



IBM Tivoli Netcool/OMNibus V7.2.1

Configuring binaries for SSL and FIPS



© 2009 IBM Corporation
Converted to video May 29, 2015

Hello, and welcome to the OMNIBUS IBM Education assistance module, Configuring binaries for SSL and FIPS .

Introduction

Although Secure Socket Layer (SSL) security is not new to the OMNibus product, significant changes in version 7.2.1 are available to address FIPS 140-2 cryptography requirements. This module highlights updating product binaries for preparation for SSL and FIPS usage.

Some key components for OMNibus secure communications with FIPS 140-2 compliance are:

- ✓ The Java™ Runtime Environment (JRE) Included With OMNibus
- ✓ The OMNibus Global FIPS 140-2 Switch
- ✓ Enabling FIPS 140-2 Support Within iKeyman

Although Secure Socket Layer (SSL) security is not new to the OMNibus product, significant changes in version 7.2.1 are available to address FIPS 140-2 cryptography requirements. This module highlights updating product binaries for preparation for SSL and FIPS usage. Some key components for OMNibus secure communications with FIPS 140-2 compliance are:

The Java Runtime Environment (JRE) Included With OMNibus

The OMNibus Global FIPS 140-2 Switch

Enabling FIPS 140-2 Support Within iKeyman

SSL and the JRE

- **Secure Sockets Layer (SSL)** is a protocol for transmitting sensitive information over data networks.
- SSL, a cryptographic system, uses two keys to encrypt data.
 - ▶ Public key
 - ▶ Private or secret key
- You must define elements in the `java.security` file
 - Element definitions are platform specific
 - AIX® and Linux® definitions are similar
 - Solaris and HP-UX definitions are similar
 - Examples of each set of definitions are presented in the following slides.

Secure Sockets Layer is a protocol for transmitting sensitive information over data networks.

SSL, a cryptographic system, uses two keys to encrypt data. The first key is a public key the next a private or secret key. You must define different implementations of cryptographic elements in the `java.security` file. The `java.security` defines both SSL and FIPS compliance elements. Examples of each set of definitions are presented in the following slides.

Changing JRE elements

- Create a, platform specific, backup copy of **`$NCHOME/platform/linux2x86/jre_1.5.6/jre/lib/security/java.security`**.

```
shell$> cd
    $NCHOME/platform/linux2x86/jre_1.5.6/jre/lib/security/
shell$> cp java.security java.security.bak
```

- Edit the file **`$NCHOME/platform/linux2x86/jre_1.5.6/jre/lib/security/java.security`**.

```
shell$> vi java.security
```

The first step is to create a backup copy of the java.security file. Next, open the java.security file for editing.

Changing JRE elements: AIX and Linux

Type the list of security providers in the `java.security` file exactly as follows:

```
# List of providers and their preference orders:  
#  
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
```

- Note that any Java application that relies on another JRE distribution (other than the one included with OMNibus) might require additional modification to support the FIPS-compliant SSL.

Next, for AIX and Linux platforms, modify the `java.security` file to include the list of cryptographic providers in exactly the order shown. Any other JRE distributions not included with the OMNibus product distribution might require modifications.

Changing JRE elements: Solaris and HP-UX

Type the list of security providers in the java.security file exactly as follows:

```
security.provider.1=sun.security.provider.Sun
security.provider.2=com.ibm.security.jgss.IBMJGSSProvider
security.provider.3=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.4=com.ibm.crypto.provider.IBMJCE
security.provider.5=com.ibm.jsse2.IBMJSSEProvider2
security.provider.6=com.ibm.security.cert.IBMCertPath
security.provider.7=com.ibm.security.cmskeystore.CMSProvider
security.provider.8=com.ibm.crypto.pkcs11.provider.IBMPKCS11
```

If you are configuring a Solaris or HP-UX platform, modify the java.security file to include the list of cryptographic providers in exactly the order shown.

Changing JRE elements: All platforms

- Set the key and trust manager algorithm support within the JRE. The following information is also required in the `java.security` file:

```
# the javax.net.ssl package.  
ssl.KeyManagerFactory.algorithm=IbmX509  
ssl.TrustManagerFactory.algorithm=IbmX509
```

- Inform the JRE to use IBM's implementation of `SSLConnectionFactory` and `SSLServerConnectionFactory`. The following information is also required in the `java.security` file:

```
# SocketFactory or ServerConnectionFactory implementations.  
#ssl.SocketFactory.provider=  
#ssl.ServerConnectionFactory.provider=  
ssl.SocketFactory.provider=com.ibm.jsse2.SSLConnectionFactoryImpl  
ssl.ServerConnectionFactory.provider=com.ibm.jsse2.SSLServerConnectionFactoryImpl
```

- Save and close the file

Continue to edit the `java.security` file in the section under **java.net.ssl**. Type the lines for the key and trust factories. In the Socket and ServerSocket sections, add the two additional lines. After you complete those last two entries, save and close the file.

Enabling FIPS mode

- ObjectServers, process agents, proxy servers, and ObjectServer gateways can be run in FIPS 140-2 mode.
- To operate in FIPS 140-2 mode, you must create a FIPS configuration file.
- If you want to use SSL for client and server communications, you must also enable FIPS 140-2 mode for the SSL communications.
- Create an OMNIBus V7.2.1 FIPS 140-2 switch with the following commands:

```
shell$> cd $NCHOME/etc/security
shell$> touch $NCHOME/etc/security/fips.conf
```

ObjectServers, process agents, proxy servers, and ObjectServer gateways can be run in FIPS 140-2 mode. To operate in FIPS 140 2 mode, you must create a FIPS configuration file. If you want to use SSL for client and server communications, you must also enable FIPS 140 2 mode for the SSL communications. You create an OMNIBus 7.2.1 FIPS 140-2 Switch by performing the following command: **touch \$NCHOME/etc/security/fips.conf** . This command is all that is required to enable FIPS within OMNIBus.

Enable FIPS in IBM Key Manager (iKeyman)

- You must set the relevant iKeyman properties before creating the key database.
- The **ikminit.properties** file location:
\$NCHOME/platform/arch/classes/ikminit.properties
 - ▶ Where *arch* represents your operating system directory, such as **linux2x86**.
- Values changes in the file
 - ▶ **DEFAULT_FIPS_MODE_PROCESSING=ON**
 - ▶ **DEFAULT_CRYPTOGRAPHIC_BASE_LIBRARY=ICC**

To enable FIPS in iKeyman you must set iKeyman properties before creating the key database. The iKeyman.properties file is located under the operating system specific directory. There are two values you must change.

Enable FIPS in IBM Key Manager (iKeyman)

- To perform the iKeyman changes, use the following commands:

```
shell$> cd $NCHOME/platform/linux2x86/classes
shell$> cp
  $NCHOME/linux2x86/arch/classes/ikminit.properties
  $NCHOME/platform/arch/classes/ikminit.properties.bak
shell$> vi
  $NCHOME/linux2x86/arch/classes/ikminit.properties.
```

- After opening the file, modify the following properties (uncomment) using an editor:

```
DEFAULT_FIPS_MODE_PROCESSING=ON
DEFAULT_CRYPTOGRAPHIC_BASE_LIBRARY=ICC
```

To perform the iKeyman changes, use the following commands:

first, change directories to `$NCHOME/platform/linux2x86/classes`.

Next, copy `ikminit.properties` to `ikminit.properties.bak`

then use `vi ikminit.properties`.

After opening the file, modify the following properties by uncomment them using the editor:

```
DEFAULT_FIPS_MODE_PROCESSING and
DEFAULT_CRYPTOGRAPHIC_BASE_LIBRARY.
```

Encryption

- Your OMNibus binaries are now configured to use FIPS compliant encryption within a SSL-secured environment.
- You must use the Advanced Encryption Standard (AES) to encrypt SSL password values while in FIPS mode.
- To maintain FIPS compliance, you might also encrypt string values in files.
- You can use the AES encryption mechanism in ObjectServer, proxy server, probe, and gateway properties files and process agent configuration files.
- To use the Advanced Encryption Standard (AES), use the **nco_aes_crypt** command in place of other encryption commands used in version 7.2.0 and earlier of the OMNibus product.
- You can find additional information regarding AES property value encryption online or in the IBM Education Assistance Tivoli Netcool/OMNibus V7.2.1 Security AES property value encryption.

Your OMNibus binaries are now configured to use FIPS compliant encryption within a SSL-secured environment.

You must use the Advanced Encryption Standard (AES) to encrypt SSL password values while in FIPS mode.

To maintain FIPS compliance, you might also encrypt string values in files.

You can use the AES encryption mechanism in ObjectServer, proxy server, probe, and gateway properties files and process agent configuration files.

To use the Advanced Encryption Standard (AES), use the **nco_aes_crypt** command in place of other encryption commands used in version 7.2.0 and earlier of the OMNibus product.

Additional information regarding AES property value encryption can be referenced on-line or in the IBM Education Assistance Tivoli Netcool/OMNibus V7.2.1 Security AES property value encryption.

Training roadmap for Netcool® Tivoli OMNibus

http://www.ibm.com/software/tivoli/education/edu_prd.html

For further training, refer to the link,
http://www.ibm.com/software/tivoli/education/edu_prd.html.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX Netcool Tivoli

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>

Java, JRE, and all Java-based trademarks and logos are 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.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2009. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.

This concludes this module.