HP Operations Orchestration

Software Version: 10.20
Windows and Linux Operating Systems

Hardening Guide

Document Release Date: November 2014
Software Release Date: November 2014

Legal Notices

Warranty

Restricted Rights Legend

Copyright Notice

Trademark Notices

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: **http://h20230.www2.hp.com/selfsolve/manuals**

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to: **http://h20229.www2.hp.com/passport-registration.html**

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Online web site at: **http://www.hp.com/go/hpsoftwaresupport**

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

HP Software Solutions Now accesses the HPSW Solution and Integration Portal Web site. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this Web site is **http://h20230.www2.hp.com/sc/solutions/index.jsp**

Contents

Hardening HP Operations Orchestration

This document describes how to configure security hardening for HP Operations Orchestration.

For information about administrative tasks, see the *HP OO System Administration Guide*.

Disclaimer

This guide provides recommendations for safeguarding your HP OO deployment from security risks or threats. Some of the most important reasons to secure an application include protecting the confidentiality, integrity, and availability of an organization's critical information. However, to protect your HP OO data, it is necessary to secure both HP OO and the computing environment (for example, the infrastructure) upon which the application runs.

This guide is limited to recommendations to help secure HP OO at the application level and does not cover how to secure the infrastructure within the customer environment. The customer is solely responsible for understanding his/her infrastructure/environment and applying the respective hardening policies.

Security Hardening Recommendations

1. Install the latest version of HP OO. For more information, see the *HP OO Installation Guide*.

2. (Optional) Configure HP OO for FIPS 140-2 Compliance. If you choose to do this, it must be configured before you start the Central server. See " Configuring HP OO for FIPS 140-2 Level 1 Compliance" on page 24.

3. Configure the Central server certificate for TLS encryption and client certificate for strong authentication (mutual).

   Note: This can be done during installation.

   For the RAS, Debugger, and OOSH, provide certificate authentication if required (for the server certificate) and use the client certificate for authentication against the Central. See "Server and Client Certificate Authentication" on the next page.

4. Harden the HP OO Central server by removing the HTTP port and replacing the passwords of the KeyStore and TrustStore with strong passwords. See "Changing or Disabling the HTTP/HTTPS Ports" on page 16 and "Changing the KeyStore/TrustStore Password" on page 13.

5. Harden HP OO Studio by replacing the KeyStore and TrustStore passwords with strong passwords. See ""Changing the KeyStore/TrustStore Password" on page 13.

6. Remove the RC4 cipher from the SSL-supported ciphers. See "Removing the RC4 Cipher from the SSL-supported Ciphers" on page 16.

7. (Optional) Configure the TLS protocol version. See "Configuring the TLS Protocol" on page 30.

8. Enable authentication in Central. See "Enabling Authentication" in the *HP OO Central User Guide*.

   Internal users are not secured, so use a secured LDAP with a strong password policy. See "Setting Up Security – LDAP Authentication" in the *HP OO Central User Guide*.

9. Harden/secure the operating system and database.

10. Add a security banner with a meaningful message. For example, "You are now logging on to our PRODUCTION environment! Do not continue unless you are familiar with the governance rules for this system and have taken the necessary training." See "Setting Up a Security Banner" in the *HP OO Central User Guide*.

11. In the Windows and SQL server environment, configure HP OO to work with Windows authentication. See "Configuring HP OO to Work with Windows Authentication" in the *HP OO Database Guide*.

12. Make sure that auditing is enabled in Central. For more information, see "Enabling Auditing" in the the *HP OO Central User Guide*.

Server and Client Certificate Authentication

Transport Layer Security (TLS) certificates digitally bind a cryptographic key to the details of an organization, enabling secure and encrypted connections from a web server to a browser.

HP OO uses the Keytool utility to manage cryptographic keys and trusted certificates. This utility is included in the HP OO installation folder, in **<installation dir>/java/bin/keytool**. For more information about the Keytool utility, see http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html.

Note: Keytool is an open source utility.

Installations of HP OO Central include two files for the management of certificates:

- **<installation dir>/central/var/security/client.truststore**: Contains the list of trusted certificates.
- **<installation dir>/central/var/security/key.store**: Contains the HP OO certificate (private key).

It is recommended that you replace the HP OO self-signed certificate after a new installation of HP OO or if your current certificate is expired.

Encrypting the Communication Using a Server Certificate

Replacing the Central TLS Server Certificate

You can use a certificate signed by a well-known certificate authority or a custom server certificate from a local certificate authority.

Replace the parameters that are highlighted in <yellow> to match the location of the **key.store** file and other details on your computer.

> **Note:** The following procedure uses the Keytool utility that is located in **<installation dir>/java/bin/keytool**.

1. Stop Central and back up the original **key.store** file, located in **<installation dir>/central/var/security/key.store**.

2. Open a command line in **<installation dir>/central/var/security**.

3. Delete the existing server certificate from the Central **key.store** file, using the following command:

```
keytool -delete -alias tomcat -keystore key.store -storepass changeit
```

4. If you already have a certificate with **.pfx** or **.p12** extension, then go to the next step. If not, then you need to export the certificate with private key into PKCS12 format (.pfx,.p12). For example, if the certificate format is PEM:

   ```
   >openssl pkcs12 –export –in <cert.pem> -inkey <.key> -out <certificate 
   name>.p12 –name <name>
   ```

   If the certificate format is DER, add the `–inform DER` parameter after pkcs12.For example:

   ```
   >openssl pkcs12 –inform DER –export –in <cert.pem> -inkey <.key> -out 
   <certificate name>.p12 –name <name>
   ```

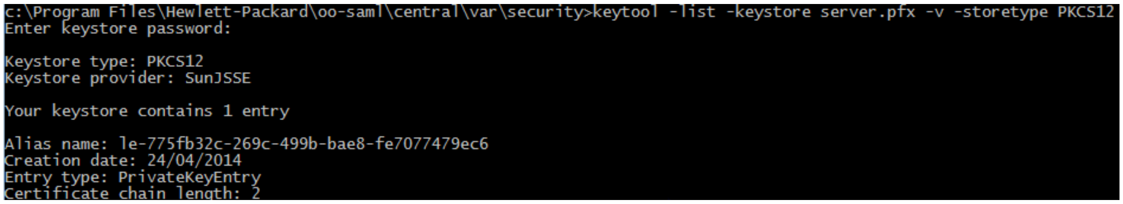
   > **Note:** Make a note of the password that you provide. You will need this password for the private key when you input the KeyStore passphrase later in this procedure.
   >
   > Make sure to choose a strong password.

5. List the alias for your certificate’s alias, using the following command:

   ```
   keytool -list -keystore <certificate_name> -v -storetype PKCS12
   ```

   The certificate alias is displayed and should be provided in the next command.

   In the example below, it is the fourth line from the bottom.

   ```
   c:\Program Files\Hewlett-Packard\oo-saml\central\var\security>keytool -list -keystore server.pfx -v -storetype PKCS12
   Enter keystore password:

   Keystore type: PKCS12
   Keystore provider: SunJSSE

   Your keystore contains 1 entry

   Alias name: le-775fb32c-269c-499b-bae8-fe7077479ec6
   Creation date: 24/04/2014
   Entry type: PrivateKeyEntry
   Certificate chain length: 2
   ```

6. Import the PKCS12 format server certificate to the Central **key.store** file using the following command:

   ```
   keytool -importkeystore -srckeystore <PKCS12 format certificate path> -
   destkeystore key.store -srcstoretype pkcs12 -deststoretype JKS -alias <cert 
   alias> -destalias tomcat
   ```

7. If the imported server certificate has a different password from the original server certificate, it is important to change the keyPass password. Follow the instructions in "Changing the KeyStore/TrustStore Password" on page 13.

   It is also recommended to change the default “changeit” password in the automatically generated KeyStore in the Central server. See "Changing the KeyStore/TrustStore Password" on page 13.

8. Start Central.

Importing a CA Root Certificate to a RAS TrustStore

After installing a RAS, if you are using a custom root certificate for Central and you didn't provide this root certificate during the RAS installation, you will need to import the trusted root certificate authority (CA) to the RAS **client.truststore**. If you are using a well known root CA (like Verisign) you do not have to perform the following procedure, because the certificate will already be in the **client.truststore** file.

By default, HP OO supports all self-signed certificates. However, in a production environment, it is recommended to change this default to a custom CA or a well known CA for security reasons.

Replace the parameters that are highlighted in <yellow>.

> **Note:** The following procedure uses the Keytool utility that is located in **<installation dir>/java/bin/keytool**.

1. Stop RAS and back up the original **client.truststore** file, located in **<installation dir>/ras/var/security/client.truststore**.

2. Open command line in **<installation dir>/ras/var/security**.

3. Open the **<installation dir> ras/conf/ras-wrapper.conf** file and set the `-Dssl.support-self-signed` value to **false**. This enables the trusted root certificate authority (CA).

   For example:

   ```
   wrapper.java.additional.<x>=-Dssl.support-self-signed=false
   ```

4. Open the **<installation dir> ras/conf/ras-wrapper.conf** file and set the `-Dssl.verifyHostName` to **true**. This verifies that the FQDN in the certificate matches the FQDN of the request.

   For example:

   ```
    wrapper.java.additional.<x>=-Dssl.verifyHostName=true
   ```

5. Import the trusted root certificate authority (CA) to the RAS **client.truststore** file if it doesn't already exist in the CA list (by default, all the well known CAs are there):

   ```
   keytool -importcert -alias <any_alias> -keystore client.truststore -file
   <certificate_name.cer> -storepass <changeit>
   ```

6. Start RAS.

Importing a CA Root Certificate to the OOSH TrustStore

If you are using a custom root certificate for Central, you will need to import the trusted root certificate authority (CA) to the OOSH **client.truststore**. If you are using a well known root CA (like Verisign) you do not have to perform the following procedure, because the certificate will already be in the **client.truststore** file.

By default, HP OO supports all self-signed certificates. However, in a production environment, it is recommended to change this default to a custom CA or a well known CA for security reasons.

Replace the parameters that are highlighted in <yellow>.

> **Note:** The following procedure uses the Keytool utility that is located in **<installation dir>/java/bin/keytool**.

1. Stop Central and back up the original **client.truststore** file, located in **<installation dir>/central/var/security/client.truststore**.

2. Edit the **oosh.bat** from **<installation dir>/central/bin**.

3. Set the `-Dssl.support-self-signed` value to **false**. This enables the trusted root certificate authority (CA).

   For example:

   ```
   -Dssl.support-self-signed=false
   ```

4. Set the `-Dssl.verifyHostName` to **true**. This verifies that the FQDN in the certificate matches the FQDN of the request.

   For example:

   ```
   -Dssl.verifyHostName=true
   ```

5. Import the trusted root certificate authority (CA) to the Central **client.truststore** file if it doesn't already exist in the CA list (by default, all the well known CAs are there):

   ```
   keytool -importcert -alias <any_alias> -keystore client.truststore -file
   <certificate_name.cer> -storepass <changeit>
   ```

6. Run OOSH.

7. Start Central.

Importing a CA Root Certificate to the Studio Debugger TrustStore

After installing Studio, if you are using a custom root certificate for Studio, you will need to import the trusted root certificate authority (CA) to the Studio **client.truststore**. If you are using a well known root CA (like Verisign) you do not have to perform the following procedure, because the certificate will already be in the **client.truststore** file.

By default, HP OO supports all self-signed certificates. However, in a production environment, it is recommended to change this default to a custom CA or a well known CA for security reasons.

Replace the parameters that are highlighted in <yellow>.

> **Note:** The following procedure uses the Keytool utility that is located in **<installation dir>/java/bin/keytool**.

1. Close Studio and back up the original **client.truststore** file, located in **<installation dir>/studio/var/security/client.truststore**.

2. Edit the **Studio.l4j.ini** file from **<installation dir>/studio**.

3. Set the `-Dssl.support-self-signed` value to **false**. This enables the trusted root certificate authority (CA).

   For example:

   ```
   -Dssl.support-self-signed=false
   ```

4. Set the `-Dssl.verifyHostName` to **true**. This verifies that the FQDN in the certificate matches the FQDN of the request.

   For example:

   ```
   -Dssl.verifyHostName=true
   ```

5. Import the trusted root certificate authority (CA) to the Studio **client.truststore** file if it doesn't already exist in the CA list (by default, all the well known CAs are there):

   ```
   keytool -importcert -alias <any_alias> -keystore client.truststore -file
   <certificate_name.cer> -storepass <changeit>
   ```

6. Start Studio.

For more information, see "Debugging a Remote Central with Studio" in the *Studio Authoring Guide*.

Changing the KeyStore/TrustStore Password

Changing the KeyStore, TrustStore, and Server Certificate Passwords in the Central Configuration

1. Make sure that Central is running.

   > **Note:** Before doing this step, make sure that there are encrypted passwords. For information about how to encrypt a password, see 'Encrypting Passwords" in the *HP OO Administration Guide*.

   From OOSH, run the following command:

   ```
   set-sys-config --key <keyName> --value <ecryptedPassword>
   ```

   where `<keyName>` is one the values from the table below:

Configuration item	Action
`key.store.password`	To set the password used to access to the **key.store**. The default value is `"changeit"`. This needs to correspond with the value for `keystorePass`, as set in the steps below.
`key.store.private.key.alias.password`	To set the password used to access the server certificate (private key) from the **key.store**. The default value is `"changeit"`. This needs to correspond with the value for `keyPass`, as set in the steps below.

2. Stop the Central service.

3. Change the KeyStore, TrustStore, and server certificate password using Keytool.

4. Change the passwords also in the **server.xml** file located in **<installation dir>/central/tomcat/conf/server.xml**.

a. Locate the HTTPS connector. For example:

```
keyPass="changeit" keystoreFile="C:/Program Files/Hewlett-Packard/HP
Operations Orchestration/central/var/security/key.store"
keystorePass="changeit" keystoreType="JKS" maxThreads="200" port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https"
secure="true" sslProtocol="TLSv1.2"
sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2" truststoreFile="C:/Program
Files/Hewlett-Packard/HP Operations
Orchestration/central/var/security/client.truststore"
truststorePass="changeit" truststoreType="JKS"/>
```

b. Change the required password.

- `keyPass` - the password used to access the server certificate private key from the specified key.store file. The default value is "`changeit`".

- `keystorePass` - the password used to access the specified key.store file. The default value is the value of the `keyPass` attribute.

> **Note:** It is recommend not to use the same password as the **keyPass**, and to use a strong password.

- `truststorePass` - the password to access the TrustStore (that includes all of the trusted CAs). The default is the value of the **javax.net.ssl.trustStorePassword** system property. If that property is null, no TrustStore password will be configured. If an invalid TrustStore password is specified, a warning will be logged and an attempt will be made to access the TrustStore without a password, which will skip validation of the TrustStore contents.

c. Save the file.

5. Edit the **central-wrapper.conf** file located in **<installation dir> central\conf\central-wrapper.conf** and change:

```
wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword=changeit
```

6. Start the Central sevice.

Changing the RAS, OOSH, and Studio TrustStore Passwords

> **Note:** You should change the KeyStore, TrustStore, and server certificate password using Keytool, before completing the following steps.

- **To change the standalone RAS TrustStore password**: Edit the **ras-wrapper.conf** file and change the `changeit` parameter of the TrustStore.

- **To change the OOSH TrustStore password**: Edit the **oosh.bat** file and change the `changeit` parameter of the TrustStore.
- **To change the Studio TrustStore password**: Edit the **<installation dir>/studio/Studio.l4j.ini** file and replace the `changeit` parameter of the TrustStore with the new password in obfuscated form.

  For information about how to obfuscate a password, see "Obfuscating Passwords" in the *HP OO Administration Guide*.

Obfuscating the Studio KeyStore and TrustStore Passwords

In HP 10.20 and later, the KeyStore and TrustStore passwords from Studio are obfuscated. After an upgrade from 10.10 to 10.20, these passwords will be obfuscated only if they are left unchanged in the **<install_dir>/studio/Studio.l4j.ini** file. Any other passwords that were changed in previous versions will not be automatically obfuscated during upgrade.

If you want to change the TrustStore or KeyStore password, you can do this in the **Studio.l4j.ini** file in obfuscated form or in plain text. After the passwords are changed, you will need to obfuscate them manually, to ensure that they do not appear in plain text in the Task Manager for the Studio process:

1. Close Studio.
2. Locate the **encrypt-password** script, in **<installation_folder>/central/bin**.
3. Obfuscate the custom password by issuing the following command:

   ```
   encrypt-password.bat --obfuscate <your password>
   ```

4. Change the passwords from the **<install_dir>/studio/Studio.l4j.ini** file for the following parameters:

   ```
   -Djavax.net.ssl.keyStorePassword={OBFUSCATED}<obfuscated_password>

   -Djavax.net.ssl.trustStorePassword={OBFUSCATED}<obfuscated_password>
   ```

5. Change the Studio KeyStore and TrustStore password with the keytool from the **<install_dir>/studio/var/security/** folder.

   Note: If the client certificate is not configured for Studio Remote Debugger, the keyStore path argument will be ignored.

6. Start Studio.

Important! After using the **encrypt-password** script, clear the command history.

This is because, on a Linux OS, the password parameter will be stored in cleartext under **/$USER/.bash_history** and accessible by the `history` command.

Removing the RC4 Cipher from the SSL-supported Ciphers

The remote host supports the use of the RC4 cipher. This cipher is flawed in its generation of a pseudo-random stream of bytes so that a wide variety of small biases are introduced into the stream, decreasing its randomness.

If plain text is repeatedly encrypted (for example, HTTP cookies), and an attacker is able to obtain many (i.e., tens of millions of) cipher texts, the attacker may be able to derive the plain text.

Disable the RC4 cipher on the JRE level (starting with Java 7):

1. Open the **$JRE_HOME/lib/security/java.security** file.

2. Disable the RC4 cipher by removing the comments and changing the parameters according to the example below:

   ```
   jdk.certpath.disabledAlgorithms=RC4, MD2, RSA keySize < 1024

   jdk.tls.disabledAlgorithms=RC4, MD5, DSA, RSA keySize < 1024
   ```

3. Restart the HP OO Central server.

For more information, see http://stackoverflow.com/questions/18589761/restict-cipher-suites-on-jre-level.

Note: After upgrading from an earlier version of HP OO 10.x, repeat these steps.

Changing or Disabling the HTTP/HTTPS Ports

The file **server.xml** under **[OO_HOME]/central/tomcat/conf** contains two elements named **<Connector>** under the element **<Service>**. These connectors define or enable the ports that the server are listening to.

Each connector configuration is defined through its attributes. The first connector defines a regular HTTP connector and the second defines an HTTPS connector.

By default, the connectors look as follows.

HTTP connector:

```
<Connector URIEncoding="UTF-8" compression="on" connectionTimeout="20000"
port="8080" protocol="org.apache.coyote.http11.Http11NioProtocol"
redirectPort="8443"/>
```

HTTPS connector:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false"
compression="on" keyAlias="tomcat" keyPass="changeit" keystoreFile="C:/Program
Files/Hewlett-Packard/HP Operations
Orchestration/central/var/security/key.store" keystorePass="changeit"
keystoreType="JKS" maxThreads="200" port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https"
secure="true" sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
truststoreFile="C:/Program Files/Hewlett-Packard/HP Operations
Orchestration/central/var/security/client.truststore" truststorePass="changeit"
truststoreType="JKS"/>
```

By default, both are enabled.

> **Important!** If you change or disable one of the Central ports in the **server.xml** file, you will also need to update the **central-wrapper.conf** file and each **RAS-wrapper.conf** file to point to the Central URL with the updated port. Otherwise, all your flows will fail when run from Central. Also, make sure to check the load balancer configurations.

Changing Port Values

To change the values of one of the ports:

1. Edit the **server.xml** file located in **<installation_dir>/central/tomcat/conf/server.xml**.
2. Locate the HTTP or HTTPS connector, and adjust the **port** value in the line.

   > **Note:** If you are keeping both HTTP and HTTPS active and you want to change the HTTPS port, you will need to change the **redirectPort** value for the HTTP connector and the **port** value for the HTTPS connector.

3. Save the file.
4. Restart Central.

Disabling a Port

For example, you might want to disable the HTTP port, for security reasons, so that the only communication channel will be on TLS and encrypted.

To disable one of the ports:

1. Edit the **server.xml** file located in **<installation_dir>/central/tomcat/conf/server.xml**.
2. Locate the HTTP or HTTPS connector, and delete or comment out the line.
3. Import the trusted root certificate authority (CA) to the Central **client.truststore** file, if it doesn't

already exist in the CA list:

```
keytool -importcert -alias <any_alias> -keystore client.truststore -file
<certificate_name.cer> -storepass <changeit>
```

> **Note:** If you are using a well known root CA (like Verisign) you do not have to perform this step, because the certificate will already be in the **client.truststore** file.

4. Save the file.
5. Restart Central.

> **Note:** It is also possible to disable the HTTP port during installation.

Troubleshooting

If the server doesn't start, open the **wrapper.log** file and look for an error in `ProtocolHandler ["http-nio-8443"]`.

This can happen when Tomcat is initializing or starting the connector. There are many variations but the error message can provide information.

All the HTTPS connector parameters are in the Tomcat configuration file located at **C:\HP\oo\central\tomcat\conf\server.xml**.

Open the file and scroll to the end, until you see the HTTPS connector:

```
<Connector SSLEnabled="true" clientAuth="false" keyAlias="tomcat"
keystoreFile="C:/HP/oo/central/var/security/keystore.p12" keystorePass="tomcat-
keystore-password" keystoreType="PKCS12" maxThreads="200" port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="true"
sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"/>
```

See if there is any mismatch in the parameters, by comparing them to the parameters you entered in the previous steps.

Client Certificate Authentication (Mutual Authentication)

The most common use of X.509 certificate authentication is in verifying the identity of a server when using TLS, most commonly when using HTTPS from a browser. The browser automatically checks that the certificate presented by a server has been issued by one of a list of trusted certificate authorities, which it maintains.

You can also use TLS with mutual authentication. The server requests a valid certificate from the client as part of the TLS handshake. The server authenticates the client by checking that its certificate is signed by an acceptable authority. If a valid certificate has been provided, it can be obtained through the servlet API in an application.

Configuring Client Certificate Authentication in Central

Before you configure the client certificate authentication in Central, make sure you have configured the TLS server certificate, as described in "Server and Client Certificate Authentication" on page 7.

Set the `clientAuth` attribute to `true` if you want the TLS stack to require a valid certificate chain from the client before accepting a connection. Set to `want` if you want the TLS stack to request a client certificate, but not fail if one is not presented. A `false` value (default) does not require a certificate chain unless the client requests a resource protected by a security constraint that uses CLIENT-CERT authentication. (For more information, see the Apache Tomcat Configuration Reference).

Set the **Certificate Revocation List (CRL)** file. This can contain several CRLs.In the operation of some cryptosystems, usually public key infrastructures (PKIs), a certificate revocation list (CRL) is a list of certificates (or more specifically, a list of serial numbers for certificates) that have been revoked, and therefore, entities presenting those (revoked) certificates should no longer be trusted.

> **Note:** The following procedure uses the Keytool utility that is located in **<installation dir>/java/bin/keytool**.

1. Stop the Central server.

2. Import the appropriate root certificate (CA) into Central **client.truststore**: **<installation dir>/central/var/security/client.truststore**, if it doesn't already exist in the CA list (by default, all the well known CAs are there). For example:

   ```
   keytool -importcert -alias <any_alias> -keystore <path>/client.truststore -file
   <certificate_path> -storepass <changeit>
   ```

3. Edit the **server.xml** file located in **<installation_dir>/central/tomcat/conf/server.xml**.

4. Set the `clientAuth` attribute in the `Connector` tag to `want` or `true`. The default is `false`.

   > **Note:** We recommend starting the server at the end of this procedure, but note that it is also possible to start the server at this point.

5. (Optional) Add the `crlFile` attribute to define the certificate revocation list file for the TLS certificate validation, for example:

   ```
    crlFile="<path>/crlname.<crl/pem>"
   ```

The file can be with the `.crl` extension for a single certificate revocation list or with the `.pem` (PEM CRL format) extension for one or more certificate revocation lists. The PEM CRL format uses the following header and footer lines:

```
-----BEGIN X509 CRL-----
-----END X509 CRL-----
```

Example of the `.pem` file structure for one CRL (for more than one, concatenate another CRL block):

```
-----BEGIN X509 CRL-----
MIIBbzCB2QIBATANBgkqhkiG9w0BAQUFADBeMQswCQYDVQQGEwJVUzEYMBYGA1UE
ChMPVS5TLiBHb3Zlcm5tZW50MQwwCgYDVQQLEwNEb0QxEDAOBgNVBAsTB1Rlc3Rp
bmcxFTATBgNVBAMTDFRydXN0IEFuY2hvchcNOTkwMTAxMTIwMTAwWhcNNDgwMTAx
MTIwMTAwWjAiMCACAScXDTk5MDEwMTEyMDAwMFowDDAKBgNVHRUEAwoBAaAjMCEw
CgYDVR0UBAMCAQEwEwYDVR0jBAwwCoAIq5rr+cLnVI8wDQYJKoZIhvcNAQEFBQAD
gYEAC7lqZwejJRW7QvzH11/7cYcL3racgMxH3PSU/ufvyLk7ahR++RtHary/WeCv
RdyznLiIOA8ZBiguWtVPqsNysNn7WLofQIVa+/TD3T+lece4e1NwGQvj5Q+e2wRt
GXg+gCuTjTKUFfKRnWz7O7RyiJKKim0jtAF4RkCpLebNChY=
-----END X509 CRL-----
```

6. Start the Central server.

Note: For each client certificate, you need to define a user, either an internal user or LDAP user. The name of the user should be defined in the certificate attributes. The default is value of the CN attribute. See the Processing a Certificate Principal section for more details.

Note that even if HP OO is set up with multiple LDAP configurations, it is only possible to authenticate the user using the client certificate attributes with the default LDAP. Central will first try to authenticate the user with the default LDAP, and if this fails, will try to authenticate within the HP OO internal domain.

Updating the Configuration of a Client Certificate in RAS

The client certificate is configured during the installation of the RAS. However, if you need to update the client certificate, you can do this manually in the **ras-wrapper.conf** file.

Prerequisite: You must import the CA root certificate of Central into the RAS TrustStore. See "Importing a CA Root Certificate to a RAS TrustStore" on page 10.

To update the configuration of the client certificate in an external RAS:

1. Stop the RAS server.

2. Open the **ras-wrapper.conf** file from <**installation dir>ras/conf/ras-wrapper.conf**.

3. Change the following according to the your client certificate:

```
wrapper.java.additional.<x>=-Djavax.net.ssl.keyStore=<installation
dir>/var/security/certificate.p12"

wrapper.java.additional.<x>=-Djavax.net.ssl.keyStorePassword=changeit

wrapper.java.additional.<x>=-Djavax.net.ssl.keyStoreType=PKCS12
```

4. Start the RAS server.

> **Important Note!** The X.509 client certificate needs to have the principal name of the RAS, which is the RAS ID (see Processing a Certificate Principal).
>
> You can find the RAS ID under the **Topology** tab in Central. See "Setting Up Topology – Workers" in the *HP OO Central User Guide*.

Configuring a Client Certificate in Studio Remote Debugger

Prerequisite: You must import the CA root certificate of Central into the Studio Debugger TrustStore. See "Importing a CA Root Certificate to the Studio Debugger TrustStore" on page 11.

To configure the client certificate in the Studio Remote Debugger:

1. Close Studio.
2. Edit the **Studio.l4j.ini** file from **<installation dir>/studio**.
3. Change the following according to the your client certificate:

```
-Djavax.net.ssl.keyStore="<installation
dir>/studio/var/security/certificate.p12"

-Djavax.net.ssl.keyStorePassword=changeit

-Djavax.net.ssl.keyStoreType=PKCS12
```

4. Start Studio.

> **Notes**:
>
> - In HP OO 10.20 and later, the `keyStorePassword` parameter in **Studio.l4j.ini** is obfuscated by default if the password was kept as default. You can change this parameter and store it in either clear text or obfuscated.
> - For the client certificate, you need to define a user, either an internal user or LDAP user. The name of the user should be defined in the certificate attributes. The default is value of the CN attribute. See the Processing a Certificate Principal section for more details.
> - Note that even if HP OO is set up with multiple LDAP configurations, it is only possible to authenticate the user using the client certificate attributes with the default LDAP. Central will

first try to authenticate the user with the default LDAP, and if this fails, will try to authenticate within the HP OO internal domain.

Configuring a Client Certificate in OOSH

Prerequisite: You must import the CA root certificate of Central into the OOSH TrustStore. See "Importing a CA Root Certificate to the OOSH TrustStore" on page 10.

1. Stop OOSH.

2. Edit the **oosh.bat** from **<installation dir>/central/bin**.

3. Change the following according to the your client certificate:

   ```
   -Djavax.net.ssl.keyStore="<installation dir>/var/security/certificate.p12"
   ```

   ```
   -Djavax.net.ssl.keyStorePassword=changeit
   ```

   ```
   -Djavax.net.ssl.keyStoreType=PKCS12
   ```

4. Start OOSH.

Note: For the client certificate, you need to define a user, either an internal user or LDAP user. The name of the user should be defined in the certificate attributes. The default is value of the CN attribute. See the Processing a Certificate Principal section for more details.

Note that even if HP OO is set up with multiple LDAP configurations, it is only possible to authenticate the user using the client certificate attributes with the default LDAP. Central will first try to authenticate the user with the default LDAP, and if this fails, will try to authenticate within the HP OO internal domain.

Processing Certificate Policies

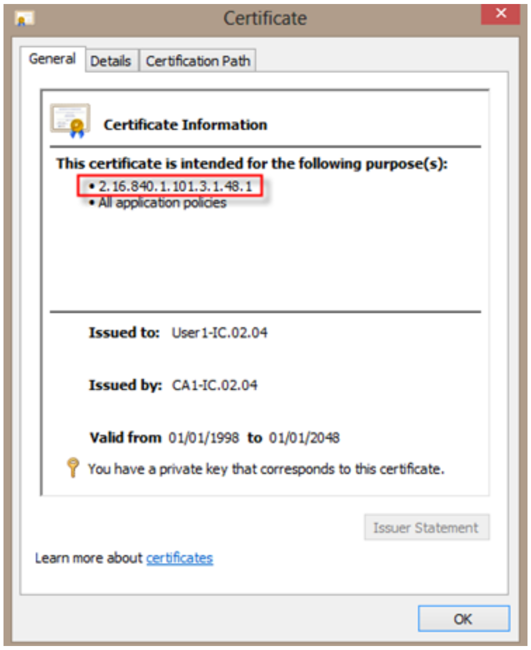
HP OO handles the processing of certificate polices for the end certificate.

- You can set the purpose string in the certificate.

- HP OO lets you add the policy string(s) as a configuration item and check the policy string of each end certificate. If it does not match, reject the certificate.

- Enable or disable the certificate policy verification by adding the following configuration item: `x509.certificate.policy.enabled=true/false` (default is `false`).

- Define the policy list by adding the following configuration item: `x509.certificate.policy.list=<comma_separated_list>` (the default is an empty list).



Certificate Information

This certificate is intended for the following purpose(s):

- 2.16.840.1.101.3.1.48.1
- All application policies

Issued to: User1-IC.02.04

Issued by: CA1-IC.02.04

Valid from 01/01/1998 **to** 01/01/2048

You have a private key that corresponds to this certificate.

Learn more about certificates

For more information about how to change HP OO system properties, see the *HP OO Shell Guide*.

Processing a Certificate Principal

You can define how to get the principal from a certificate using a regular expression match against the `Subject`. The regular expression should contain a single group. The default expression `CN=(.?)` matches the common name field. For example, `CN=Jimi Hendrix, OU=` assigns a user name of `Jimi Hendrix`.

- The matches are case-insensitive.
- The principal of the certificate is the user name in HP OO (LDAP or internal user).
- To change the regular expression, change the configuration item: `x509.subject.principal.regex`.

For more information about how to change HP OO system properties, see the *HP OO Shell User Guide*.

Configuring HP OO for FIPS 140-2 Level 1 Compliance

This section explains how to configure HP Operations Orchestration to be compliant with Federal Information Processing Standards (FIPS) 140-2 Level 1.

FIPS 140-2 is a standard for security requirements for cryptographic modules defined by the National Institute of Standards Technology (NIST). To view the publication for this standard, go to: csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

After you have configured HP OO for FIPS 140-2 compliance, HP OO uses the following security algorithm:

- Symmetric-key algorithm: AES256
- Hashing algorithm: SHA256

HP OO uses the security provider: RSA BSAFE Crypto software version 6.1.This is the only supported security provider for FIPS 140-2.

Note: Once you have configured HP OO to be compliant with FIPS 140-2, you cannot revert back to the standard configuration unless you re-install HP OO.

Prerequisites

Note: If you are upgrading from an installation of HP OO 10.10 (and later) that was already configured with FIPS, you must repeat steps 4 and 5 below, and then repeat the steps in the "Configure the Properties in the Java Security File" section in "Configuring HP OO to be Compliant with FIPS 140-2" on page 26.

Before configuring HP OO to be compliant with FIPS 140-2, perform the following steps:

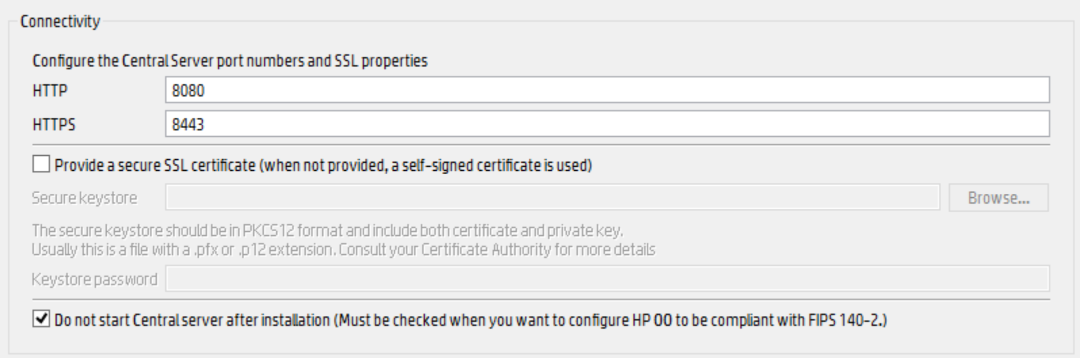
Note: In order to be FIPS140-2 compliant, you need to turn off LWSSO.

1. Verify that you are configuring a new installation of HP OO version 10.10 or higher to be compliant with FIPS 140-2, and that it is not in use.

   You cannot configure an installation of HP OO that is in use (whether version 9.x or 10.x).

2. Verify that when HP OO was installed, it was configured not to start the Central server after installation:

- In a silent installation, the `should.start.central` parameter was set to **no**.
- In a wizard installation, in the **Connectivity** step, the **Do not start Central server after installation** check box was selected.



Connectivity

Configure the Central Server port numbers and SSL properties

HTTP 8080

HTTPS 8443

□ Provide a secure SSL certificate (when not provided, a self-signed certificate is used)

Secure keystore

The secure keystore should be in PKCS12 format and include both certificate and private key.
Usually this is a file with a .pfx or .p12 extension. Consult your Certificate Authority for more details

Keystore password

☑ Do not start Central server after installation (Must be checked when you want to configure HP OO to be compliant with FIPS 140-2.)

3. Back up the following directories:
   - **<installation dir>\central\tomcat\webapps\oo.war**
   - **<installation dir>\central\tomcat\webapps\PAS.war**
   - **<installation dir>\central\conf**
   - **<oo_jre>\lib\security** (where **<oo_jre>** is the directory in which the JRE used by HP OO is installed. By default, this is **<installation dir>\java**)
4. Download and install the Java Cryptographic Extension (JCE) Unlimited Strength Jurisdiction Policy Files from the following site: http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html.

   Note: See the **ReadMe.txt** file from the downloaded content for information on how to deploy the files and upgrade the JRE used by HP OO.
5. Install the RSA BSAFE Crypto software files. On the system on which HP OO is installed, copy the following to **<oo_jre>\lib\ext\** (where **<oo_jre>** is the directory in which the JRE that is used by HP OO is installed. By default, this is **<installation dir\java**).
   - **<installation dir>\central\lib\cryptojce-6.1.jar**
   - **<installation dir>\central\lib\cryptojcommon-6.1.jar**
   - **<installation dir>\central\lib\jcmFIPS-6.1.jar**

Note: If you are upgrading from an installation of HP OO 10.10 (and later) that was already configured with FIPS, you must repeat steps 4 and 5 from the "Prerequisites" section above, and then repeat the steps in the "Configure the Properties in the Java Security File" section in

"Configuring HP OO to be Compliant with FIPS 140-2" below.

Configuring HP OO to be Compliant with FIPS 140-2

The following list shows the procedures that you need to perform in order to configure HP OO to be compliant with FIPS 140-2:

- Configure the Properties in the Java Security File
- Configure the encryption.properties File and Enable FIPS Mode
- Create FIPS-Compliant HP OO Encryption
- Re-encrypt the database password with the new encryption
- Start HP OO

Configure the Properties in the Java Security File

Edit the Java security file for the JRE to add additional security providers and configure the properties for FIPS 140-2 compliance.

Note: The upgrade to HP OO 10.10 completely replaces the installed JRE files. Therefore, the following steps must be done after upgrading to 10.10.

Note: If you are upgrading from an installation of HP OO 10.10 (and later) that was already configured with FIPS, you must repeat steps 4 and 5 from the "Prerequisites" section in " Configuring HP OO for FIPS 140-2 Level 1 Compliance" on page 24, and then repeat the steps here.

Open the **<oo_jre>\lib\security\java.security** file in an editor and perform the following steps:

1. For every provider listed, in the format **security.provider.<nn>=<provider_name>**, increment the preference order number `<nn>` by two.

   For example, change a provider entry from:

   ```
   security.provider.1=sun.security.provider.Sun
   ```

   to

   ```
   security.provider.3=sun.security.provider.Sun
   ```

2. Add a new default provider (RSA JCE). Add the following provider at the top of the provider list:

```
security.provider.1=com.rsa.jsafe.provider.JsafeJCE
```

3. Add the RSA BSAFE SSL-J Java Secure Sockets Extension (JSSE) Provider.

   ```
   security.provider.2=com.rsa.jsse.JsseProvider
   ```

4. Copy and paste the following line into the **java.security** file to ensure **RSA BSAFE** is used in FIPS 140-2 compliant mode:

   ```
   com.rsa.cryptoj.fips140initialmode=FIPS140_SSL_MODE
   ```

   You can paste this line anywhere in the **java.security** file.

5. Because the default DRBG algorithm ECDRBG128 is not safe (according to NIST), set the security property **com.rsa.crypto.default** to **HMACDRBG**, by copying the following line into the **java.security** file:

   ```
   com.rsa.crypto.default.random=HMACDRBG
   ```

   You can paste this line anywhere in the **java.security** file.

6. Save and exit the **java.security** file.

Configure the encryption.properties File and Enable FIPS Mode

The HP OO encryption properties file must be updated to be FIPS 140-2 compliant.

1. Back up the **encryption.properties** file, which is located in **<installation dir>\central\var\security**.

2. Open the **encryption.properties** file in a text editor. For example, edit the following file:

   C:\Program Files\Hewlett-Packard\HP Operations Orchestration\central\var\security\encryption.properties.

3. Locate `keySize=128` and replace it with `keySize=256`.

4. Locate `secureHashAlgorithm=SHA1` and replace it with `secureHashAlgorithm=SHA256`.

5. Locate `FIPS140ModeEnabled=false` and replace it with `FIPS140ModeEnabled=true`.

   Note: If `FIPS140ModeEnabled=false` does not exist, add `FIPS140ModeEnabled=true` as a new line to the end of the file.

6. Save and close the file.

Create FIPS-Compliant HP OO Encryption

To create or replace the HP OO encryption store file, so that it is FIPS compliant, see "Replacing the FIPS Encryption" below.

Note: AES has three approved key lengths: 128/192/256 by NIST SP800-131A publication.

The following secure hash algorithms are supported in FIPS: SHA1, SHA256, SHA384, SHA512.

Note: It is recommended to change the passwords of the key.store (and its private key entry) and the truststore. See "Changing the KeyStore/TrustStore Password" on page 13.

Note: It is recommended to delete all the default CA root certificates that are not in use from the HP OO truststore. (The **client.truststore** is located at **<installation>/central/var/security**.)

Re-encrypt the database password with the new encryption

Re-encrypt the database password, as described in the *HP OO Administration Guide*, in "Changing the Database Password".

Start HP OO

Start HP OO, as described in the *HP OO Installation Guide*.

Replacing the FIPS Encryption

HP OO, Central, and RAS comply with Federal Information Processing Standard 140-2 (FIPS 140-2), which defines the technical requirements to be used by federal agencies when these organizations specify cryptographic-based security systems for protection of sensitive or valuable data.

After a fresh installation of HP OO 10.10, you have the option to change the FIPS encryption key.

Note: This procedure is only for fresh installations. You cannot perform it after an upgrade.

Changing the FIPS Encryption Key on Central

1. Go to **<Central installation folder>/var/security**.
2. Back up and delete the **encryption_repository** file.
3. Go to **<Central installation folder>/bin**.

4. Run the **generate-keys** script.

   A new master key is generated in **<Central installation folder>/var/security/encryption_repository**.

Changing the RAS Encryption Properties

If the installation of the RAS is in a new location, you need to complete all the steps below.

> **Note:** These changes are only valid if you working on a new RAS installation after you have changed the Central encryption properties.

To change the RAS encryption properties:

1. Complete all the steps in the "Prerequisites" section in " Configuring HP OO for FIPS 140-2 Level 1 Compliance" on page 24.

2. Complete all the steps in the "Configure the Properties in the Java Security File" in "Configuring HP OO to be Compliant with FIPS 140-2" on page 26.

3. Copy the current **encryption.properties** file from **<installation dir>\ras\var\security** to the **<installation dir>\ras\bin** folder.

4. Using any text editor, edit and change the **encryption.properties** file as required.

   For more information, see "Configure the encryption.properties File and Enable FIPS Mode" in "Configuring HP OO to be Compliant with FIPS 140-2" on page 26.

5. Save the changes.

6. Open a command line prompt in the folder **<installation dir>\ras\bin**.

7. Run **oosh.bat**.

8. Run the OOShell command: `replace-encryption --file encryption.properties`

   > **Note:** If you copied the **encryption.properties** file to a different folder, make sure you enter the correct location in the OOShell command.

9. Restart the RAS service.

Configuring the TLS Protocol

You can configure HP OO to define the supported TSL protocol version. By default, HP OO allows TLS v1, TLS v1.1 and TLS v1.2, but you can narrow this down.

Note: SSLv3 and other versions of SSL are not supported.

1. Open the file **<installation_folder>/central/tomcat/conf/server.xml**.
2. Locate the SSL connector (at the end of the file).
3. Edit the default value of `sslEnabledProtocols`. For example, change

   `sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"` to

   `sslEnabledProtocols="TLSv1.2"`
4. Restart the server.

Java™
COMPATIBLE