Tivoli. software

IBM

IBM Tivoli Netcool/OMNIbus V7.2.1 Features

White Paper

Stephen Meyers (meyerss@us.ibm.com)

November 2008

Table of Contents

Introduction

White Paper

Conclusion

Introduction

About this Paper

This white paper describes features that are new to IBM® Tivoli® Netcool/OMNIbus V7.2.1.

After reading this white paper, you will understand:

- New OMNIbus security enhancements and Federal Information Processing Standard (FIPS) 140-2 compliance
- New Windows Process Automation daemon
- OMNIbus and Tivoli License Compliance Manager
- Event pump for z-Series

OMNIbus V7.2.1 brings enhanced functions to users and delivers significant benefits, specifically in the following areas:

- Improved security with FIPS 140-2 alignment
- Improved process control agent on the Windows® platform, including the ability for agents to run external commands from ObjectServers across mixed UNIX® and Windows environments
- Support for IPv6 on Windows, complementing the support on UNIX platforms
- Integration with Tivoli Service Request Manager, Tivoli Data Warehouse, and Tivoli Common Reporting
- Recognition of the need to process events from IBM System z™ applications and to support Linux® on System z

Audience

The audience for this white paper is anyone with existing knowledge of OMNIbus who wants more information about the features for V7.2.1.

White Paper

1 New OMNIbus 7.2.1 Security Enhancements and Federal Information Standard 140-2 Compliance

In this white paper, the new security enhancements in the OMNIbus V7.2.1 are examined. Included in the paper are items such as:

- Federal Information Standard (FIPS) 140-2 compliance
- IPv6 support
- Secure Socket Layer (SSL) and IKeyman© functions

1.1 Federal Information Processing Standard 140-2 Compliance

Where encryption is used, it must be performed using a FIPS 140-2 certified implementation to disable obfuscation when using fixed algorithms and keys.

FIPS 140-2 is a cryptographic function validation program defining security standards for cryptographic modules used in Information Technology (IT) software. Vendors of IT software that submit bids to U.S. government agencies must meet FIPS 140-2 certification and compliance to qualify for selection. Consequently, IBM Software Group (SWG) offerings with cryptographic function must meet this certification and compliance requirement so that the offerings can be marketed to the federal sector. To achieve this, SWG requires its offerings with cryptographic function to use specific SWG cryptographic modules that have been FIPS 140-2 certified.

1.1.1 Scope of IBM Compliance

All IBM Software Group products and components with cryptographic function are included in the scope.

1.1.1.1 Steps for Adoption

The following steps are required for adoption of new security enhancements:

- All products with cryptographic function must use the Tivoli-provided ICC, JCE, and JSSE modules.
- Components with cryptographic function must use the Tivoli-provided ICC, JCE, and JSSE modules.
- Product-specific cryptographic modules and non-FIPS cryptographic modules are not included in the adoption.

All cryptographic modules used by a SWG product must be FIPS 140-2 tested and certified for the product to meet the FIPS 140-2 requirement. Testing and certification of the modules must be obtained through National Institute of Standards and Technology (NIST) approved, third-party vendors.

To minimize costs associated with certification and maximize the returns on development investments, SWG provides the following strategic FIPS 140 cryptographic providers for SWG products: IBMJCEFIPS, IBMJSSEFIPS, and ICC. The Tivoli team develops these modules and includes them in the following IBM components:

- IBM Java JDK 1.4.x, Java JDK 5.0
- IBM JDK 1.3.1 as an extension provided via the Java Information Manager
- GSKit© version 7

The certificates for FIPS 140-2 certified providers are posted as part of the FIPS140 process by NIST along with the FIPS 140 required security policy documents. The certificates are available at the NIST Web site at http://csrc.nist.gov/cryptval/140-1/1401val2004.htm. Certificates are posted for specific releases of the certified providers.

The following certifications are listed by environment and release number:

1.1.1.2 Java© Environment

Java JDK 1.3.1

- IBMJCEFIPS and IBMJSSEFIPS available to products as extensions to the JDK via the Java Information Manager: IBMJCEFIPS v1.1 Certificate number 376
- IBMJCEFIPS code included with IBM JDK/JRE 1.4.2: IBMJSSEFIPS Certificate number 409

Java JDK 1.4.0 and 1.4.1

- IBMJCEFIPS and IBMJSSEFIPS based elements of the IBM JDK:

  - IBMJCEFIPS v1.1 Certificate number 376
  - IBMJSSEFIPS Certificate number 409

- IBMJCEFIPS code included with IBM JDK/JRE 1.4.2: IBMJSSEFIPS Certificate number 409

  In JDK 1.4.2 a new IBM JSSE provider named IBMJSSE2 was added that uses only IBMJCE/IBMJCEFIPS and does not contain any crypto. Therefore, it does not need a FIPS 140-2 certification. To determine if you have the correct version of the provider, you can use the Java IBMJCEFIPS meta data located in the manifest.mf of the ibmjcefips.jar file.

- IBMJCEFIPS code included with IBM JDK 5.0

  In JDK 5.0, IBMJSSE/IBMJSSE2 uses only IBMJCE/IBMJCEFIPS and does not have any crypto, so IBMJSSE/IBMJSSE2 does not need a FIPS 140-2 certification. To determine if you have the correct version of the provider, you can use the Java IBMJCEFIPS meta data in the manifest.mf of the ibmjcefips.jar file, which should be available in the following versions:

  - For IBM JCE FIPS version 1.1 Implementation: Version 1.1 build_20030930
  - For IBM JCE FIPS version 1.2 Implementation: Version 1.2 build_20040903
  - For IBM JSSE FIPS version 1.1 Implementation: Version 1.0.3

1.1.1.3 C and C++ Environments

The most current certificate issued is 384 on 2/17/2004 and updated on 4/27/2004 and 12/02/2004. The Security policy is at http://csrc.nist.gov. This certificate covers IBM Crypto for C (ICC) v1.1, v1.2, v1.2.1, and v1.2.2. Global Security Kit (GSKit) has included ICC for FIPS from release 6F up to the current release 7C.

1.1.2 FIPS Certificates Used for OMNIbus Compliance

Tivoli Netcool/OMNIbus uses the following FIPS 140-2 approved cryptographic providers for cryptology:

- IBMJCEFIPS (Certificate number 376) or IBMJSSEFIPS (Certificate number 409)
- IBM Crypto for C (ICC) (Certificate number 384)

The implemented certificates are described in the following sections.

1.1.2.1 Certificate IBMJCEFIPS 376

The IBM Java Cryptographic Extension (JCE) FIPS provider (IBMJCEFIPS) for Multi-platforms is a scalable, multi-purpose cryptographic module. It supports only FIPS-approved cryptographic operations using Java2 Application Programming Interfaces (APIs).

The cryptographic module is a software module implemented as a Java archive (JAR). The software module is accessible from Java language programs through an API. Some of the available API functions are listed in the Cryptographic Module Services section. Usage guidelines and details of the full API function set are available in the IBM Java JCE FIPS (IBMJCEFIPS) Cryptographic Module API Javadoc.

1.1.2.2 Certificate IBMJSSEFIPS 409

IBMJSSEFIPS is a Secure Sockets Layer (SSL) interface that provides APIs to users who write SSL applications. IBMJSSEFIPS uses an embedded FIPS-approved module. IBM SSLite in Java (Certificate number 406) is used for the actual SSL interfaces. IBM SSLite in Java is a Transport Layer Security (TLS) V1.0 protocol implementation including Public Key Infrastructure (PKI) functionality for the hand-shake in Java. This implementation, in turn, relies exclusively on the internal FIPS-approved IBM CryptoLite in Java module for all cryptographic functionality. See the IBM SSLite in the Java Security Policy for more information.

1.1.2.3 Certificate IBM Crypto for C (ICC) 384

The IBM Crypto for C (ICC) versions 1.1, 1.2, 1.2.1, and 1.2.2 is a cryptographic module implemented in the C programming language. It is packaged as dynamic (shared) libraries usable by applications written in a language that supports C language linking conventions, such as C, C++, and Java Assembler on commercially available operating systems. The ICC allows these applications to access cryptographic functions using an API that is provided through an ICC import library and based on the API defined by the OpenSSL group. The cryptographic boundary is defined as the enclosure of the workstation that runs the ICC software.

1.1.3 FIPS Configuration in OMNIbus 7.2.1

1.1.3.1 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, you must create the FIPS configuration file as follows:

- Create an empty text file called fips.conf.
- Save this file in the relevant directory for your operating system:
  - UNIX: $NCHOME/etc/security
  - Windows: %NCHOME%\ini\security
- Configure the server components for FIPS 140-2 mode.

1.1.3.2 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 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, the components 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.

Where encryption is used, it must be performed using a FIPS 140-2 certified implementation to disable some of the data obfuscation encryptions when using fixed algorithms and keys. Certain forms of password encryption, like the DES algorithm, are disabled in FIPS mode. This is done to prevent intentional password attacks against fixed algorithm keys.

The current encryption algorithm introduced in OMNIbus 7.2.0 is AES and resides in a client-side file. Password encryption across a network should always be used with the SSL implementation. The workstation and operating system must be configured in a FIPS compliant way. Since the physical cryptographic boundary includes the workstation, the hardware must be physically secure. Then, a single instance of OMNIbus 7.2.1 can be installed within that cryptographic boundary.

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 specify plain text passwords within these files. Or, you can specify passwords that are encrypted by running the $NCHOME/omnibus/bin/nco_aes_crypt utility with a key file and a 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. For the process agent, which does not make use of properties, you must specify command-line options to decrypt the passwords in the configuration file when you run $NCHOME/omnibus/bin/nco_pad. You should not use the nco_g_crypt, nco_pa_crypt, and nco_sql_crypt utilities to encrypt passwords when running in FIPS 140-2 mode.

1.1.3.3 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 encrypt passwords within the proxy server, probe, or gateway properties files, run the nco_aes_crypt utility and use the -c command-line option to specify AES_FIPS as the encryption algorithm.

1.1.3.4 Property Encryption Prior to OMNIbus 7.2.1

The nco_aes_crypt script was officially introduced in 7.2.0 to handle string encryption. The nco_keygen script creates a key file, and then the nco_aes_crypt script encrypts the string values. These values can be used for any string property (not numeric or boolean properties) provided the ConfigCryptoAlg and ConfigKeyFile are set correctly. The properties are decrypted when they are read. Therefore, no protection of value is sent across the network. The key file must be protected by the operating system, or all security is lost. These scripts were introduced before OMNIbus version 7.2.1.

1.1.3.5 Property Encryption in OMNIbus 7.2.1

The new property encryption uses the nco_aes_crypt script and must be used with the -c AES_FIPS extension. Property files that contain properties encrypted with this tool must have ConfigCryptoAlg set to AES_FIPS to match. A non-fatal error will be logged when you start the system if fips.conf exists and the older AES algorithm is used.

Key files generated with a 7.2 version of nco_keygen must be replaced with keys generated with the 7.2.1 version. This is not enforced, but the random number generator in the old tool is not FIPS compliant. Periodically regenerating these keys is a good security practice.

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 used primarily for compatibility with the AES property encryption that is available in Tivoli Netcool/OMNIbus V7.2 and when the use of the AES_FIPS algorithm is preferred.
- To recover property values encrypted with the old tool use the following:
  - nco_aes_crypt -d -c AES -k oldkeyfile oldpropvalue
  - nco_aes_crypt -c AES_FIPS -k newkeyfile outputfromabove

1.1.3.6 Disabled Client Side Encryption

The following criteria apply when you are working with disabled client side encryption:

- You can no longer use nco_crypt to encrypt probe passwords. You must use nco_aes_crypt to protect the password while it is on disk. You must also use SSL to protect the password over the network.
- You can no longer use nco_g_crypt to obfuscate passwords for gateways. You must use nco_aes_crypt to protect the password on disk. You must also use SSL to protect the password over the network.
- You can no longer use nco_sql_crypt to obfuscate passwords when logging into the ObjectServer. You must use SSL to protect the password over the network.
- You can no longer use nco_pa_crypt to obfuscate passwords in the security section of the nco_pa.conf file. You must use nco_aes_crypt to protect the password on disk. The PA has no properties file. Use -cryptalgorithm and -keyfile to specify the algorithm (which must be AES_FIPS) and key file.

1.1.3.7 Disabled Encryption on the Network

The *secure login* technique used to encrypt passwords being sent between clients and servers can no longer be used. Instead use SSL for enhanced security.

1.1.3.8 Server Side ObjectServer Passwords

The ObjectServer PasswordEncryption property must be set to AES. When upgrading, the property must be changed and all stored passwords mustbe re-entered before the ObjectServer is fully compliant. You must 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. If the fips.conf file exists but PasswordEncryption is set to DES, the ObjectServer will issue an error message on start up but will continue to run.

1.1.3.9 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.
- 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.

On Windows, process agent connections are authenticated against the Windows user accounts, and no additional configuration is required for FIPS 140-2 mode.

1.1.3.10 Server Side Gateway Passwords

UNIX passwords cannot be used by the gateways. You must use PAM authentication instead. You need to 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. The user is responsible for ensuring that the PAM module is compliant.

1.1.3.11 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.

1.1.3.12 FIPS and GUI Tools

In FIPS mode the nco and nco_event tool %encrypted_password variable is populated with the plain text password. If the process being executed is nco_event or nco_elct the child will remove the password argument from its command line after starting so that the ps command cannot be used to view the password.

1.1.3.13 Using OMNIbus Without FIPS 140-2 Enabled

Inclusion of the FIPS configuration file is required in the following locations:

- UNIX: $NCHOME/etc/security/fips.conf
- Windows: %NCHOME%\ini\security\fips.conf

Contents of the fips.conf file might be defined in a future version. The ObjectServer, proxy server, and ObjectServer gateways log a debug message on startup if the configuration file is present. The ObjectServer, proxy server, and ObjectServer gateways will operate as normal.

1.1.3.14 Configuration Requirements for Connecting V7.2 or Earlier Clients to V7.2.1 Servers in FIPS 140-2 Mode

Tivoli Netcool/OMNIbus V7.2.1 is compatible with earlier versions of 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 servers running in secure mode. It also shows the configuration changes required for FIPS 140-2 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 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 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 ObjectServer running in secure mode.
SQL interactive interface (nco_sql)	Yes	Clients can connect to a V7.2.1 ObjectServer running in secure mode if the clients 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 ObjectServer running in secure mode without any changes.
Process agent client (nco_pad)	Yes	Process agents can connect to a V7.2.1 ObjectServer running in secure mode without any changes.
Probes	Yes	Probes can connect to a V7.2.1 ObjectServer running in secure mode if the probes 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.
Other clients	-	When the ObjectServer is in FIPS 140-2 mode, clients supplying encrypted passwords cannot connect to the ObjectServer.

1.2 Internet Protocol version 6 (IPv6) Support Inclusion

1.2.1 Internet Protocol version 6 (IPv6) Enablement Introduction

IPv6 is an industry standard for the IP network. It removes the IP addressing limitation of IPv4 and provides better quality of service and more IP security. It is also a key requirement for government projects. The U.S. Federal Government has mandated that all software products considered for U.S. government projects support IPv6 by June 30, 2008. To comply with this requirement, a product must work with full function parity over IPv6 on the supported platforms. Additionally, the product must provide *dual-stack* support. This means that the product must be able to perform the following tasks:

- Communicate using both IPv4 and IPv6 protocols simultaneously with other applications using IPv4 or IPv6
- Support both IPv4 and IPv6
- Select the proper protocol (IPv4 or IPv6) at run time on a case-by-case basis depending on whether the remote peer also supports IPv6, has IPv6 connectivity, and has advertised a desire to use IPv6 via the DNS

1.2.2 IPv6 Implementation in OMNIbus V7.2.1

1.2.2.1 Availability

IPv6 is already available on all UNIX platforms. With the instantiation of OMNIbus V7.2.1, Windows XP/Server 2003 and Windows Vista/Server 2008 are also supported. IPv6 is supported on all OMNIbus products (Administrator, desktop, and so on) with the exception of current probes. which will be addressed in a subsequent section of this white paper. The implementation is achieved by entering an IPv6 address where previously only IPv4 addresses were allowed. The implementation should be mostly transparent to the end user. On Windows, the Tivoli Netcool/OMNIbus components can now also operate and coexist on a network that supports IPv4 only, IPv6 only, or a dual IPv4 and IPv6 configuration.The current Windows implementation has some additional limitations. Windows XP/Server 2003 and Windows Vista/Server 2008 are implemented in different ways. Windows XP/Server 2003 have separate IPv4 and IPv6 protocol stacks while Windows Vista/ Server 2008 have true dual IP protocol stack support similar to Unix-based systems. This should make no difference to users, but it might require alternate troubleshooting methodologies.

1.2.2.2 Dual Stack Socket Support in OMNIbus

If you use a hostname in the Servers Editor, it will take the first address that the name resolves to and create a listening socket to that IP version. If the hostname resolves to an

IPv4 address, then it cannot connect using pure IPv6. You must use a different IPv6 hostname and then the IPv4 hostname or enter an IPv6 address in the Servers Editor. It is possible to have an IPv6 address entry and an IPv4 address entry for the same server name in the Servers Editor.

If you have an IDUC listening port listening on all interfaces, and a client connects to a hostname (also known as a callback), the callback can define the address it uses. You can define that callback address in the Iduc.ListeningHostname property of ObjectServer.

If you use the netstat command, then you will see that Windows XP/Server 2003 have one instance listening on an interface for the IPv4 stack and one instance listening on an interface for the IPv6 stack. Both stacks use the same port.

You can use the nco_pad to force the IPv6 by identifying the subnet with the slash 64 address as follows: ::1234:2000:3004/64. This slash 64 subnet address format can be used in the nco_security section for the subnet definition.

1.2.2.3 Configuring IPv6 Support

1.2.2.3.1 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 named $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 of 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.

From a command line, enter the following command:

```
$NCHOME/bin/nco_igen -all
```

This command 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 or interfaces.solaris2. You can copy the relevant operating system interfaces file to the $NCHOME/etc directory on each of the host computers.

From the Server Editor, you can specify your communications settings and then select the Generate All check box. Clicking the Apply button generates an interfaces file named $NCHOME/etc/interfaces.arch, where arch represents individual UNIX operating system

names. You can copy the relevant operating system interfaces file to the $NCHOME/etc directory on each of the host computers.

On UNIX, the connections data file, $NCHOME/etc/omni.dat, is used to create the interfaces file for Tivoli Netcool/OMNIbus communications. The following examples are for IPv4 and IPv6 settings within this file.

Example 1: Configuring the omni.dat file with a host name and an IPv6 address

```
[NCOMS]
{
        Primary:        presley 9000
}
[BARROW]
{
        Primary:
fec0:0000:0000:7777:0218:fcef:fe8c:4f3b 8002
}
```

Example 2: 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 omni.dat file from Example 1, the IPv4 address of the dual stack IPv4 and IPv6 ObjectServer is 192.168.0.1, and its IPv6 address is 2094:82a:2a6e:123:503:badd:fe43:f552.

```
[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.

1.2.2.3.2 Windows Configuration

On each Windows computer, you can 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.

1.2.2.4 Using HTTP or FTP to Load Remote Event List Configurations 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 an .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 and all subsequent releases. Therefore, you must ensure that one 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.

A good practice is to verify whether the event list can load the .elc file. To do this, you can open a Web browser and enter the IPv6 format of the URL that points to the .elc file to see whether you can access the file.

1.2.2.5 Windows-Specific Issues

On Windows XP/Server 2003 the IPv6 protocol must be installed and a valid external IPv6 interface must be configured. On a Windows Vista/Server 2008 only a valid external IPv6 interface must be configured because the IPv6 protocol is installed by default. If it was not installed by default, then only ::1 will work on the local workstation. The same issue occurs on UNIX operating systems.You can address this issue by using the appropriate IPv6 DNS entries with a domain name that resolves to only the IPv6 address.

1.2.2.6 Additional Information about IPv6 Configuration

- Do not include square brackets [ ] in the IP address unless you are using the brackets in probe rule include files such as http://[::1]:8080/foo.rules.
- Use IPv4 addresses with IPv6 notation, for example, ::ffff:127.0.0.1 (which is the same as 127.0.0.1).
- ::1 is the IPv6 loopback and will work only on the local workstation.
- **::** is the network address (IPv4 equivalent 0.0.0.0). This should work in the Server Editor, but should be avoided.
- fe80::/10 is the local link and should not be used; it will fail on Windows.
- fec0::/10 is the local site and should not be used; it will fail on Windows.
- ff00::/10 is multicast and should be avoided; it will fail on Windows.

1.2.2.7 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"
```

1.2.2.8 Windows Vista and Windows Server 2008 Concerns

1.2.2.8.1 Administrator Privileges

Security has been improved on Windows Vista/Server 2008, but the security is more complex. It now uses the Windows User Account Control (UAC) system. Adding a user to the Administrators group is not the same as adding a user to the Administrators group on a Windows XP/Server 2003. With Windows Vista/Server 2008, you must use the UAC system to add a user to an administrative group.

Problems with installing into the Program Files directory can occur with Windows Vista/ Server 2008 because of the enhanced UAC authentication. It is recommended to always install and run Windows Vista/ Server 2008 as the same user. During an installation, each administrative user is assigned administrative ownership of configuration, properties, and database files. Be careful not to install from a user account that is not part of the Administrator group. Installation performed by a nonadministrative user will install into a user-specific directory.

1.2.2.8.2 Windows User Account Control

During an installation, Windows UAC causes windows to pop up asking for confirmation during the installation into the Program Files directory. Pop-ups also occur when configuration files are edited. If an application does not trigger the UAC, then the application will fail when writing the file. Consider performing some of the following workarounds before you install the IPv6 protocol:

- Do not install into the default program files directory.
- Disable the UAC and revert to the hidden *root* Administrator if your security policy allows. You can accomplish this by using the following process:
  - Right-click the Computer icon on the Start menu and select Manage from the context menu. When the UAC is displayed, click OK.
  - In the Computer Management console, go to the navigation pane and click the arrow next to Local Users And Groups. Click the Users folder and select the Administrator Account.

– Click More Actions under Administrator in the Actions panel and select Properties from the context menu. Under the Administrator Properties dialog box, clear the Account Is Disabled check box and click OK.

1.3 Secure Socket Layer Functionality and the IBM Key Management Application

1.3.1 Introduction

Using SSL is important for the following reasons:

- Ensure data secrecy, integrity, and authentication
- Prevent eavesdropping on sensitive data such as:
  - Usernames and passwords
  - Sensitive network information
- Guarantee information is not tampered with in transit
- Guarantee information is going to authorized server

Using SSL in OMNIbus V7.2.1 is important for the following reasons:

- Certificates are tied to server names.
- Certificates and keys are now stored in CMS key stores.
- FIPS 140-2 certified modules are used.
- IBM/SWG configuration tools are available.

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:

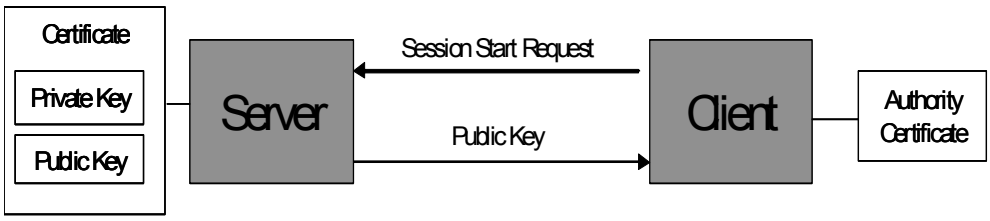
- Create entries for SSL connections in the Server Editor.
- If you are running in FIPS 140-2 mode, configure cryptographic properties for encryption and key generation.
- Create and distribute certificates and keys to servers and clients by using the IBM Key Management (iKeyman) utility, which is used for certificate management.

1.3.2 SSL Concepts

1.3.2.1 Certificates

The three main components in any SSL connection are the client, server, and Certificate Authority (CA).

The Certificate Authority signs its own certificate and should be a trusted party. The Certificate Authority signs the server's certificate with its private key. The Certificate Authority then provides the client with its public key, as illustrated in the following diagram:

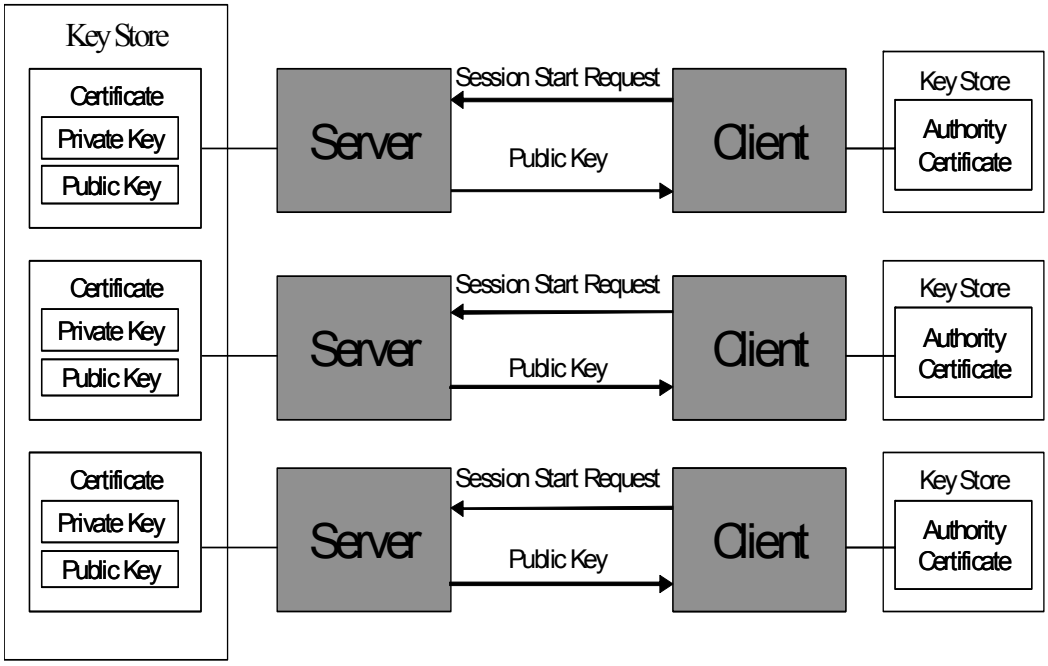


When a connection is established, the server sends its signed public key to the client. The client checks this public key against the authority certificate to verify that the server is

trusted. Session key exchange can now take place using the server public key as a basis for secrecy.

1.3.2.2 Key Stores

Server and trusted certificates are now stored in CMS keystores as opposed to flat text files. These are stored as single files representing a password protected key database. Keys are then referenced by labels. Labels then match ObjectServer or Process Agent names in the omni.dat/sql.ini file. Key Stores are now integral to generating and using certificates in OMNIbus v7.2.1, as illustrated in the following diagram:



1.3.3 Configuring SSL Communications

1.3.3.1 Creating Entries for SSL Connections in the UNIX Server Editor

In the UNIX Server Editor, you can enable encrypted connections, unencrypted connections, or both.

1.3.3.1.1 Configuring SSL On the Server Host Computer

- You can set the unencrypted port and leave the SSL port unset (or set it to 0), but then only unencrypted connections are allowed.
- You can set the SSL port and un-set the unencrypted port (or set it to 0), but then 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.

  If the server allows both encrypted and unencrypted connections, clients that use the same interfaces file as the server (including local clients) will connect using the unencrypted port. If you want to use SSL to connect to these computers, do not specify an unencrypted port for the server.

Generating the Interface File: 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.

When adding an encrypted port for Unix, add “ssl” to omni.dat. This configuration applies to objectservers, proxy servers, and process agents.

Without SSL encryption:

[NCOMS]

```
{
Primary: omnihost 4100
}
```

With SSL encryption:

[NCOMS]

```
{
Primary: omnihost ssl 4100
}
```

1.3.3.1.2 Configuring SSL On Each Client Computer

If you want the client to connect to the server from a 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 a computer 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.

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.

1.3.3.2 Creating entries for SSL connections in the Windows Server Editor

In the Windows Server Editor, you can enable encrypted connections, unencrypted connections, or both.

1.3.3.2.1 Configuring SSL 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 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 clearing the SSL check box.

When adding an encrypted port for Windows, add “ssl” to omni.dat. This configuration applies to objectservers, proxy servers, and process agents.

Without SSL encryption:

[NCOMS]

```
master=NLWNSCK,OMNIHOST,4100
query=NLWNSCK, OMNIHOST,4100
```

With SSL encryption:

[NCOMS]

```
master=NLWNSCK, OMNIHOST,4100,ssl
query=NLWNSCK, OMNIHOST,4100,ssl
```


1.3.3.2.2 Configuring SSL On Each Client Computer

If you want the client to connect to the server from a computer without using encryption, clear the SSL check box. If you want the client to connect to the server from a computer using encryption, select the SSL check box.

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.

1.3.3.3 FIPS 140-2 Mode Configuration (If Implemented) and SSL Communications

If FIPS 140-2 mode is used, perform the following setup before other configuration steps:

- Ensure the $NCHOME/etc/security/fips.conf file is present so that you can put the system into FIPS mode.
- Consider strong encryption enablement in the JRE information found at the following URL:

  http://www-106.ibm.com/developerworks/java/jdk/security/

- Update Java configuration files as follows:
  - Use $NCHOME/platform/arch/jre_1.5.6/jre/lib/security/java.security to configure security providers and SSL socket settings.
  - Use $NCHOME/platform/arch/classes/ikminit.properties to enable FIPS mode for the key store management utilities.
- Regenerate certificates that were generated using nonapproved crypto modules (nco_ssladmin). These certifications should not be migrated in an upgrade scenario. They must be regenerated.

To set up SSL connections in FIPS 140-2 mode, you must enable the use of FIPS 140-2 certified cryptography by configuring the relevant iKeyman properties. Set these properties before creating the key database.

To enable FIPS 140-2 mode, perform the following tasks:

- Open the following properties file for editing: $NCHOME/platform/arch/classes/ikminit.properties (where architecture represents your operating system directory)
- Uncomment the following properties:
  DEFAULT_FIPS_MODE_PROCESSING=ON
  DEFAULT_CRYPTOGRAPHIC_BASE_LIBRARY=ICC
- Save and close the file.

1.3.3.4 Managing Digital Certificates for SSL Communication and the IBM Key Management Application

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 to protect private keys that 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.

1.3.3.4.1 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.

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

File extension	Description	Tivoli Netcool/ OMNIbus file name
.sth	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. 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. Tivoli Netcool/OMNIbus requires a stash file.	omni.sth

The key database files are stored in the following location:

- UNIX: $NCHOME/etc/security/keys
- Windows: %NCHOME%\ini\security\keys

1.3.3.4.2 Guidelines for Certificate Management

To set up SSL connections between your clients and servers, you must have 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:

- 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.
- 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.
- 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.
- 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.

- 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.
- 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.
- 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.

1.3.3.4.3 Starting iKeyman

You can perform most of your certificate management tasks from the iKeyman GUI. To start iKeyman, perform the tasks that are relevant for your operating system, as follows:

UNIX: From a command line, enter the following command:

```
$NCHOME/bin/nc_ikeyman
```

Windows: Perform one of the following actions:

- From the command line, enter the following command:

  ```
  %NCHOME%\bin\nc_ikeyman.bat
  ```

  Use this command 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 nc_ikeyman.vbs file.

1.3.3.4.4 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.

To create a key database, perform the following tasks:

1. Start iKeyman.

2. From the IBM Key Management window, click Key Database File > New. The New window opens.

3. Perform the following tasks:

   – **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**: Verify the 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\

   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\.

   On UNIX, accept the default directory. On Windows, ensure that the location is %NCHOME%\ini\security\keys\.

4. Click the OK 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.

5. Complete the following tasks:

   – **Password field**: Type a password. As you type the characters, an indication of the password strength is given. 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 field**: Retype the password.

   – **Set expiration time field**: 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.

6. Click the OK 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.

The IBM Key Management window now shows the file location and name of the key database, and the default signer certificates. 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.

2 New Windows Process Automation Daemon

2.1 Introduction

Topics examined in this section include the newly incorporated Microsoft Windows Process Automation (PA) Daemon and related functions. In OMNIbus version 7.2.0 and previous versions, Process Agents (PA) were only used for external automations. In OMNIbus version 7.2.1 the UNIX PA was ported to Windows.

The configuration is similar to the UNIX implementation. OMNIbus V7.2.1 is managed by the Netcool/OMNIbus Administrator, which is accessible through the GUI. The GUI can be managed remotely.

You can use the Process Control feature in UNIX to install backend processes such as the ObjectServer.You can also use the feature as an alternative to Win32 Services. New UNIX to-and-from Windows interplatform connectivity has been incorporated so that communications can occur between the ObjectServer and Process Agents and between Process Agents themselves. New levels of improved security have also been introduced.

2.2 Installation and Post Installation Activities

All installation processes for OMNIbus V7.2.1 are similar to previous versions. However, post-installation activities for Win32 service installations now include the following activities:

- Recommended system reboot
- Manual installation as a Win32 service

2.2.1 Windows Installation File Differences

In the new Windows installation, the differences that exist between the previous versions of OMNIbus and OMNIbus V7.2.1 are as follows:

- The following executable codes are located in the %NCHOME%\omnibus\bin directory.
  - nco_pad.exe replaces nco_pa.exe
  - nco_pad.dll replacesnco_pa.dll
  - nco_pa_addentry.exe
  - nco_pa_status.exe
  - nco_pa.shutdown.exe
  - nco_pa_start.exe
  - nco._pa_stop.exe
  - The executable file, libPa.1.dll, is located in %NCHOME%\platform\win32\bin.
- The configuration file, nco_pa.conf replaces nco_pa.props, is located in the %NCHOME%\omnibus\etc directory.

The Windows file names now resemble their UNIX counterparts.

2.2.2 Post Installation: Windows PA Modes

The Windows PA can run in two modes: Win32 Service and Console Application.

- **Win32 Service**: In a production environment it is recommended to run the PA as a Win32 service that starts up automatically when the system reboots. You can install and run other Netcool components, such as ObjectServer, Probes, and Gateways as Win32 services also.
- **Console Application**: In Console mode you are initiating activities from the command prompt. Using console mode is recommended when you are troubleshooting or developing and testing in a lab environment.

2.2.2.1 Console Mode Used for PA Configuration

All of the runtime arguments for PAs are set using the command line. There is no .props file. To set any non-default parameters, use the console mode to set PA configurations.

2.2.2.1.1 Running the PA in Console Mode

Open a command prompt from the Start menu using the following command:

```
> cd %OMNIHOME%\bin
> nco_pad –help
```

The nco_pad command with the -help option lists all the available arguments to the nco_pad command.

The following nco_pad command with the /? option lists all of the available Win32 service installation options:

```
> nco_pad /?
```

When run from the command line, all of the windows of child processes are visible. You will see the ObjectServer and Probes running as Console applications.

2.2.2.1.2 Command-line Runtime Arguments: Default Settings

The following default runtime settings are used if no command line arguments are used when launching in console mode. The Authentication System displays the only valid authentication option currently available for Windows.

```
Name of server              : NCO_PA
Log file                    : %OMNIHOME%\log\NCO_PA.log
Configuration File          : %OMNIHOME%\etc\nco_pa.conf
Child Output File           : Unsupported
Maximum logfile size        : 1024
Start Auto-start services   : True
Authentication System       : Windows standard logon
Host DNS name               : <local computer>
Secure Mode                 : False
```

2.2.3 Manual Installation of PA as a Win32 Service

The manual setup using a command line is the same process for all Netcool Win32 services.

```
> cd %OMNIHOME%\bin
> nco_pad /install /cmdline "<runtime arguments here>"
```

To see other options, use the /? option as follows:

```
> nco_pad /?
```


To uninstall the service again, make sure that the service has stopped first by issuing the following command:

```
> nco_pad /remove
```

By default the PA installs the user account as LocalSystem. You can use another account that belongs to the Administrators group on the local workstation. Pay careful attention to password expiration.

Use the following default Win32 service attributes:

- **Name:** NCO Process Agent
- **Startup Type:** Automatic
- **Account:** LocalSystem
- **Command line:** Empty
- **Working directory**: %SystemRoot%\system32
  This location might change in the future because of security concerns. As a workaround, use full path names on log files and other fully qualified paths.

2.2.4 Configuring PA for ObjectServer External Automations

The configuration of the PA for ObjectServer external automations is similar to other PA configurations with the exception of the following ObjectServer parameters:

- **PA.Name**: 'NCO_PA'
- **PA.Username**: '<windows account>'
  You must now supply a valid administrative/Windows account.
- **PA.Password**: '<password>'
  You must now supply a valid password associated with the Username account.

2.2.5 Configuring PA for Process Control

The PA configuration file defines the processes to be managed. You can use the Process Agents command-line argument, -configfile <filepath>, to modify the configuration. The default path is %OMNIHOME%\etc\nco_pa.conf. The structure and syntax used is the same as UNIX.

When you create a PA configuration file using a GUI, consider the following issues:

- You can launch the Administrator.
- The PA must be up and running.
- You can add, delete, update processes, grouped into services.
- You can overwrite the configuration file that the PA started with.

If you want to run the PA configuration file on external implementations, you can run the PA with the –noconfig argument. Then, the default nco_pa.conf will be overwritten.

You can edit the configuration file manually with a text editor if you need to set up before starting the PA.

2.2.5.1 Process Agent Configuration File Details

In the following example, the only value to change would be the Host identifier to the Windows hostname. By doing so, the process agent would parse the UNIX command parameter to a Windows recognizable command.

```
#
#List of processes
#
nco_process 'MasterObjectServer'
{
Command '$NCHOME/omnibus/bin/nco_objserv -name NCOMS -pa
NCO_PA' run as 0
Host             =        'omnihost'
Managed          =        True
RestartMsg       =      '${NAME} running as ${EUID} has been
restored on ${HOST}.'
AlertMsg         =      '${NAME} running as ${EUID} has died
on ${HOST}.'
RetryCount       =        0
ProcessType      =        PaPA_AWARE
}
```

In the nco_services core section, note that the ServiceStart value has changed from Auto to Non-Auto. This is a change in the default configuration file.

In the routing table section, the values that relate to UNIX-based group authentication are no longer valid using Windows. The nco_routing hostname mapping to Process Agent identifiers uses the Windows-based username and password. This PA would access another PA, so you must be careful about password validity.

2.3 Command Syntax Recognition

The new Windows PA supports both UNIX and Windows syntax, including the following syntax for file paths:

- UNIX style forward slashes (/)
- WINDOWS standard backslash (\)
- Double backslash (\\)

In Windows PA, spaces in file path names are supported.

The following Windows environment variables are supported:

- $NCHOME (UNIX syntax is supported.)
- %NCHOME% (WINDOWS syntax)

UNIX will not support Windows syntax parameters.

The following examples are Windows PA supported syntax:

- '$OMNIHOME/bin/nco_objserv …'
- '$OMNIHOME\bin\nco_objserv …'
- 'C:\\Program Files\\IBM\\Tivoli\\Netcool\\omnibus\\bin\\nco_objserv …'
- '%OMNIHOME%\bin\nco_objserv …'
- 'C:/Program Files/IBM/Tivoli/Netcool/omnibus/bin/nco_objserv …'

2.4 PA-Aware on Windows

The following Netcool components are PA-Aware on Windows:

- ObjectServer (nco_objserv)
- ProxyServer (nco_proxyserv)
- ObjectServer Gateways (nco_g_objserv_bi, nco_g_objserv_uni)

The following Netcool components are not PA-Aware on Windows:

- Probes
- Gateways

2.5 PA Utilities on Windows

The following PA utilities have also been ported from UNIX to Windows:

- nco_pa_status
- nco_pa_shutdown
- nco_pa_addentry
- nco_pa_start
- nco_pa_stop

You can use these utilities from the command prompt or in a script. To log in, you must use the –user and –password options with any valid Windows account.

Currently, no concept similar to ncoadmin user group is in place to restrict access to the PA on Windows. All valid users can log in.

2.6 UNIX-based Items Not Supported on Windows

The following process attributes will cause warnings to be logged to the log file, but the commands will be executed.

- SETGID (Ignored by Windows PA if present in the process definition)
- UMASK (Ignored if present in the process definition)
- ‘run as’

  The ‘run as’ attribute is still required in the PA. Otherwise, the parser will return an error for that process. The process will run with the same account as the PA. In this case, Windows will span a child process.

Other command-line arguments not supported on Windows are listed in the following table:

Command-line argument	Description	Comment
-admingroup	Specifies the name of the UNIX user group that has administrator privileges. Members of this group can access the process control system. The default group is ncoadmin.	If you have a valid account to log into the workstation on which PA is running, and you have access to the Administrator of the PA utilities, you can log into the PA.

Command-line argument	Description	Comment
-killprocessgroup	On UNIX, specifies that when the process agent daemon stops a process, it must also send a signal to kill any processes in the same operating system process group.	A mechanism to kill the children of a process is unavailable at this time.
-nodaemon	This argument is used on UNIX to instruct the PA not to detach from its parent and run as a daemon process in the background.	This argument is not required. The daemon function is achieved on Windows by installing the PA as a Windows service.
-pidfile	Pidfiles are used on UNIX to restrict only one PA daemon per host.	No pidfile locking is required on Windows. There is no restriction on the number of PA Windows services per host.
-pidmsgpool	This argument specifies the size of the signal-handling message pool. This pool is used in processing the SIGCHLD signals.	Killing child processes on Windows is not implemented with SIGCHLD signals.
-ticketdir	The directory for Kerberos tickets if -authenticate is set to KERBEROS.	No support for KERBEROS is available on Windows. The PA only authenticates against the Windows account.
-recirectfile	On UNIX this argument allows users to see the stdout/stderr output of the child processes being launched by the PA. It is used for troubleshooting.	

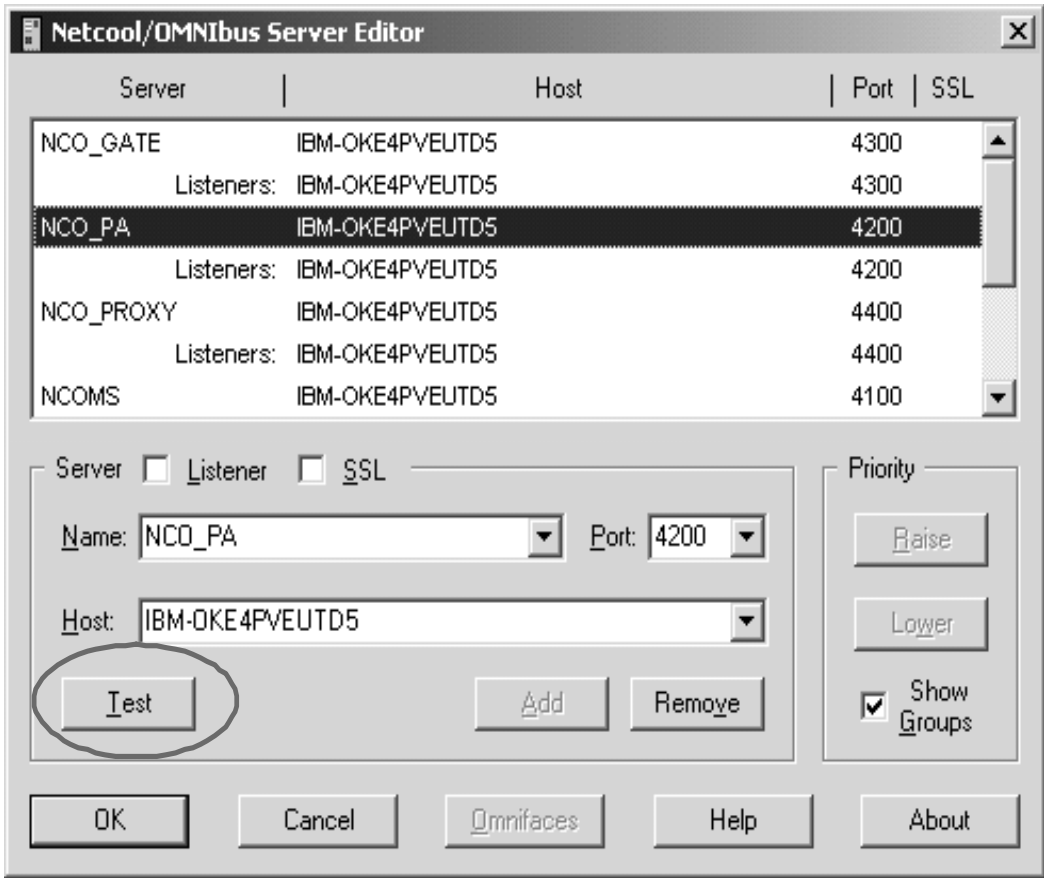
2.7 Connection Detail Configuration

The connection detail configuration for OMNIbus V7.2.1 is the same as the configuration in OMNIbus v7.2.0. The port number default is 4200. To configure the connection, navigate to the Servers Editor from a Windows workstation as follows: Start Menu > All Programs >Netcool Suite > System Utilities > Servers Editor

Use the Servers Editor to save the port information in the %NCHOME%\ini\sql.ini file.

2.8 Servers Editor New TEST Button Functionality

OMNIbus 7.2.1 has the ability to use the TEST button in the Servers Editor to ping a Process Agent.



3 OMNIbus and the IBM Tivoli License Compliance Manager

This section describes OMNIbus version 7.2.1 and the IBM Tivoli License Compliance Manager.

The IBM Tivoli License Compliance Manager detects the installation and use of all IBM distributed software currently in support. The ITLM process ensures OMNIbus V7.2.1 maintains its compliance.

Tivoli Netcool/OMNIbus V7.2.1 does not require a license key to run. However, some probes and gateways that have not been through a recent maintenance cycle still require

license keys. If you are running these older probe or gateway packages, they still require the NETCOOL_LICENSE_FILE environment variable to be set, and the availability of a Netcool license server.

If preferred, you can set this environment variable when setting the other environment variables for your Tivoli Netcool/OMNIbus V7.2.1 installation. Full details about setting this environment variable are provided in the *Netcool/OMNIbus V7.1 Installation and Deployment Guide*.

On UNIX, if you need to install the process agent startup scripts that cause process agents to automatically start on reboot, you are prompted to set the NETCOOL_LICENSE_FILE environment variable because some older probes and gateways still require this setting. You can either press Enter to accept the default value of 27000@localhost or specify another value that reflects the settings for your configured license server. Ensure that you download the latest version of probe and gateway components for use with V7.2.1. Do not use the V7 probe bundle PINSTALL function.

Netcool/Webtop version 2.2 does not require a license key to run. Netcool/Webtop is compatible with IBM Tivoli License Compliance Manager. To maintain compatibility with IBM Tivoli License Compliance Manager, Netcool/Webtop creates one or more signature files with the .SYS2 file extension. You must not delete any signature files, or IBM Tivoli License Compliance Manager will not be able to detect Netcool/Webtop.

3.1 Hardware Requirements

The following minimal hardware requirements are for IBM Tivoli License Compliance Manager V2.3 administration and runtime servers (one server only without the database). The available disk space required for the servers is listed in the Software requirements section.

- **Microsoft Windows and Linux on xSeries**: Intel® Pentium® IV 1.5 GHz with 1 GB RAM
- **AIX and Linux on pSeries**: RS/6000 Model 7044/270, two processor IBM RS/6000 375, 1 GB RAM
- **Solaris**: 2000 model, two Ultrasparc III processors, 1 GB RAM
- **HP**: rp2470 two PA RISC 650 MHz processors, 1 GB RAM
- **Linux on zSeries**: 1 dedicated processor with 1 GB RAM
- **Linux on iSeries**: 1 GB RAM

For the agent, the program requires 180 MB available disk space (30 MB for code and data and 150 MB for the agent) and 190 MB available disk space for logs in the Tivoli common directory.

The memory requirements for the IBM Tivoli License Compliance Manager V2.3 are:

- Administration server only: 1 GB RAM
- Runtime server only: 1 GB RAM
- A server and its database on a single computer: 1.5 GB RAM
- Both servers and their databases on a single computer: 3 GB RAM

3.2 Software Space Requirements

On Windows platforms, a minimum of 100 MB of available temporary space must be available on the default drive. On UNIX, at least 250 MB of free space must be available in the /tmp directory.

Refer to the appropriate documentation for DB2® and WebSphere Application Server for details about the space requirements for these products.

The following software prerequisites are for IBM Tivoli License Compliance Manager V2.3 components:

- Administration and runtime servers require IBM WebSphere Application Server Advanced Edition V6.0.2.5 or IBM WebSphere Application Server V6.1.
- Database for administration and runtime servers require IBM DB2 UDB Enterprise Edition V8.2 (fresh install).
- Web interface requires Internet Explorer V6.x or later on Windows, or Mozilla 1.4, or 1.5, 1.6 or 1.7 on Solaris, HP-UX, SLES for IA32, RHEL for IA32.

The server hardware and installation support for IBM Tivoli License Compliance Manager for IBM Software V2.3 is a customer responsibility. Software maintenance is included with licenses purchased through Passport Advantage and Passport Advantage Express. Installation and technical support is provided by the Software Maintenance offering of the IBM International Passport Advantage Agreement. This service enhances productivity with voice and electronic access into IBM support organizations.

IBM Tivoli License Compliance Manager V2.3 is distributed with an International Program License Agreement (Z125-3301), a License Information document, and CD-ROMs. Publications include information about security, auditability, and control. IBM Tivoli License Compliance Manager uses the security and auditability features of the operating system software. The customer is responsible for evaluation, selection, and

implementation of security features, administrative procedures, and appropriate controls in application systems and communication facilities.

3.3 The Event Pump for zSeries

This section describes the process of event propagation for the zSeries platform. The zSeries event pump is a new event source for OMNIbus. The pump is an emitter of Event Integration Facility (EIF) events. Integration to OMNIbus is accomplished with the EIF probe. The Z Data Pump ships with rules files for the EIF probe and provides new and modified automations for the ObjectServer. Two of the key differences with Omnibus 7.2.1 on zLinux are a new installer that is based on COI and no native event list.

3.4 Overview of the Deployment Engine

The Deployment Engine (DE) is an IBM service component that is packaged with the Linux on System z installer and installed as part of the Tivoli Netcool/OMNIbus installation. The Deployment Engine records files, components, and packages that are installed, and maintains a database of the installation transactions for a user or workstation.

You can install Tivoli Netcool/OMNIbus as either a root or non-root user. If you run the installer as the root user, the Deployment Engine database is created by default in the /var/ibm/common directory and is globally accessible to subsequent installers for related IBM products. The benefits of running the installer as the root user are as follows:

- Only a single instance of the Deployment Engine is required, and it can then be a shared resource for other related IBM products that are installed on the same computer. Updated versions of the Deployment Engine are installed automatically with new product versions and patches, but this does not affect the operation of any existing installations.
- The Deployment Engine can provide a system-wide audit and management of the installed products.

If you run the installer as a non-root user, the Deployment Engine database is created by default in your home directory. If Tivoli Netcool/OMNIbus is installed on the same computer by multiple local users, multiple instances of Deployment Engine databases and files can exist on that computer.

The preferred option is for the installer to be run by the root user or an identified non-root system 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, be aware also that some product installations can be performed only by a non-root user.

3.5 Linux on System z installation requisites

If you are installing Tivoli Netcool/OMNIbus on Linux on System z, you must take note of the following requisites:

- Root access to the system, or root privileges, are not required for the installation process. 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.

3.6 Additional requirements for Linux on System z

If you are installing Tivoli Netcool/OMNIbus on Linux on System z operating systems, the following two operating system packages must be available on your system before the installation: libX11-devel and libXtst.

3.7 Configuring the JRE for FIPS 140-2 mode on Linux on System z

You must configure the Tivoli Netcool/OMNIbus JRE for FIPS 140-2 by making some configuration changes to the security properties file. You can also download and add policy files to use enhanced encryption algorithms.

3.7.1 Configuration file changes

Make the following configuration changes to the security properties file:

1. Open the **$NCHOME/platform/arch/jre_1.5.6/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 preference orders section, ensure that all of the following providers are included for your operating system:

   – security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
   – security.provider.2=com.ibm.jsse2.IBMJSSEProvider2
   – security.provider.3=com.ibm.spi.IBMCMSProvider
   – 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

3. Set the default key and trust manager factory algorithms for the javax.net.ssl package:

   – ssl.KeyManagerFactory.algorithm=IbmX509
   – ssl.TrustManagerFactory.algorithm=IbmX509

4. 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

5. Save and close the file.

3.7.2 Enhanced encryption algorithms

To enable strong encryption, perform the following tasks:

1. Open a browser, and go to the developerWorks Java Technology Security Web page at **http://www-106.ibm.com/developerworks/java/jdk/security/**.

2. Click the J2SE 1.4.2 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. The Un-restricted JCE policy files for SDK 1.4 page is displayed.

4. If you 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, enter your IBM ID and password.
6. Select Un-restricted JCE Policy files for SDK 1.4.2 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 unrestrict142.zip file.
10. Extract the local_policy.jar and US_export_policy.jar files from the unrestrict142.zip archive.
11. Save these two files to the $NCHOME/platform/arch/jre_1.5.6/jre/lib/security directory, replacing the existing files of the same names.
12. Update the policy files on each workstation, and optionally run tests.

Conclusion

Summary

IBM Tivoli Netcool/OMNIbus V7.2.1 has new functions that deliver benefits to users, specifically in the following areas:

- Improved security with FIPS 140-2 alignment
- Improved process control agent on the Windows platform, including the ability for agents to run external commands from ObjectServers across mixed UNIX and Windows environments
- Support for IPv6 on Windows, complementing the support on UNIX platforms
- Integration with Tivoli Service Request Manager, Tivoli Data Warehouse, and Tivoli Common Reporting
- Recognition of the need to process events from IBM System z applications and supports Linux on System z

This white paper described how these new features have been implemented.

Resources

Information contained in this document was either copied in its entirety, reworded, or summarized in whole or in part, or derived from interpretations of presentations, online information, and public and private documents accessible through electronic medias. Sources include internal IBM Web sites, both secure and non-secure, internal IBM controlled documentation, stored event playback media, product image builds, and public Web sites and resources. The creator of this document is not liable or held accountable for any omissions or errors related to the content of this document or the reference to copyrighted or trademarked items.

OMNIbus Product Development Team

- Netcool/OMNIbus 7.2.1 and Netcool/Webtop 2.2 Release Content Specification (RSC) Document. Document Owner Don Wildman. Document Location:

Server:SSGTECH3/SSD/SNJ Filename: YBQA\ NWM Project Repository 2008.nsf. notes://ssgtech3.tucson.ibm.com/YBQA/NWM%20Project%20Repository1%202008.nsf?OpenDatabase

- Development Team's TTEC-GO live presentation playback stored October 23, 2008, IBM-Tivoli-Netcool-OMNIbus-7.2.1-Session-1-of-2-(PNN846033)_en_US_08-07-08_14.41, located internally at http://ausgsa.ibm.com/projects/t/ttec/public/Network%20Management%20Proviso/2008_07_08-07_10_GO_IBM_Tivoli_Netcool_OMNIbus_v7.2.1/
- Development Team's TTEC-GO live presentation playback stored October 23, 2008, IBM-Tivoli-Netcool-OMNIbus-7.2.1-Session-1-of-2-(PNN846033)_en_US_08-07-08_14.41 PowerPoint Slide Deck located at http://ausgsa.ibm.com/projects/t/ttec/public/Network%20Management%20Proviso/2008_07_08-07_10_GO_IBM_Tivoli_Netcool_OMNIbus_v7.2.1/omnibus721-tecgo-final_03.ppt
- Development Team's internal OMNIbus 7.2.1 image located at /hurgsa/projects/n/netcool_omnibus/Dingo_03.

Internal IBM Web sites

- http://w3-03.ibm.com/software/developers/page/348
- http://w3-03.ibm.com/software/developers/page/360
- http://w3-03.ibm.com/software/developers/page/361
- http://w3-03.ibm.com/software/developers/page/362
- http://w3-03.ibm.com/software/developers/page/363
- http://w3-03.ibm.com/software/developers/page/1131
- http://w3-03.ibm.com/software/developers/page/1809
- http://w3-03.ibm.com/software/developers/page/1136
- http://w3.tap.ibm.com/w3ki/download/attachments/370821/FIPS+140+Guidelines.pdf?version=1
- https://eventpumpz.bluehost.ibm.com/
- http://www-01.ibm.com/cgi-bin/common/ssi/ssialias?infotype=an&subtype=ca&htmlfid=897/ENUS207-015&appname=usn&language=enus
- http://devinfo.hursley.ibm.com/teamdocs/omnibus-docs/fipsrelease/training/

- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/concept/omn_con_ipv6_configurationunix.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/reference/omn_con_ipv6_unixconfiguringsupp.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/concept/omn_con_ssl_usingssl.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/task/omn_con_os_configuringobjservi18n.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/admin/reference/omn_adm_howpropscmdlnopts.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/admin/concept/omn_adm_pa_usingpctrlmanageprocesses.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/concept/omn_pln_supportedoperatingsystems.html
- http://www-01.ibm.com/common/ssi/index.wss?buttonpressed=DET001PT005&hfdd=&hfud=&timestamp=1227044178201&user=EXT&page=0&specific_index=DET001PEF011&paneltext_ann=Announcement+letter+search&DET001PGL002=DET001PEF011&DET001PEF001=OMNIbus+7.2.1&submit.x=11&submit.y=10

Public Websites

- http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2004.htm
- http://www.ipv6.org/

Special Thanks

- Fiona Davies1/UK/IBM@IBMGB
- Stephen Cook/UK/IBM@IBMGB

- Ryan Boyd/Kansas City/IBM@IBMUS

Publicly Accessible IBM Enablement Web Sites:

- http://www.ibm.com/common/ssi/index.wss
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/relnotes/omn_relnotes721.html
- http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc_7.2.1/install/concept/omn_con_fipssupport.html

