the General Problem Type. Similarly, if you encounter issues with an event list on a Windows client, which is connected to an ObjectServer on a Solaris computer, start the ISA on the Windows computer and run the System Collector. Then run the data collecting tool for the Desktop Problem Type.

Training material for IBM Support Assistant

The following training material is available:

- IBM Support Assistant comes with a built-in user guide.
- The installation image that you download includes an HTML Installation and Troubleshooting Guide.
- IBM Education Assistant available at <http://www-306.ibm.com/software/info/education/assistant/> provides training modules that have been created to show how to install and use IBM Support Assistant.
- IBM Support Assistant tutorial for version 4 is available directly at http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.isa/isa/ISAv40_Task.html.

Related concepts

“Integration with other Tivoli products” on page 31

Tivoli Netcool/OMNIBus and the Log and Trace Analyzer

With the Log and Trace Analyzer, you can gather system and performance data from local and remote systems. The data can be used for problem determination should a less than optimal system event occur.

You can use the Log and Trace Analyzer to create resource sets. Resource sets are sets of definitions that contain the path locations of the logs that you need to examine and the levels of information that they contain. You can keep customized definitions to reuse. The definitions provide the same set of instructions about where to find a log, and what kind of information to gather from the log, saving time during subsequent log imports.

The Log and Trace Analyzer also makes it possible for you to download and store symptom database catalogs to your local system. These catalogs provide detailed diagnostic solutions to a variety of scenarios, which can give direction to your troubleshooting tasks.

To use the Log and Trace Analyzer, you must have downloaded and installed the ISA and the product plug-in for Tivoli Netcool/OMNIBus.

Downloading the Log and Trace Analyzer

To download the Log and Trace Analyzer, complete the following steps:

1. Using the ISA built-in Updater component, download and install the plug-in for Log and Trace Analyzer from the IBM Web site at <http://www.ibm.com/software/support/isa/>.
 - a. Select the Log Analyzer tool add-on from the list of **JVM-based Tools** and click **Next**.
 - b. Review and indicate that you accept the associated license agreements and click **Next**.
 - c. Review the list of add-ons to be downloaded and installed and click **Finish**.
2. After installation of the Log and Trace Analyzer is complete:
 - a. Start the ISA.
 - b. Select **Analyze Problem**.
 - c. Click the **Tools** tab.
 - d. Select the **Log Analyzer** from the list of tools in the **Tools Catalog**.
 - e. Click **Launch**. The Log Analyzer starts.

Importing Tivoli Netcool/OMNIBus log files to the Log and Trace Analyzer

To import the Tivoli Netcool/OMNIBus log files to the Log and Trace Analyzer, complete the following steps:

1. Copy the relevant log files from the Tivoli Netcool/OMNIBus servers to the system where you installed the IBM Support Assistant workbench. Put the log files for each server in a unique directory. For example, C:\OMNI\logs\NCOMS1.
2. Import the Tivoli Netcool/OMNIBus log files. The Log and Trace Analyzer organizes related log files into log sets. Log sets can be used to import and analyze a set of related log files. This facility is used to organize and import your Tivoli Netcool/OMNIBus log files. Log set definitions provide information to the Log and Trace Analyzer specifying where log and trace data are located, and what kind of data to gather from local and remote systems. The Log and Trace Analyzer allows you to import predefined log sets that contain the necessary path information required for retrieving log files on demand.
3. Use one of the following procedures:

Procedure	Steps
Create the initial Tivoli Netcool/OMNIbus log set	<ol style="list-style-type: none"> 1. Click File > Import Log File. 2. Create a new log set. 3. Type the name for the log set; for example: Tivoli Netcool/OMNIbus Log files for NCOMS1 4. Click Add. 5. Complete the following steps: <ol style="list-style-type: none"> a. In the Name Filter window, to limit the list of log files to the Tivoli Netcool/OMNIbus log files, type Discovery. b. Select the type of log file that you are adding to the log set. c. Type the name of the log file on your local system. Ensure that the type of log file matches the log file you specified. d. Enter the correct version of the Tivoli Netcool/OMNIbus product that corresponds to the log file. See the Log and Trace Analyzer online help for additional options. e. To add the log file to the log set, click OK. <p>For every log file that you want to include in the log set, repeat step 5.</p> <p>Best practice: The first time that you create the log set, you can save time later by including every log file that you want to include in the log set.</p>
Reuse an existing Tivoli Netcool/OMNIbus log set	<ol style="list-style-type: none"> 1. Click File > Import Log File. 2. Select an existing Log Set Definition from the drop-down list of defined log sets. 3. If necessary, change the contents of the log set definition. You can add, edit, or remove from the list of log files in the log set.

4. To indicate that the file should be imported to the log set, select the check box that is adjacent to the log file.
5. To import the log files, click **Finish**.

To reuse an existing Tivoli Netcool/OMNIbus log set, complete the following steps:

1. To indicate that the file should be imported to the log set, select the check box that is adjacent to the log file.
2. To import the log files, click **Finish**.

You can create and reuse as many log sets as you need. For example, when importing log files from multiple servers, you need more than one log set.

Analyzing Tivoli Netcool/OMNIbus log files with the Log and Trace Analyzer

Using the Log and Trace Analyzer, you can correlate multiple Tivoli Netcool/OMNIbus log files into a single view. The Tivoli Netcool/OMNIbus log

files can be combined in a single view, ordered by time stamp, to correlate the operation of the Tivoli Netcool/OMNIBus components. There are two ways to correlate log files:

1. Simple: To correlate all imported log file, complete the following steps:
 - a. In the Log and Trace Analyzer navigation tree view, right-click **Logs**.
 - b. Click **View All Logs**.
2. Advanced: To correlate a set of log files by creating a custom correlation, complete the following steps:
 - a. In the Log and Trace Analyzer navigation tree view, right-click **Correlations**.
 - b. Click **New > Log Correlation**.
 - c. In the window that is displayed, type the name for the correlation.
 - d. Add the log files that you want to include for the correlation.
 - e. Click **Finish**.
 - f. Refresh the navigation tree view.
 - g. In the navigation tree view, right-click the correlation name you typed and click **Open With > Log View**.

After you create a view of the logs, you can organize the log data to isolate problems. The following list identifies some of the ways that you can organize the data:

- Sort log records: For example, you can sort by time, component, and server name.
- Highlight log records: For example, you can highlight all error events in red, or show all events from a specific component in blue. Highlighting is similar to filtering, but instead of eliminating data from a view, you can highlight the relevant information within the full list of events.
- Filtering log records: You can narrow the scope of a problem and the data shown based on filter criteria. Examples of filter criteria include time stamps, severity, component, and server.
- Finding log records: You can search for specific information in a log file. For example, you can search to see events that are related to interaction with a specific server or user.

For more information about how to organize the data, in the Log and Trace Analyzer online help, search for the "Analyzing log files" topic. "Filtering, Sorting, Finding, and Highlighting" is a subheading in this topic.

In addition, there are some other topics in the online help that you might find useful:

- When trying to correlate log files from multiple servers, the time clocks on those servers can be out of sync. This synchronization problem could be something simple, like different time zones, or more subtle, such as a clock being a few milliseconds off from the clock of another server. The Log and Trace Analyzer imbeds a function to synchronize the time between multiple log files by allowing you to adjust the time stamps in a log file. For more information, see the topic titled "Synchronizing time of log records for distributed applications" in the Log and Trace Analyzer online help.
- You can use symptom catalogs to quickly recognize known problems. The Log and Trace Analyzer provides a log analysis capability that allows it to recognize known problems that are defined in a knowledge database, called the *symptom catalog*. IBM provides a symptom catalog for known problems with several products. It also provides a way for you to capture and define your own

symptom information. For more information, see the topic titled "Synchronizing time of log records for distributed applications" in the Log and Trace Analyzer online help.

Appendix B. Troubleshooting

Use this information to troubleshoot Tivoli Netcool/OMNIBus.

Troubleshooting installation

Use this information to troubleshoot installation issues.

Absence of Start menu shortcut icons on 64-bit Windows Server 2003 and Windows Server 2008

When the installer runs in silent mode on Windows Server 2003 and Windows Server 2008 64-bit operating systems only, the Tivoli Netcool/OMNIBus shortcuts are added to the **Start** menu without icons. This does not affect the operation of the shortcuts.

The shortcut icons are otherwise always added to the Windows **Start** menu.

Viewing the installation and migration log files, and installed packages

After installing or upgrading Tivoli Netcool/OMNIBus, you can view the installation and migration log files to verify that you installed Tivoli Netcool/OMNIBus successfully, or for troubleshooting purposes.

Information about the list of installed packages and their versions can also be useful when troubleshooting. You might be asked for package information by IBM Software Support.

Related tasks

“Viewing the installation log files (UNIX and Linux)” on page 60

“Viewing the migration log file (UNIX and Linux)” on page 80

“Viewing the installation log files (Windows)” on page 124

“Viewing the migration log file (Windows)” on page 140

“Viewing installed packages (UNIX and Linux)” on page 61

“Viewing installed packages (Windows)” on page 124

Installation error messages

Error messages provide information about problems that occur when installing Tivoli Netcool/OMNIBus. You can use the information that they contain to resolve such problems.

The following table describes the error messages that may occur when installing Tivoli Netcool/OMNIBus and how to resolve the associated problem.

Table 107. Common installation error messages

Error	Description	Action
Wrong JRE Detected	<p>The Java Runtime Enviromnet (JRE) installed with the Deployment Engine is not the recommended IBM JRE. Additional products which use the DE are packaged with different JREs. The Tivoli Netcool/OMNIbus installer includes an IBM JRE, therefore this error will only occur when using a Tivoli Netcool/OMNIbus uninstaller when another JRE is present.</p>	<p>Terminate the installer.</p> <p>The required JRE is located in the following Tivoli Netcool/OMNIbus installation directory: \$NCHOME/platform/ARCH/jre_1.6.7/jre.</p> <p>If this error occurs when you are attempting to run the ./uninstall on UNIX, use the following command-line script:</p> <pre>cp -R \$NCHOME/platform/ <ARCH>/jre_1.6.7/jre /tmp/jre ./uninstall LAX_VM /tmp/jre/bin/java rm -rf /tmp/jre</pre> <p>Where <ARCH> is your computer's platform identifier.</p> <p>If the error occurs when you are uninstalling Tivoli Netcool/OMNIbus on Windows, use the following commands:</p> <pre>xcopy /e/i %NCHOME%\platform\win32\ jre_1.6.7\jre c:\tmpjre %NCHOME%_uninst\ <OMNIbus>\uninstall.exe LAX_VM C:\tmpjre\bin\ java.exe rmdir /s/q "c:\tmpjre</pre> <p>Note: When attempting to uninstall another product or fix pack, replace <OMNIbus> with the relevant product name.</p>

Table 107. Common installation error messages (continued)

Error	Description	Action
Wrong operating system	The \$prop.os.name\$ operating system on \$prop.os.arch\$ is not recognized.	<p>Refer to the Tivoli Netcool/OMNIBus Installation Guide and confirm that the operating system is supported on your computer.</p> <p>Note: You should terminate this installation and resolve the problem before attempting to install Tivoli Netcool/OMNIBus.</p>

Table 107. Common installation error messages (continued)

Error	Description	Action
Wrong deployment engine	<p>You are updating an instance of Tivoli Netcool/OMNIBus that was installed using the Tivoli Deployment Engine (DE) located at \$COI_PRECHECK_DEHOME\$. However, you are using the DE located at \$IAGLOBAL_ACU_INSTALL_LOCATION\$.</p>	<p>Check the IA-Netcool-OMNIBus log file for the following message:</p> <p>Current DE {0} not original DE {1}</p> <p>Confirm which version of the Deployment Engine is currently being used, and which version of the DE was previously being used. If the original version of the DE was a local installation, confirm that you are logged on as the correct user, and confirm that the version of the DE is complete.</p> <p>Use the de_lsrootiu command and confirm that Tivoli Netcool/OMNIBus is registered in the DE that is going to be used (refer to the first entry in the error message). If it is registered, the error message can be ignored.</p> <p>If it is not registered, confirm whether it is registered in the original DE (refer to the second entry in the error message). If it is registered, confirm that this DE is the one that will be used.</p> <p>Additionally, confirm that you have full read/write/execute permissions on the associated DE and, if necessary, remove the read/write/execute permissions from the wrong DE.</p> <p>Note: You should terminate this installation and resolve the problem before attempting to install Tivoli Netcool/OMNIBus.</p>

Table 107. Common installation error messages (continued)

Error	Description	Action
Creating global installation of DE	You are running the installer as the super user or computer administrator. If you continue a new global installation of the Deployment Engine will be installed. If a normal user then installs another product using the DE, they will need full read-write access to your installation of the DE.	Consider whether you want other users to use the global installation of the DE. If you continue with the installation, once the DE has been installed, you have the option to change the user access policy of the installation. You can restrict access to the DE to the super user or administrator only, a given operating system group, or allow all users to access the DE installation. If you require each user to use their own installation of the DE: terminate the installer and then run the installer as a normal user.
Using global installation of DE	You are not running the installer as the super user or computer administrator. A global installation of the DE is installed. If you continue, you will need full read-write access to this global installation of the DE.	If you can confirm that you have read-write access to the global DE you can continue with the installation. Otherwise you should check with the super user or computer administrator. Alternatively, you can change the user access policy by running the <code>de_security</code> command in the DE bin directory, or ask the super user or computer administrator to install this product.

Table 107. Common installation error messages (continued)

Error	Description	Action
Wrong Netool/OMNIBus directory	You are updating an instance of Tivoli Netcool/OMNIBus located at \$USER_INSTALL_DIR\$ but this installation appears to have been originally located at \$COI_PRECHECK_NCHOME\$.	<p>Check the IA-Netcool-OMNIBus log file for the following message:</p> <p>Current NCHOME {0} not original NCHOME {1}.</p> <p>If your installation of Tivoli Netcool/OMNIBus has been moved to another location, you must move it back to the original location.</p> <p>If both directory paths are valid and point to the same installation, run the installer once again using the second directory path displayed in the error message.</p> <p>Use the <code>de_lsrootiu</code> command to retrieve the root IU information of deployed applications from the Deployment Engine installation database.</p> <p>Note: You should terminate this installation and resolve the problem before attempting to install Tivoli Netcool/OMNIBus.</p>
Wrong user	You are updating an instance of Tivoli Netcool/OMNIBus that was installed by \$COI_PRECHECK_USER\$. However, your user name is \$prop.user.name\$.	<p>The specified user must perform the operation.</p> <p>This can be ignored if the ownership of the Deployment Engine and Tivoli Netcool/OMNIBus has been changed, and you have the correct version of the Deployment Engine.</p> <p>Note: You should terminate this installation and resolve the problem before attempting to install Tivoli Netcool/OMNIBus.</p>

Troubleshooting the Deployment Engine

Use this information to troubleshoot Deployment Engine issues.

UnknownHostException

When you run the Deployment Engine installation an `UnknownHostException` is generated.

You must ensure that your computer hostname is configured correctly before you run the Deployment Engine installation on a Windows or Unix platform.

Configuring the hostname on a Windows platform

You must ensure that your computer hostname is configured correctly before you run the Deployment Engine installation on a Windows platform.

To configure the hostname on Windows platforms:

1. Right-click on **My Computer**.
2. Click on **Properties > Computer Name** tab, and then click **Change** to display the **Computer Name Changes** dialog,
3. In the **Computer Name** field, enter the new hostname of the Domain Controller, and then click **OK**.
4. And finally, restart your computer.

Configuring the hostname on a Unix platform

You must ensure that your computer hostname is configured correctly before you run the Deployment Engine installation on a Unix platform.

To ensure that your computer hostname is configured correctly on a Unix platform, the following IP address/hostname mapping should appear in the `/etc/hosts` file:

Table 108. Example /etc/hosts file

IP address	hostname	short hostname
127.0.0.1	localhost.localdomain	localhost
<ip_address>	<hostname>	<short hostname>

Where:

<ip_address> is the IP address of your computer, <hostname> is your computer's hostname, and <short hostname> is an alias for your computer.

Backing up and restoring the Deployment Engine

Before installing additional components or other products that are based on the Deployment Engine (DE), reinstalling or upgrading Tivoli Netcool/OMNIBus, or applying a fix pack, back up the DE database. To recover the original configuration after a failure, run the DE restore script.

Perform this task:

- If you are installing Tivoli Netcool/OMNIBus or the Web GUI component on a server on which the DE is currently installed, for example, a server that currently hosts another DE-based product.
- If you are installing another DE-based product or component on a server that currently hosts Tivoli Netcool/OMNIBus or the Web GUI.

- If you are upgrading Tivoli Netcool/OMNIbus or the Web GUI.
- If you are applying a fix pack
- During routine system administration

You do not need to perform this task if you are installing the product on a clean server.

To back up and restore the DE:

- Back up the DE as follows:
 1. Change to the acsi directory:
 - **UNIX** **Linux** The path to the directory varies depending the installation type:
 - Root: The path is as follows:
`/var/ibm/common/ascii`
 - Non-root: The path is relative to your home directory, for example, `nonrootuserhomedirectory>/.ascii_username`
 - **Windows** `C:\Program Files\IBM\Common\ascii`
 2. Change to the bin directory:
 - **UNIX** **Linux** The path to the directory varies depending on the installation type:
 - Root: The path is as follows:
`/var/ibm/common/ascii/bin`
 - Non-root: The path is relative to your home directory, for example, `nonrootuserhomedirectory/.ascii_username/bin`
 - **Windows** `C:\Program Files\IBM\Common\ascii\bin`
 3. Run the DE backup script as follows:
 - **UNIX** **Linux** `de_backupdb`
 - **Windows** `de_backupdb.cmd`
- To restore the DE database, initialize the DE, change to the bin and run the restore script:
 - **Linux** **UNIX** `de_restoredb`
 - **Windows** `de_restoredb.cmd`

What to do next

If you backed up the DE database, you can install Tivoli Netcool/OMNIbus, install additional components or products on the server, upgrade Tivoli Netcool/OMNIbus, or apply the fix pack. If you restored the DE database, you can resume using the original installed environment.

Uninstalling the Deployment Engine

If the installation or uninstallation of Tivoli Netcool/OMNIBus failed, you might have to remove the Deployment Engine (DE). Under normal circumstances, it is not required to remove the DE, so perform this action only if you have been advised to do so by IBM Software Support.

Before you begin

Make sure you have already attempted to use the Tivoli Netcool/OMNIBus uninstaller to uninstall the DE. If Tivoli Netcool/OMNIBus is the only installable unit that is installed on the DE, the Tivoli Netcool/OMNIBus uninstaller removes the DE.

You must have a working knowledge of the DE and be familiar with the installation directory structure and the common directory structure, and the different user modes.

Related concepts

“The Deployment Engine” on page 34

“Directory structure for the Deployment Engine” on page 35

Related tasks

“Backing up and restoring the Deployment Engine” on page 619

Related reference

“Deployment Engine user modes” on page 36

Appendix C, “Deployment Engine command reference,” on page 627

Uninstalling a root instance of the Deployment Engine (UNIX and Linux)

Follow these steps to remove a root instance of the Deployment Engine (DE) from a UNIX or Linux system.

Before you begin

To uninstall the DE:

- Run one of the following commands:

- `si_inst.sh -r`
- `si_inst.sh -r -f`

Use the `-r` command-line option if no Installable Units (IUs), for example Tivoli Netcool/OMNIBus, are installed into the IU registry of the DE. Use the `-r` and `-f` command-line options to force the removal of the DE if installable units, for example, Tivoli Netcool/OMNIBus remain installed into the IU registry of the DE. If IUs are installed into the IU registry, the `si_inst` command with the `-r` command option does not remove the DE.

- If you need to remove files left behind from a failed attempt to install the DE, or a failed attempt to uninstall the DE, proceed as follows:
 1. Remove the following directories:
 - `/var/ibm/common`
 - `/usr/ibm/common` (This is the default location.)
 2. Clean the `/tmp` directory, by removing the `acu_de.log` file, if it exists.
 3. Remove the `/tmp/username` directory, where *username* is the ID of the user that installed DE, for example root.

4. Remove all DE references from the `/etc/inittab` system file. The DE entries are delimited by `#Begin AC Solution Install` block and `#End AC Solution Install` block. Remove all of the text between the delimiters, and the delimiting text.

Uninstalling a non-root instance of the Deployment Engine (UNIX and Linux)

Follow these steps to remove a non-root instance of the Deployment Engine (DE) from a UNIX or Linux system.

Before you begin

To uninstall the DE:

- Run one of the following commands:

```
- si_inst.sh -r
- si_inst.sh -r -f
```

Use the `-r` command-line option if no Installable Units (IUs), for example Tivoli Netcool/OMNIbus, are installed into the IU registry of the DE. Use the `-r` and `-f` command-line options to force the removal of the DE if installable units, for example, Tivoli Netcool/OMNIbus remain installed into the IU registry of the DE. If IUs are installed into the IU registry, the **si_inst** command with the `-r` command option does not remove the DE.

- If you need to remove files left behind from a failed attempt to install the DE, or a failed attempt to uninstall the DE, remove the following directories:

```
- /home/username/.acsi_hostname
- /tmp/acsitempLogs_username
- /tmp/acsitemp_username
```

In the above directory paths, *username* is the identified non-root user that installed the DE and *hostname* the name of the server on which the DE was installed.

Uninstalling the Deployment Engine (Windows)

Follow these steps to remove an instance of the Deployment Engine (DE) from a Windows system. On Windows, only the Admin user can install the DE. Installation by a non-Admin user is not supported.

Before you begin

To uninstall the DE:

- Run one of the following commands:

```
- si_inst.bat -r: Use the -r command-line option if no Installable Units (IUs),
  for example Tivoli Netcool/OMNIbus, are installed into the IU registry of the
  DE.
- si_inst.bat -r -f: Use the -r and -f command-line options to force the
  removal of the DE if installable units, for example, Tivoli Netcool/OMNIbus
  remain installed into the IU registry of the DE. If IUs are installed into the IU
  registry, the si_inst command with the -r command option does not remove
  the DE.
```

- If you need to remove files left behind from a failed attempt to install the DE, or a failed attempt to uninstall the DE, proceed as follows:

1. From Windows Task Manager, search for a process called **jservice.exe**. If the process is listed, end it.

2. Remove the C:\Program Files\IBM\Common directory.
3. Remove the acu_de.log file from the %TEMP% directory.
4. Remove the %TEMP%\username directory, where *username* is the ID of the user that installed the DE.
5. Check to see if the following services are running. If you identify these services then stop them.
 - IBM ADE Service
 - ASCI Service: If you identify this service then complete step 6
6. Optional: If you identified the ASCI Service, you must remove it from the registry.
Run the **regedit** utility. Then, from the Registry Editor , navigate to the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services directory, delete the acsisrv entry, and reboot.

Troubleshooting security

Use this information to troubleshoot security issues.

root access requirements for Tivoli Netcool/OMNIBus processes

Tivoli Netcool/OMNIBus does not require root access to operate. Exceptions apply to process control and PAM usage.

Root access is required when the process agent is configured to execute processes as a different user from the one who started the process agent.

Root access is required when PAM is being used and is configured such that it accesses objects that are owned by root.

Note also that the SNMP Probe can be run as `suid root` without compromising system security, when root access to ports is required. In this mode, the probe drops its root privileges after it has opened the SNMP session, and before the IBM Tivoli Netcool/OMNIBus probe library starts. For further information about running the probe as `suid root`, see the publication for the SNMP Probe. You can access this publication as follows from the IBM Tivoli Network Management Information Center (<http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp>):

1. Expand the *IBM Tivoli Netcool/OMNIBus* node in the navigation pane on the left.
2. Expand the *Tivoli Netcool/OMNIBus probes and TSMs* node.
3. Go to the *Universal* node.

Related reference

“User authentication failure with Pluggable Authentication Modules (PAM)” on page 624

nco_pad fails when using PAM authentication on SUSE Linux

The process control agent daemon (nco_pad) fails when using PAM authentication on SUSE Linux.

When you run the process control agent daemon (nco_pad) with PAM authentication on SUSE Linux, the default nco_pad stack size must be increased. To increase the nco_pad stack size to accommodate PAM authentication, run the `$NCHOME/omnibus/bin/nco_pad` command, specifying one of the following command-line options:

- `-stacksize 139248` (for SUSE Linux version 9.0)
- `-stacksize 278496` (for SUSE Linux version 10.0).

User authentication failure with Pluggable Authentication Modules (PAM)

Authentication to an external PAM authentication system can fail if the ObjectServer, process agent, or gateway process is not running as root.

This is not a limitation of the Tivoli Netcool/OMNIbus processes, but is instead caused by the underlying PAM and operating system configuration. For example, this issue typically occurs if your system is configured to use the `pam_unix` (or equivalent) module, and the operating system is configured (using the `/etc/nsswitch.conf` file or similar) to check the local shadow password file, rather than NIS or LDAP. The Tivoli Netcool/OMNIbus processes require read access to all the files that the PAM module will access, including the `/etc/shadow` file (or `/etc/security/passwd` on AIX), which stores secure user account information. However, operating system permissions are generally set so that the `/etc/shadow` file can be read only by the root user:

```
r----- root /etc/shadow
```

A non-root process therefore cannot read the `/etc/shadow` file in order to validate user passwords, and this results in an authentication error.

There are several workarounds to resolve this issue:

- Ask your system administrator to reconfigure the operating system so that the files are not checked.
- Change the PAM configuration to use a different module that does not require access to protected resources (for example, `pam_krb5`).
- Change the permissions on the protected resources. For example, grant read access to the `/etc/shadow` file for the user that the ObjectServer, process agent, or gateway is running as. (While this workaround might be deemed unsuitable in production environments, it could be temporarily applied in a test environment to investigate whether the authentication failure is linked to the operating system and PAM configuration.)
- Specify an Access Control List (ACL) for the `/etc/shadow`.

Example: On Solaris operating systems, you can use the `setfacl` command to create an ACL (beforehand, check with your administrator to ensure that this command is available on your system). The following example shows how to create such an ACL for the user netcool:

```
vi /tmp/shadow.acl
user::r--
user:netcool:r--
group:----
```

```

mask:r--
other:---

setfacl -f /tmp/shadow.acl /etc/shadow

getfacl /etc/shadow

# file: /etc/shadow
# owner: root
# group: sys
user::r--
user:netcool:r--      #effective:r--
group:----             #effective:----
mask:r--
other:---

```

- Run the ObjectServer, process agent, and gateway processes as root. That way, these processes can read the /etc/shadow file, and passwords entered in Tivoli Netcool/OMNIbus can be validated against the encrypted passwords in the shadow file.

Note: The pam_aix module requires the calling process (for example, the ObjectServer, process agent, or gateway) to be run as root. If the process is running as a non-root user, granting read access to protected operating system files is not a viable workaround, and will still result in authentication failures.

Logging into the Web GUI after the LDAP server has failed

If the Web GUI is configured to authenticate against an LDAP server, no user can log into the Web GUI if the LDAP server fails.

This problem also affects the default tipadmin user. To enable access to the Web GUI installation for the tipadmin user if the LDAP server fails:

1. Change to the /opt/IBM/tip_v2/profiles/TIPProfile/bin directory and start the **wsadmin** utility.
2. Use the **updateIdMgrRealm** command to change the **allowOperationIfReposDown** parameter from false to true, on the defaultWIMFileBasedRealm realm:
`$AdminTask updateIdMgrRealm {-name defaultWIMFileBasedRealm -allowOperationIfReposDown true}`
3. Restart the Tivoli Integrated Portal server.

You can now log into the Web GUI by using the tipadmin user and password.

For more information about the **wsadmin** and its associated commands, see the *Websphere Application Server* information center at <http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.wim.doc/UnableToAuthenticateWhenRepositoryIsDown.html>

Related tasks

“Restarting the server” on page 568

Troubleshooting multicultural support

When running the ObjectServer in UTF-8 encoding on Windows, the desktop event list on Windows might fail to correctly display some characters in the ObjectServer due to limitations of the event list.

The desktop event list does not fully support UTF-8 encoding, so you must instead use the Active Event List in the Web GUI component to view event data in UTF-8 encoding.

Related concepts

“Setting your locale” on page 403

Appendix C. Deployment Engine command reference

A number of administration commands are available for the Deployment Engine (DE)..

The following table provides a description for each of the administration commands that are available for the DE.

Table 109. Deployment Engine commands

Command	Description
<div>UNIX</div> de_backupdb	Use this command to perform an immediate back up of the current Deployment Engine installation database. By default, a Backups directory is created in same location as your DE installation.
<div>Windows</div> de_backupdb.cmd	<p>Note: The command is used to back up the DE installation database only, and not the global DE installation.</p> <p>To create the backup in a different location, run the following command with the -bfile option:</p> <p>de_backupdb -bfile <i>location</i></p> <p>where <i>location</i> is where you want to create the backup DE installation.</p>
<div>UNIX</div> de_lsrootiu.sh	Use this command to retrieve the root package information of deployed applications from the Deployment Engine installation database.
<div>Windows</div> de_lsrootiu.cmd	
<div>UNIX</div> de_restoredb	Use this command to restore a Deployment Engine installation database from an existing copy of that database.
<div>Windows</div> de_restoredb.cmd	<p>Note: The command is used to restore the DE installation database only, and not the global DE installation.</p> <p>Note: This command deletes your current DE installation before restoring the database. If you do not specify any options for the de_restoredb command, the restore is performed using the most recent backup of the installation database which is located in the DE_installation_dir/backups directory.</p> <p>For example, the following command restores the installation database from the backup file, backup.db, in your top-level C:\ directory:</p> <p>de_restoredb -bfile c:\backup.db</p>

Table 109. Deployment Engine commands (continued)

Command	Description
<div>UNIX</div> <div>de_security.sh</div> <div>Windows</div> <div>de_security.cmd</div>	<p>Use this command to change the filing system protection settings on a multi-user mode Deployment Engine. This command should be used by a root user on UNIX or a member of the Administrator group on Windows. One of the following command line options can be selected:</p> <ul style="list-style-type: none"> -singleUser: only the user that installed the Deployment Engine can modify it or use it to install or remove products. -group <i>groupname</i>: only the user that installed the Deployment Engine and users in the specified group can modify it or use it to install or remove products. -global: any user can modify the Deployment Engine or use it to install or remove products. <p>Note: If an option is not selected the current setting is displayed.</p>
<div>UNIX</div> <div>de_version</div> <div>Windows</div> <div>de_version.cmd</div>	<p>Use this command to display the version of the Deployment Engine runtime environment currently installed on your system.</p>
<div>UNIX</div> <div>listIU.sh</div> <div>Windows</div> <div>listIU.bat</div>	<p>Use this command to query the Deployment Engine databases for all existing packages.</p>
<div>UNIX</div> <div>si_inst.sh</div> <div>Windows</div> <div>si_inst.bat</div>	<p>Use this command to install the Deployment Engine runtime environment on your system.</p> <p>To install the Deployment Engine in a custom location, run the installation script with the -i option. For example:</p> <p><code>si_inst.bat/sh -i <i>custom_install_path</i></code></p> <p>where <i>custom_install_path</i> is your specified location.</p>
<div>UNIX</div> <div>si_inst.sh -r [-f]</div> <div>Windows</div> <div>si_inst.bat -r [-f]</div>	<p>Use this command to remove the Deployment Engine runtime environment from your system.</p> <p>Use the -r command-line option if no Installable Units (IUs), for example Tivoli Netcool/OMNIBus, are installed into the IU registry of the DE. If IUs are installed into the IU registry, the si_inst command with the -r command option does not remove the DE. Use the -r and -f command-line options to force the removal of the DE if installable units, for example, Tivoli Netcool/OMNIBus remain installed into the IU registry of the DE.</p>
<div>UNIX</div> <div>de_help</div> <div>Windows</div> <div>de_help.cmd</div>	<p>Use this command to display a list of all the available administration commands. Alternatively, you can display help about a specific command by entering the command name followed by the -help option.</p>

Related concepts

“The Deployment Engine” on page 34

Appendix D. Default port numbers used by Tivoli Netcool/OMNIBus

A number of default port numbers are defined for Tivoli Netcool/OMNIBus. You can change these default values.

The following table lists the default ports and specifies how to change these port numbers.

Table 110. Default ports

Component and default port	Port configuration
Tivoli Netcool/OMNIBus servers: <ul style="list-style-type: none">• ObjectServer (NCOMS): 4100• Process agent (NCO_PA): 4200• Gateway server (NCO_GATE): 4300• Proxy server (NCO_PROXY): 4400	<p>These default port numbers are defined in the Server Editor, but they are configurable and rarely used with the default values. Amend the port numbers as necessary, and then save your changes.</p> <p>On UNIX systems that do not have a graphical interface, you can amend the port numbers by editing the <code>\$NCHOME/etc/omni.dat</code> file.</p> <p>For information about amending port numbers, see “Configuring server communication details in the Server Editor” on page 240.</p>
IDUC: Variable value	<p>The operating system supplies the port number. To change the port number, perform any of the following actions:</p> <ul style="list-style-type: none">• Edit the Iduc.ListeningPort property in the <code>\$NCHOME/omnibus/etc/servername.props</code> file, where <i>servername</i> is the ObjectServer name.• Use the command-line option <code>-listeningport</code> when running the <code>nco_objserv</code> command.• Specify the port in the <code>/etc/services</code> file on the host workstation. <p>For information about using this property or command-line option, see the <i>IBM Tivoli Netcool/OMNIBus Administration Guide</i>.</p>
IBM Eclipse Help System (IEHS)	<p>The default port number used to access the IEHS server is 8888.</p> <p>The server can typically be accessed by specifying the following address:</p> <p><code>http://IP_address:port</code></p> <p>Where <i>IP_address</i> is the IP address of the host computer, and <i>port</i> is 8888.</p>
Probes: See the individual probe publications	<p>The port numbers for individual probes vary, and they are documented in the publication for each specific probe.</p> <p>From the information center at <code>http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp</code>, you can access probe publications as follows: expand the <i>IBM Tivoli Netcool/OMNIBus</i> node in the navigation pane on the left, and go to the <i>Tivoli Netcool/OMNIBus probes and TSMs</i> node.</p>

Appendix E. Web GUI initialization file properties

The environmental and server session properties of the Web GUI server are stored in the *webgui_home_dir/etc/server.init* initialization file. This file is an ASCII initialization file that can be edited directly and is read on server startup.

After you have edited the *server.init* file, restart the Tivoli Integrated Portal server.

The properties contained within the *server.init* file are listed alphabetically.

A

admin.stylesheets

System file location - do not modify.

The default value is *%%/etc/system/stylesheets/*.

ael.top-n.mode

Specifies the top-n mode. Possible values are as follows:

- 1: StateChange will be appended to AEL SQL event update requests.
- 0: StateChange will not be appended to AEL requests.

The default value is 0.

ael.top-n.refresh

Specifies the type of refresh. Possible values are as follows:

- 1: The AEL is only updated with new and updated rows from the ObjectServer. StateChange is enforced to be greater than 0. The running of SQL tools and the checks for discrepancy between the AEL and the ObjectServer do not reset StateChange to 0.
- 0: If the number of rows in the AEL and the number of rows in the ObjectServer do not match, the AEL is refreshed with data from the ObjectServer up to the number of rows specified in the **ael.top-n.value** property, and StateChange is reset to 0. The running of SQL tools resets StateChange to 0.

The default value is 0.

ael.top-n.value

The Web GUI supports the TOP keyword in ObjectServer SQL syntax. The **ael.top-n.value** property allows Web GUI administrators to impose a limit on the number of alerts returned to the AEL. If this property is set to a value greater than 0 (zero), AEL queries are modified to include a TOP condition. For example, if an AEL filter matches 8000 rows in the ObjectServer, and the **ael.top-n.value** value is set to 4000, only the top 4000 alerts are displayed. Displaying more than 20,000 events in a single AEL might impact performance. The AEL status bar displays the total count of alerts for each severity level, and the total count of alerts displayed. A Top Set to message is also shown above the distribution status bar in the AEL indicating that a TOP condition is being applied. The **ael.top-n.value** property can be considered for systems:

- That regularly contain a high volume of events
- Where AEL filters match more than 20000 alerts
- Where the number of concurrent AEL users is adversely affecting system performance

If the value of the **ael.top-n.value** is set to 0 (zero), this value is ignored; the AEL will not show zero rows.

The default value is 0.

aelview.queries.enabled

When set to true, the user can make advanced queries against the AELView servlet by adding configuration criteria to the URL containing the AEL address.

The default value is true.

B

browser.prp

System file location - do not modify.

The default value is `%%/etc/browsers.prp`.

C

cluster.hostname

The identity of the host that the Web GUI server is running on. The value is the host name or its TCP/IP address. Set this property only when **cluster.mode** is set to on.

cluster.mode

Indicates whether the Web GUI server is operating in a load-balancing cluster. The possible values are as follows:

- on: The server is part of a cluster.
- off: The server is a stand-alone system.

When this property has the value on, provide values for **cluster.hostname** and **cluster.port** also.

The default value is off.

cluster.port

The SSL port that the Web GUI server uses. The value is a numeric port value. Set this property only when **cluster.mode** is set to on.

cluster.waapi.notification.delay

Defines a delay period (in milliseconds) before notifying other nodes in the cluster of configuration changes made using WAAPI command files. The default value is 2000.

D

datasource.failback.delay

Specifies the time to wait after a failback before the Web GUI switches back to the primary ObjectServer. During this wait period, the backup ObjectServer is used. You can adjust this value depending on the latency of a tiered ObjectServer architecture

The default value is 120.

datasource.response.timeout

The timeout threshold, in milliseconds, for checking the response time of a data source associated with a map.

The default value is 3000.

E

ee.entitydir

System file location - do not modify.

The default value is `%%/etc/entities/`.

eventprovider.eventsummarydataservice.threadpool.size

Specifies the default thread pool size for the Event Summary Data service. If you increase the value of this property beyond the default, performance might be impaired.

The default value is 20.

eventprovider.eventdataservice.threadpool.size

Specifies the default thread pool size for the Event Data service. If you increase the value of this property beyond the default, performance might be impaired.

The default value is 20.

F

fips.security.key

The name of the Web GUI security key file. The default value points to the default Tivoli Integrated Portal security key.

The default value is `%%/etc/encrypt/vault.key`.

G

groups.reload.mode

A setting for the algorithm used to request a list of Web GUI groups from the authentication system. Possible values are as follows:

- 0: All groups are requested.
- 1: Only groups with role names that begin with `ncw_` are requested.

The default value is 1.

I

illegalchar.file

This file defines characters that are not permitted in the names of filters, views, and tools, and characters that cannot be used as the initial character in the names of filters, views, and tools.

The default value is `%%/etc/illegalChar.prop`.

internationalisation.cache.enabled

Specifies whether the Web GUI server caches language resources in memory. When set to false, this prevents caching of localization data, and forces the server to regularly re-read the configuration files for the selected locale.

The default value is true.

L

lel.pagesize.default

Specifies the number of rows returned per page in the LEL.

The default value is 500.

log.count

The maximum number of log files to retain.

The default value is 5.

log.directory

The directory in *tip_home_dir*/profiles/TIPProfile/ that contains the log and trace files. Do not modify this property.

The default value is /logs/ncw

log.filename

The name of the log file. Do not modify.

The default value is ncw.%g.log

log.level

The minimum severity of events to record in the log file. The possible values are:

- NONE
- FINEST
- FINER
- FINE
- CONFIG
- INFO
- AUDIT
- WARNING
- SEVERE
- ALL

The default value is INFO.

log.maxsize

The maximum size of the log file in megabytes.

The default value is 10.

logfile

System file location - do not modify.

The default value is <data>%%/log/webtop.log</data>.

M

maplet.noeventcolor

Specifies the color of active elements that have no associated events. Specify a hexadecimal color value for this parameter, for example 0xDDDDDD for gray or 0xFFFFFFFF for white. If no value is specified, the color associated with severity 0 is used.

The default value is None.

maplet.plugin.classic

Specifies HTML markup for embedding map objects:

- If true, the map is embedded with the <APPLET> element, and the default Netscape or Internet Explorer JVM is used.
- If false, the map is embedded with the <OBJECT><EMBED> elements.

The default value is false.

maplet.refresh

Specifies the time interval, in seconds, between map object refreshes. Do not

set the value of this property to a value less than 10. In addition, if your site uses complex maps, use a higher value for this property.

The default value is 10.

maps.directory

System file location - do not modify.

The default value is `%%/etc/maps/`.

maxtablesize

The maximum number of rows allowed in a table.

The default value is 200.

metricdataservice.threadpool.size

Specifies the default thread pool size for the Metric Data service. If you increase the value of this property beyond the default, performance might be impaired.

The default value is 20.

P

passwd.file

System file location - do not modify.

The default value is `%%/etc/users/passwds`.

plugin.classid

Specifies the version of the Java plug-in used by AEL applets by using the classid attribute of the OBJECT tag, and allows you to enforce which plug-in is used. If the `maplet.plugin.classic` property is set to false, and the user has an older version of the plug-in than shown in the classid attribute of the <OBJECT> element, the user is prompted to download the newer version. If the user has the same or a newer version, that version is used.

The default value is `clsid:8AD9C840-044E-11D1-B3E9-00805F499D93`.

plugin.iedownload

Specifies the full URL pointing to a .cab file from which the Java plugin can be installed. This ensures that the client has the appropriate plug-in version. If the plug-in version is not correct, the user is automatically redirected to the latest .cab for the latest version in the family. This is used in the <OBJECT> tag for Windows Internet Explorer.

The default value is `http://java.sun.com/update/1.5.0/jinstall-1_5_0_11-windows-i586.cab`.

plugin.page

Specifies the full URL pointing to a Web page from which the Java plugin can be downloaded if the appropriate version is not already installed. This is used in the <EMBED> element for Mozilla browsers.

The default value is `https://java.sun.com/products/plugin/index.jsp`.

plugin.type

Specifies the version of the Java plug-in used by the AEL by using the type attribute of the <EMBED> element, and allows you to enforce which plugin is used. If the `maplet.plugin.classic` property is set to false, and the user has a lower version than specified in this property, then they are prompted to download the newer version. If the user has the same or a higher version, that version is used.

The default value is `application/x-java-applet;version=1.5`.

profile.count

The maximum number of profile log files to retain.

The default value is 5.

profile.filename

The name of the profile log file. Do not modify this property.

The default value is `ncw.%g.profile`

profile.maxsize

The maximum size of the profile log file in megabytes.

The default value is 10.

profilereport.runperiod

Defines the frequency (in seconds) for generating the profile report. The default value is 60.

profilereport.startdelay

defines the length of time (in seconds) before generating the first profile report.

R

resources.directory

System file location. Do not modify.

The default value is `%%/etc/resources/`.

S

server.mode

Defines whether to make certain Web GUI features unavailable. The features are defined in the file `webgui_home_dir/etc/restricted_urls.lst`. Possible values are as follows:

- 0: The server runs in normal mode. All Web GUI features are available.
- 1: The server runs in restricted mode. The URLs that match patterns in `restricted_urls.lst` are not available to users.

The default value is 0.

T

tableview.escapehtml

Prevents rendering of HTML script in Table View fields. If true, HTML script text is treated as simple text in the Table View fields. If false, HTML script text is rendered in the Table View fields.

The default value is false.

tableview.pixelmultiply

Optional parameter passed to Table Views and for table rendering.

The default value is 10.

tableviewparams

Optional parameters that are passed to Table Views and which govern table rendering.

The default value is `border="0" cellpadding="1" cellspacing="1" width="100%"`.

timedtasks.default.runperiod

The run period (in seconds) for configstore update timer tasks.

The default value is 120.

timedtasks.default.startdelay

The start delay (in seconds) for configstore update timer tasks.

The default value is 120.

timedtasks.enabled

Indicates if timed tasks are enabled or disabled. Can be true or false.

The default value is false.

trace.count

The maximum number of trace files to retain.

The default value is 5

trace.filename

The name of the trace file. Do not modify this property.

The default value is ncw.%g.trace.

trace.level

The minimum severity of events to record in the trace file. The possible values are:

- NONE
- FINEST
- FINER
- FINE
- PROFILE
- CONFIG
- INFO
- AUDIT
- WARNING
- SEVERE
- ALL

The default value is FINE.

trace.maxsize

The maximum size of the log file in bytes. Use the suffixes M or K to indicate megabytes and kilobytes, respectively.

The default value is 100M.

U

uploadfile.maxsize

The maximum size of a file that the Page Manager can load, in megabytes.

the default value is 5.

users.credentials.sync

Specifies whether the automatic synchronization of user credentials between VMM and the ObjectServer is enabled. If this property is set to true, synchronization is enabled.

The default value is false.

users.credentials.sync.groupname

Specifies the name of the user group that is used in the ObjectServer if the automatic synchronization of user credentials between VMM and the ObjectServer is enabled. All synchronized users are members of this group.

The default value is `vmusers`.

users.global.filter.mode

Setting for non-administrative user permissions to modify global filters. Possible values are as follows:

- 0: Non-administrative users cannot add, modify, or delete global filters.
- 1: Non-administrative users can add and modify global filters, but cannot delete them.

The default value is 1.

users.global.view.mode

Setting for non-administrative user permissions to modify global views. Possible values are as follows:

- 0: Non-administrative users cannot add, modify, or delete global views.
- 1: Non-administrative users can add and modify global views, but cannot delete them.

The default value is 1.

users.reload.mode

A setting for the algorithm used to request a list of Web GUI users from the user authentication system. Possible values are as follows:

- 0: All users are requested. This option enables faster data retrieval.
- 1: Only users with role names that begin with `ncw_` are requested. This option can be slow if there is a large number of system users.

The default value is 1.

utility.debug

Sets the debug level, in increasing detail from 0 (critical messages only) to 9 (all messages).

The default value is 0.

utility.debug.destination

Sets the destination for debug messages. The options are:

- `stdout`: Standard output
- `stderr`: Standard error output
- `log`: The log file specified by the `log.filename` option in `server.init`

The default value is `log`.

utility.monitor

Sets the user monitoring level, in increasing detail from 0 (critical messages only) to 9 (all messages).

The default value is 0.

utility.monitor.destination

Sets the destination for monitor messages. The options are:

- `stdout`: Standard output
- `stderr`: Standard error output
- `log`: The log file specified by the `log.filename` option in `server.init`

The default value is log.

V

views.directory

System file location. Do not modify.

The default value is `%%/etc/views/`.

W

webtop.fips

Enables FIPS 140–2 mode and can be on or off.

Note: If set to on, FIPS 140–2 must also be set up in Tivoli Integrated Portal.

The default value is off.

webtop.keepalive.interval

The Web GUI server periodically pings the Tivoli Integrated Portal server to avoid the server timing out when an AEL or Maplet page is still active. This property specifies the time period (in minutes) between each ping operation. The default value is 3.

webtop.password.encryption

Sets the passwords stored in `server.init` to be encrypted. Possible values are as follows:

- none: Passwords in `server.init` are not encrypted.
- aes: Passwords in `server.init` can be encrypted by using the **ncw_aes_crypt** tool.
- fips: Passwords in `server.init` can be encrypted by using the **ncw_aes_crypt** tool.

Note: If FIPS 140–2 has been enabled for Web GUI, you can chose only none or FIPS.

The default value is none.

webtop.ssl.trustManagerType

The type of trust manager used. Set this property to `IbmX509` if you are using the JRE bundled with Web GUI or an AIX JRE. Set this property to `SunX509` if you are not using the bundled JRE or an AIX JRE.

The default value is `IbmX509`.

webtop.ssl.trustStore

Sets the location of the SSL truststore used by the Web GUI. If no value is entered, the Tivoli Integrated Portal default truststore is used, which also gives you access to the Tivoli Integrated Portal truststore UI for truststore configuration.

webtop.ssl.trustStorePassword

Sets the password used to access the truststore. If left blank, no password is required to access the truststore. For PKCS12 store types (set in `webtop.ssl.trustStoreType`) a password must be provided. For JKS store types (set in `webtop.ssl.trustStoreType`), a password is optional.

webtop.ssl.trustStoreType

The type of truststore used.

The default value is `PKCS12`.

Related tasks

“Changing data source configurations” on page 512

Notices

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

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

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

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

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

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

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

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

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

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

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

IBM Corporation
958/NH04
IBM Centre, St Leonards
601 Pacific Hwy
St Leonards, NSW, 2069
Australia

IBM Corporation
896471/H128B
76 Upper Ground
London SE1 9PZ
United Kingdom

IBM Corporation
JBF1/SOM1
294 Route 100
Somers, NY, 10589-0100
United States of America

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 document 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.

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.

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

Trademarks

AIX, DB2, IBM, the IBM logo, ibm.com®, iSeries, Netcool, Passport Advantage, pSeries, Service Request Manager, System p, System z, Tivoli, Tivoli Enterprise Console, TotalStorage, WebSphere, and zSeries are trademarks or registered 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 logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.



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

Linux is a registered trademark of Linus Torvalds 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.

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

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

Index

Special characters

Restricted user group
Delete tool migration 204

A

AboutPortlet
migration 204
accessibility xii
adding
backup ObjectServers 245
certificates 387
encrypted values to properties
files 358
administration tools
overview 3
AELAction
migration 204
AELPortlet
migration 204
AIX
DISPLAY environment variable 528
application server
FIPS enablement 497
application.sql file 230
arch
operating system directory xiii
attributes
configuration files 526
audience ix
audit trails 355
authentication
LDAP 340
PAM 345
authentication security 337
authorization 350
authorization security 338
automated failover and failback 230
automation
description 2
automation.sql file 230

B

BAROC conversion tool 91, 152
BAROC data 90, 151
migration 92, 153
bundles 453
creating 455, 457

C

CA (Certificate Authority) 377
CA certificates
activating 494
adding 387, 491
adding to the store 493
obtaining 492
receiving 493

CA certificates (*continued*)
requesting 491
certificate 210
Certificate Authority (CA) 377
certificate details
viewing 393
certificate management guidelines 382
certificate request file
signing 390
certificate requests 378
certificates
activating 494
adding 387, 491, 494
adding to the store 493
assigning 495
deleting 393
extracting 386
generating 494
migration 87, 148
obtaining 492
receiving 391, 493
replacing 491
requesting 389, 491
changing
key database password 394
priority of backup ObjectServers 247
ChartAction
migration 204
ChartPortlet
migration 204
checklist
FIPS 140-2 configuration 26
Commands
Deployment Engine 627
communication protocol 19
communications
distributing the interfaces file to
multiple platforms 249
compatibility
gateways 19
licensing 19
previous versions 19
components 38
Tivoli Netcool/OMNIBus 1
configuration bundles 453
configuration files
attributes 526
DTD 520
elements 523
structure 519
XML 520
configuration list files
creating 309
editing 312
example 313
using 305, 309
configuration packages 305
configuration requirements
FIPS 140-2 mode 298
configuring
communications using SSL 377

configuring (*continued*)
desktop ObjectServer 329
FIPS 140-2 mode 295, 298
FIPS 140-2 mode for servers 295
IPv6 399
JRE for FIPS 140-2 102, 158
load balanced mode 334
localized sorting 407
predictive eventing 428
server communication
information 242
TADDM events 438
virtualization 447
Web GUI 477
configuring VMM 485
Conpack utility
overview 3
connections data file 247
editing for SSL 379
console commands 182
console installation mode
Linux 52
UNIX 52
Windows 117, 133
console installer for the Web GUI 180
console mode commands 182
console uninstaller for the Web GUI 215
console upgrade mode
Linux 72
UNIX 72
control_shutdown directory 413
controlled failback 287
conventions, typeface xiii
conversion
UTF-8 146
creating
bundles 455, 457
configuration list files 309
deployable bundles 455, 457
desktop ObjectServer 329
key database 383
ObjectServer 231
self-signed certificate 385
stash file 383
CRL files
omni.crl 381

D

DAT files
omni.dat 247
data source
configuration file 512
data sources
dual server desktop 518
multiple 517
samples 512
database initialization
files 230
nco_dbinit 229
DB2 539

- deduplication
 - description 2
- default certificate
 - specifying 392
- default groups 353
- default port numbers 631
- default roles 351
- default users 354
- Delete tool
 - Restricted* user group 204
 - migration 204
- deleting
 - certificates 393
- deploying probes 460
 - multiple computers 463, 465
 - single computer 462, 465
- deploying Tivoli Netcool/OMNIBus 460, 461
- Deployment Engine
 - commands 627
 - common directory structure 35
 - installation 35
 - installation directory structure 35
 - multi-user mode 36
 - non-root user installation 34
 - overview 34
 - root user installation 34
 - single-user mode 36
 - user modes 36
- Deployment Engine (DE) 226
- desktop ObjectServer
 - architecture overview 327
 - authentication in 333
 - configuring 329
 - configuring load balanced mode 334
 - configuring unidirectional gateway 330
 - considerations 329
 - creating 329
 - dual-write mode 332
 - load balanced mode 333
 - manual journal entries 333
 - overview 327
- desktop tools
 - overview 3
- desktop.sql file 230
- digital certificates
 - activating 494
 - adding 387, 491, 494
 - adding to the store 493
 - assigning 495
 - deleting 393
 - extracting 386
 - generating 494
 - migration 87, 148
 - obtaining 492
 - receiving 391, 493
 - replacing 491
 - requesting 389, 491
 - viewing details 393
- directory structure
 - installation packages 226
- disk space requirements 17
- distributed installations
 - configuring for SSL connections 379
 - introduction 248

- downloading
 - IBM Support Assistant (ISA) 605
- DTD
 - for configuration files 520
- dual-write mode
 - desktop ObjectServer 332

E

- editing
 - configuration list files 312
- education
 - see Tivoli technical training xiii
- elements
 - configuration files 523
- encrypted values
 - adding to properties files 358
- encrypting
 - passwords for process control 339
 - passwords for SQL scripts 340
 - strings 357
 - values in properties files 356
 - Web GUI 486
- environment variables
 - DISPLAY 528
 - LANG 403
 - LC_ALL 403
 - LC_COLLATE 403
 - LC_CTYPE 403
 - LC_MESSAGES 403
 - LC_MONETARY 403
 - LC_NUMERIC 403
 - LC_TIME 403
 - LD_LIBRARY_PATH 95, 99
 - LIBPATH 95, 99
 - NCHOME 34, 95
 - NCO_JRE 14
 - OMNIHOME 95
 - PATH 95
 - SHLIB_PATH 95, 99
- environment variables, notation xiii
- ETai 501
- eventflood directory 413
- exclusions file
 - example 317
 - overview 317
- export module
 - migration 189
 - upgrade 188
- exporting
 - ObjectServer configurations 301
- external authentication 341, 346
- extracting
 - certificates 386

F

- failback
 - automated 230
- failover
 - automated 230
- failover configuration 285
- federated repositories
 - VMM for ObjectServer 485
- FIPS 140-2
 - configuration checklist 26

- FIPS 140-2 (*continued*)
 - introduction 26
- FIPS 140-2 configuration 102, 158
- FIPS 140-2 mode
 - backward compatibility 298
 - configuring 295
 - creating configuration file 295
 - Web GUI 496
- FIPS 140-2 mode
 - enabling for SSL 380
- FIPS support 497

G

- gateways
 - configuring servers 241
 - FIPS 140-2 configuration 295
 - overview 3
 - security 339
 - uninstalling 108, 172
- generating
 - keys 356
- groups
 - default 353
 - row level security 354
- GUI installer for the Web GUI 178
- GUI uninstaller for the Web GUI 215
- guidelines
 - converting to UTF-8 146

H

- high availability 285
 - controlled failback 287
 - controlled shutdown 289
 - failover configuration 285
 - proxy server failover 293
 - reducing event loss 288
 - reducing resync time 289
- HP-UX
 - DISPLAY environment variable 528
- HTTP and HTTPS 486
- HTTP server
 - configuring 547
 - downloading 539
- HTTP server plug-in SSL configuration
 - load balancing 553

I

- IBM Key Management (iKeyman)
 - overview 380
- IBM Support Assistant (ISA)
 - data collection 605
 - downloading 605
 - installation 605
 - Log and Trace Analyzer 605
 - overview 605
 - Tivoli Netcool/OMNIBus plug-in 605
- IEHS server 101, 157
 - starting 101, 157
 - stopping 101, 157
- iKeyman
 - overview 380
 - starting 382

- illegal characters
 - migration 191
- import and export utility
 - overview 3
- import module 188
 - migration 189
- importing
 - ObjectServer configurations 301, 320
- information center mode 100, 156
- infrastructure 38
- initialization file
 - properties 633
- install 182
 - remove by console uninstaller 215
 - remove by GUI uninstaller 215
 - remove by silent mode 217
- installable features 45, 111
- installation 173, 501
 - console mode on Linux 52
 - console mode on UNIX 52
 - console mode on Windows 117, 133
 - deployment engine
 - failure after upgrade 225
 - directory structure 61, 125, 226
 - errors 223
 - existing 185
 - failure after DE upgrade 225
 - for single sign-on 30
 - gateways 103, 104, 105, 159, 160, 161
 - harmless messages 220
 - IBM Support Assistant (ISA) 605
 - installation wizard on Linux 49
 - installation wizard on UNIX 49
 - installation wizard on Windows 115
 - installed packages 61, 124, 220
 - log file 124
 - log files 60, 223
 - logs 218
 - Netcool/OMNIBus 59
 - probes 103, 104, 105, 159, 160, 161
 - response file 55, 120
 - silent mode 57, 122
 - silent mode on Linux 54
 - silent mode on UNIX 54
 - silent mode on Windows 119
 - specifying silent mode settings 55, 120
 - supported JVMs 14
 - supported Web browsers 14
 - troubleshooting
 - installation errors 223
 - Web GUI 218
 - vault key file protection 210
 - Web GUI
 - console installer 180
 - silent installer 181
 - Web GUI prerequisites 174
 - Windows services 162
- Installation
 - Web GUI
 - GUI installer 178
- installation information 175
- installation package
 - downloading 47, 113
 - Web GUI 174
- installation prerequisites
 - Linux 47

- installation prerequisites (*continued*)
 - UNIX 47
 - Windows 113
- installation wizard
 - Linux 49
 - UNIX 49
 - Windows 115
- installer
 - launchpad 33, 178
 - modes 33
- interfaces file
 - adding a backup ObjectServer 245
 - changing priority of backup ObjectServer 247
 - configuring gateway servers 241
 - configuring process agents 241
 - distributing to multiple platforms 249
 - hiding backup ObjectServers 247
 - SSL connections 378
 - testing availability of a server 247
- IPv4
 - support 18
- IPv6
 - configuring 399
 - HP-UX configuration
 - requirements 18, 399
 - probe rules file configuration 399
 - restrictions 18
 - support 18
 - UNIX configuration 399
 - Windows configuration 399
- ISA (IBM Support Assistant)
 - data collection 605
 - downloading 605
 - installation 605
 - Log and Trace Analyzer 605
 - overview 605
 - Tivoli Netcool/OMNIBus
 - plug-in 605
- itmpredictive directory 413
- itmvirtualization directory 413

J

- journal entries
 - desktop ObjectServer 333
- JRE configuration
 - FIPS 140-2 102, 158
- JRE requirements 14

K

- KDB files
 - omni.kdb 381
- KDY.INSTALLDIR property 453
- key database files 381
- key database password
 - changing 394
- key databases
 - creating 383
- key files 357
- keys
 - generating 356

L

- LANG environment variable 403
- launch-in-context 557
- launchpad 33
- LC_ALL environment variable 403
- LC_COLLATE environment variable 403
- LC_CTYPE environment variable 403
- LC_MESSAGES environment
 - variable 403
- LC_MONETARY environment
 - variable 403
- LC_NUMERIC environment
 - variable 403
- LC_TIME environment variable 403
- LD_LIBRARY_PATH environment
 - variable 95, 99
- LDAP 340, 487
 - adding 481
 - adding OpenLDAP 482
 - configuring 341
 - enabling external authentication 341
 - prerequisites 340
 - properties 343
- LDAP (Lightweight Directory Access Protocol)
 - migration 191
- LELAction
 - migration 204
- LELPortlet
 - migration 204
- LIBPATH environment variable 95, 99
- Lightweight Directory Access Protocol 340
- Linux installation directory structure 61
- load balanced mode 333
 - configuring 334
- load balancing 278
 - add node 555
 - clone IDs 549, 550
 - configuration setup 535
 - HTTP server 539
 - preparation 539
 - remove node 555
 - server-to-server trust 545
 - starting Web GUI operations 554
- load balancing cluster
 - add node 555
 - join 547
 - remove node 555
- locales
 - setting 403
- localization environment variables 403
- localized sorting
 - configuring 407
- log
 - TIPProfile_create 219
- login
 - configure for HTTP and HTTPS 486
 - multiple to one user 568
- logon 208, 600
- logs
 - installation 218

M

- manual journal entries
 - desktop ObjectServer 333
- manual migration 80
- manuals xi
- MapAction
 - migration 204
- MapPortlet
 - migration 204
- migrating
 - Web GUI
 - from Netcool/Webtop version 1.3 199, 225
- migration
 - authorization 204
 - components 204
 - digital certificates 87, 148
 - files not migrated 208
 - illegal characters 191
 - keys 87, 148
 - layouts 204
 - Linux 80
 - localized pages 204
 - log file 80
 - migrated files 85, 144
 - nco_ssl_migrate 87, 148
 - PSML files 208
 - rollback 202
 - security IDs 204
 - security properties 191
 - UNIX 80
 - WAAPI client prerequisites 191
- migration log file 140
- migration mode 189
- migration tool
 - components and modes 188, 189
- mobile devices 14
- multicultural support
 - configuring fonts 407
 - identifying supported locales 406
 - locales for UNIX desktop 407
 - localization environment
 - variables 403
 - localized sorting 407
 - setting locale 403
 - using translated UI text 410
 - UTF-8 Windows encoding 403
 - Web GUI 213
- multiple login to a user account 568
- multitier directory 413
- multitiered architecture 251
 - additional backup collection ObjectServer 273
 - additional primary collection ObjectServer 271
 - additional unidirectional backup collection gateway 274
 - additional unidirectional primary collection gateway 272
 - backup aggregation ObjectServer 262
 - backup collection ObjectServer 265
 - bidirectional aggregation
 - gateway 263
 - creating custom triggers 279
 - display ObjectServer 266, 268
 - file locations 258
 - final steps 283

- multitiered architecture (*continued*)
 - load balancing 278
 - more collection ObjectServers 270
 - more display ObjectServers 275, 276
 - more unidirectional display gateways 277
 - naming conventions 253
 - number of ObjectServers 255
 - performance triggers 280
 - primary aggregation
 - ObjectServer 261
 - primary collection ObjectServer 263
 - Resynchronization Complete
 - events 282
 - severity handling 256
 - standard configuration 251
 - unidirectional backup collection
 - gateway 266
 - unidirectional display gateway 267, 269
 - unidirectional primary collection
 - gateway 264

N

- naming conventions
 - ObjectServer 230
- nc_gskcmd 395
- NCHOME environment variable 34, 95
- nco_aes_crypt 357
 - command-line options 358
- nco_baroc2sql 91, 152
 - command-line options 92, 153
- nco_cftp 469
- nco_cftp.props 473
- nco_confpack 301
 - command-line options 307
 - configuration list files overview 305, 309
 - configuration package overview 305
 - creating backup ObjectServer configurations 318
 - creating configuration list files 309
 - editing configuration list files 312
 - exclusions file 317
 - exporting ObjectServer configurations 313
 - import considerations 323
 - importable and exportable items 306
 - importing ObjectServer configurations 320
 - properties 307
 - viewing configuration package contents 320
- nco_dbinit 229, 231, 329
 - command-line options 232
 - properties 232
- nco_igen 247
- NCO_JRE environment variable 14
- nco_keygen 356
- nco_objserv 237
- nco_pa_crypt 339
- nco_sql 239
- nco_sql_crypt 340
- nco_ssl_migrate 87, 148
- NCOM5.props 237

- NCOS (IBM Tivoli Netcool/OMNIBus ObjectServer)
 - migration 191
- Netcool home location 34

O

- object permissions 351
- ObjectServer 485
 - adding backup 245
 - automated failback 230
 - automated failover 230
 - automation 2
 - changing priority of backup 247
 - command-line options 232
 - configuration options 238
 - creating 231
 - creating backup configurations 318
 - database directory 230
 - database initialization 229
 - deduplication 2
 - desktop ObjectServer
 - architecture 327
 - exporting configurations 313
 - FIPS 140-2 configuration 295
 - hiding backup ObjectServers 247
 - importing configurations 320
 - isql 239
 - naming conventions 230
 - nco_dbinit 231
 - nco_objserv 237
 - nco_sql 239
 - overview 2, 229
 - properties 232
 - properties file 231
 - secure mode 339
 - SSL connection 488
 - starting manually 237
 - starting using process control 236
 - stopping manually 239
 - stopping using process control 238
 - stopping using services 239
- omni.crl 381
- omni.dat file
 - editing 247
- omni.kdb 381
- omni.rdb 381
- omni.sth 381
- OMNIHOME environment variable 95
- online help
 - configuring 100, 156
 - information center mode 100, 156
 - requirements 16
 - running IEHS server 101, 157
 - standalone mode 100, 156
 - starting IEHS server 101, 157
 - stopping IEHS server 101, 157
 - UNIX environment variables 100
 - Web browsers 16
- online publications xi
- operating system directory
 - arch xiii
- ordering publications xi
- overview 6

P

- package bundles 453
- PAM
 - configuration file 349
 - configuring 346
 - configuring ObjectServer 348
 - configuring ObjectServer as authentication source 348
 - enabling external authentication 346
 - modifying ObjectServer settings 349, 350
 - ObjectServer PAM configuration file 350
 - troubleshooting 624
- PAM (Pluggable Authentication Modules) 345
- password encryption 339, 340
 - Web GUI server, AES 495
 - Web GUI server, FIPS 140-2 mode 499
- PATH environment variable 95
- permissions
 - object 351
 - system 351
- Pluggable Authentication Modules (PAM) 345
- port numbers
 - default 631
- postinstallation tasks
 - UNIX 94
 - Windows 155
- predictive eventing 428, 532
- predictive events
 - resources 425
- prerequisites
 - installation 47, 113
 - LDAP 340
 - migration 191
 - remote deployment 452
 - upgrade 47, 113, 189
- probes
 - overview 3
 - security 339
 - uninstalling 108, 172
- process agents
 - configuring servers 241
 - FIPS 140-2 configuration 295
- process control
 - FIPS 140-2 configuration 295
 - overview 3
 - security 339
 - starting an ObjectServer 236
 - stopping an ObjectServer 238
 - Windows services 237
- property value encryption 356, 357, 358
- proxy server failover 293
- proxy servers
 - FIPS 140-2 configuration 295
- PSML files 208
- publications xi

R

- RDB files
 - omni.rdb 381

- receiving
 - certificates 391
- remote deployment 451
 - configuration bundle 453
 - creating bundles 455, 457
 - deploying a probe 462, 463
 - deploying multiple probes 465
 - deploying probes 460
 - deploying Tivoli Netcool/OMNIBus 460, 461
 - monitoring status 466
 - nco_cftp 469
 - nco_cftp command-line options 473
 - nco_cftp properties 473
 - package bundle 453
 - prerequisites 452
 - running probes 467
 - tacmd commands 454
 - transferring files 469
 - updating files 469
 - workflow 453
- requesting
 - certificates 389
- requirements
 - disk space 17
 - JRE 14
 - online help 16
 - user interface 16
- response file 55, 75, 120, 136
- roi directory 413
- roles
 - default 351
- rollback
 - migration 202
- rollback mode 188
- migration 189

S

- secure connections
 - in non-FIPS mode 490
 - with FIPS 140-2 500
- secure mode
 - ObjectServer 339
 - proxy server 339
- secure sockets layer (SSL) protocol 377
- security
 - audit trails 355
 - authentication 337, 338
 - authorization 338
 - certificate 210
 - gateways 339
 - Lightweight Directory Access Protocol 340
 - migration properties 191
 - Pluggable Authentication Modules 345
 - probes 339
 - process control 339
 - proxy server 339
 - SQL interactive interface 340
 - user access 337
- security IDs
 - migration 204
- security.sql file 230
- self-signed certificate
 - creating 385
- server
 - password encryption, AES 495
 - password encryption, FIPS 140-2 mode 499
- server certificates
 - activating 494
 - adding 491, 494
 - adding to the store 493
 - assigning 495
 - deleting 393
 - generating 494
 - obtaining 492
 - receiving 391, 493
 - replacing 491
 - requesting 389, 491
 - specifying default 392
 - viewing details 393
- server communication information 240, 242
- Server Editor 240
 - adding backup ObjectServers 245
 - changing priority of backup ObjectServers 247
 - configuring gateway servers 241
 - configuring process agents 241
 - configuring SSL on UNIX 378
 - configuring SSL on Windows 378
 - connections data file 247
 - creating server definition entries 245
 - distributing the interfaces file 246
 - hiding backup ObjectServers 247
 - server communication information 240, 242
 - testing server availability 247
- server.init
 - properties 633
- settings.properties 191
- shared library paths
 - checking 95, 99
- SHLIB_PATH environment variable 95, 99
- shutdown command (ObjectServer) 239
- signing
 - certificate request file 390
- silent installation mode
 - Linux 54
 - UNIX 54
 - Windows 119
- silent installer for the Web GUI 181
- silent mode
 - response file 55, 120
- silent uninstaller for the Web GUI 217
- silent upgrade mode
 - Linux 75
 - UNIX 75
 - Windows 136
- single sign-on 30, 529
 - configuring 529, 530
 - ETai trust association 502, 503
 - import LTPA keys 531
 - installing ETai 502
 - maintaining LTPA keys 529
 - procedures 530
- single sign-onexport LTPA keys 531
- specifying
 - default certificate 392
 - key file as property 357

- SQL files 230
- SQL interactive interface 239
 - overview 3
 - security 340
- SSL
 - certificate management
 - guidelines 382
 - configuring 487, 553
 - distributed installations 379
 - editing connections data file 379
 - enabling FIPS 140-2 mode 380
 - HTTP server plug-in 553
 - key database files 381
 - managing digital certificates 380, 491
 - replacing client certificate 491
 - SSL 487
 - starting iKeyman 382
 - to ObjectServer 488
- SSL (secure sockets layer) 377
- standalone mode 100, 156
- starting 101, 157
 - ObjectServer 236, 237
- stash files 381
 - creating 383
- STH files
 - omni.sth 381
- stopping 101, 157
 - ObjectServer 238, 239
- strings
 - encrypting 357
- support information xiii
- supported operating systems 11
- system permissions 351
- system.sql file 230

T

- TableviewAction
 - migration 204
- TableviewPortlet
 - migration 204
- tacmd commands 454
- taddm directory 413
- TADDm events 435, 438
 - configuration setup 436, 534
 - resources 438
- testing
 - server availability 247
- TIPProfile_create.log 219
- Tivoli Access Manager WebSEAL 501
- Tivoli Enterprise Console
 - BAROC data migration 90, 151
- Tivoli Netcool/OMNIBus
 - components 1
 - default port numbers 631
 - overview 1
- Tivoli software information center xi
- Tivoli technical training xiii
- training, Tivoli technical xiii
- troubleshooting 613
 - multicultural support 626
 - nco_pad 624
 - PAM 624
 - root access 623
 - Web GUI
 - installation 221
 - migration 225
- troubleshooting (*continued*)
 - Web GUI (*continued*)
 - uninstallation 221
 - upgrade 203
 - user registries 222
 - typeface conventions xiii
 - types of 173
 - types of installation 173

U

- unidirectional gateways
 - using in desktop ObjectServer
 - architecture 330
- uninstall 214
- uninstalling
 - console mode 108, 171
 - gateways 106, 108, 169, 172
 - probes 106, 108, 169, 172
 - Tivoli Netcool/OMNIBus 106, 169
 - Tivoli Netcool/OMNIBus
 - services 170
 - Web GUI
 - console 215
 - GUI uninstaller 215
 - silent uninstaller 217
 - wizard 107, 170
- UNIX installation
 - distributed 248
- UNIX installation directory structure 61
- upgrade 188
 - prerequisites 189
- upgrade mode 188
- upgrade prerequisites
 - Linux 47
 - UNIX 47
 - Windows 113
- upgrading 192
 - console mode on Linux 72
 - console mode on UNIX 72
 - installation wizard on Linux 68
 - installation wizard on UNIX 68
 - installation wizard on Windows 130
 - migration log file 140
 - modifying installation 81, 140
 - ObjectServer schemas 81, 141
 - response file 75, 136
 - silent mode 78, 139
 - silent mode on Linux 75
 - silent mode on UNIX 75
 - silent mode on Windows 136
 - specifying silent mode settings 75, 136
 - Web GUI
 - from Netcool/Webtop version 1.3 199, 225
 - from Netcool/Webtop version 2.1 194, 225
 - overview 185
- user access security 337
- users
 - default 354
- usersmultiple login to one account 568
- UTF-8 conversion 146
- UTF-8 Windows encoding 403

V

- variables, notation for xiii
- vault key file 210
- verifying
 - Netcool/OMNIBus installation 59
- viewing
 - certificate details 393
 - installation log file 124
 - installation log files 60
 - installed packages 61, 124
 - migration log file 80, 140
- virtualization 447
 - resources 446
- VMM
 - for ObjectServer 485
- vmmusers 29

W

- WAAPI
 - initial setup 212
 - migration prerequisites 191
- Web browsers 14
- Web GUI 173, 175
 - coexistence with IBM Tivoli
 - Netcool/Webtop 28
 - configuration 477
 - console installer 180
 - console uninstaller 215
 - deployment considerations 28
 - features 5
 - from Netcool/Webtop version 2.2 192
 - from Web GUI version 7.3.0 192
 - GUI installer 178
 - GUI uninstaller 215
 - installation prerequisites 174
 - overview 4
 - passwords for supplied users 212
 - silent installer 181
 - silent uninstaller 217
 - user repositories
 - adding 478
 - LDAP authentication 484
 - removing 477
 - switching 479
 - synchronization 29, 480
- Windows 2008 limitations 114
- Windows installation
 - distributed 248
- Windows installation directory
 - structure 125
- Windows services 237
- Windows Vista limitations 114
- wizard upgrade
 - Linux 68
 - UNIX 68
- wizard upgrade mode
 - Windows 130

X

- XML
 - for configuration files 520



Printed in the Republic of Ireland

SC14-7604-00

