HP Operations Orchestration

For the Windows and Linux

Software Version: 10.02

Configuration and Hardening Guide

Document Release Date: January 2014

Software Release Date: January 2014

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

AMD and the AMD Arrow symbol are trademarks of Advanced Micro Devices, Inc.

Google™ and Google Maps™ are trademarks of Google Inc.

Intel®, Itanium®, Pentium®, and Intel® Xeon® are trademarks of Intel Corporation in the U.S. and other countries.

Java is a registered trademark of Oracle and/or its affiliates.

Microsoft®, Windows®, Windows NT®, Windows® XP, and Windows Vista® are U.S. registered trademarks of Microsoft Corporation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

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

Configuration and Hardening

This document describes how to configure and harden HP Operations Orchestration 10.00 and later.

Server and Client Certificate Authentication

Secure Socket Layer (SSL) certificates digitally bind a cryptographic key to the details of an organization, enabling secure 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.

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.

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

Server Certificate Authentication

Replacing the Central SSL Server Certificate

You can use a certificate signed by a well-known company or a custom server certificate.

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.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.

5. Import the PKCS12 format server certificate to the Central **key.store** file:

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

6. Start Central.

Replacing the Central SSL Server Certificate With a Self-Signed Certificate

You can generate a self-signed certificate using the Keytool utility.

> **Note:** After upgrading to HP OO 10.00:
>
> - If a new Central is installed on the same machine as the previous installation, you can use the existing self-signed certificate.
> - If new Centrals are installed on different machines, you need to generate a new self-signed certificate for each one, even if you had a certificate for the previous version.

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

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

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. Generate a self-signed certificate:

   ```
   keytool -genkey -alias tomcat -keyalg RSA -keypass changeit -keystore path/for/new/Keystore> -storepass changeit-storetype pkcs12 -dname "CN=<CENTRAL_FQDN>, OU=<ORGANIZATION_UNIT>, O=<ORGANIZATION>, L=<LOCALITY>, C=<COUNTRY>"
   ```

   Note: If you do not enter a path for generating the new keystore, it is created in the folder where you entered the command, for example **<installation dir>/central/var/security**.

5. Import the self-signed certificate to the Central **key.store** file:

   ```
   keytool -v -importkeystore -srckeystore new/path/created/Keystore -srcstoretype PKCS12 -srcstorepass changeit -destkeystore .key.store -deststoretype JKS -deststorepass changeit
   ```

6. Start Central.

Importing a Certificate to a RAS Truststore

After installing a RAS, if you are using a custom root certificate for Central, you will need to import the trusted root certificate authority (CA) to the RAS **client.truststore**. If you are using a standard signed root certificate you do not have to perform the following procedure as 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 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.15=-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 the hostname.

   For example:

   ```
    wrapper.java.additional.20=-Dssl.verifyHostName=true
   ```

5. Import the trusted root certificate authority (CA) to the RAS **client.truststore** file:

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

6. Start RAS.

Importing a 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 standard signed root certificate you do not have to perform the following procedure as 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 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 the hostname.

   For example:

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

5. Import the trusted root certificate authority (CA) to the Central **client.truststore** file:

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

6. Run OOSH.

Importing a Certificate to 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 standard signed root certificate you do not have to perform the following procedure as 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 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.bat** 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 the hostname.

   For example:

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

5. Import the trusted root certificate authority (CA) to the Studio **client.truststore** file:

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

6. Start Studio.

Changing the keystore/truststore Password

- **To change the Central password**:

  a. Edit the **server.xml** file located in **<Installation dir>/central/tomcat/conf/server.xml.**

  b. Locate the HTTPS connector. For example:

  ```
  keyPass="changeit" keystoreFile="C:/Program Files/Hewlett-Packard/HP Opera
  tions Orchestration/central/var/security/key.store" keystorePass="changeit"
  keystoreType="JKS" maxThreads="200" port="8443" protocol="org.apache.coyot
  e.http11.Http11NioProtocol" scheme="https" secure="true" sslProtocol="TLS"
  truststoreFile="C:/Program Files/Hewlett-Packard/HP Operations Orchestrati
  on/central/var/security/client.truststore" truststorePass="changeit" trust
  storeType="JKS"/>
  ```

  c. Change the required password.

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

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

     - `truststorePass` - the password to access the trust store. 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.

  d. Save the file.

  e. Open central-wrapper.conf, under central/conf and change:

  ```
  wrapper.java.additional.19=-Djavax.net.ssl.trustStorePassword=changeit
  ```

  f. Restart Central.

- **To change the 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 **studio.bat** file and change the `changeit` parameter of the truststore.

Closing 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" po
rt="8080" protocol="org.apache.coyote.http11.Http11NioProtocol" redirectPort
="8443"/>
```

HTTPS connector:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false" compress
ion="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" prot
ocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="tru
e" sslProtocol="TLS" truststoreFile="C:/Program Files/Hewlett-Packard/HP Ope
rations Orchestration/central/var/security/client.truststore" truststorePass
="changeit" truststoreType="JKS"/>
```

By default, both are enabled.

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 connector, and delete or comment out the line.
3. Save the file.
4. Restart Central.

Client Certificate Authentication (Mutual Authentication)

The most common use of X.509 certificate authentication is in verifying the identity of a server when using SSL, 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 SSL with mutual authentication. The server requests a valid certificate from the client as part of the SSL 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.

Importing the CA Root Certificate into Central Truststore

Before you configure the client certificate, make sure you have configured the SSL Server Certificate described in the Server Certificate Authentication section.

Set the `clientAuth` attribute to `true` if you want the SSL stack to require a valid certificate chain from the client before accepting a connection. Set to `want` if you want the SSL 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.

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. Import the appropriate root certificate (CA) into Central **client.truststore**: **<installation dir>/central/var/security/client.truststore**, for example:

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

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

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

4. Add the `crlFile` attribute to define the certificate revocation list file for the SSL 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-----
```

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 Certificates Principal section for more details.

Configuring a Client Certificate in RAS

To configure the client certificate in an external RAS:

1. Open the **ras-wrapper.conf** from <**installation dir>ras/var/conf/ras-wrapper.conf**.

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

   ```
   wrapper.java.additional.17=-Djavax.net.ssl.keyStore=<installation dir>/var/s
   ecurity/certificate.p12"

   wrapper.java.additional.18=-Djavax.net.ssl.keyStorePassword=changeit

   wrapper.java.additional.19=-Djavax.net.ssl.keyStoreType=PKCS12
   ```

Configuring a Client Certificate in Studio Remote Debugger

To configure the client certificate in the Studio Remote Debugger:

1. Edit the **studio.bat** from **<installation dir>/studio**.

2. 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
   ```

3. Edit the **studio.properties** file in **<installation_dir>/studio/conf/Studio.properties**.

   - Set `engine.connection.authenticated` to `false`.

   - Set `engine.connection.host` so that it is the same as the certificate FQDN of the Central certificate.

  - Set `engine.connection.protocol` to `https`.
  - Set `engine.connection.port` to the secured port of Central.

Configuring a Client Certificate in OOSH

1. Edit the **oosh.bat** from **<installation dir>/central/bin**.
2. Change the following according to the your client certificate:

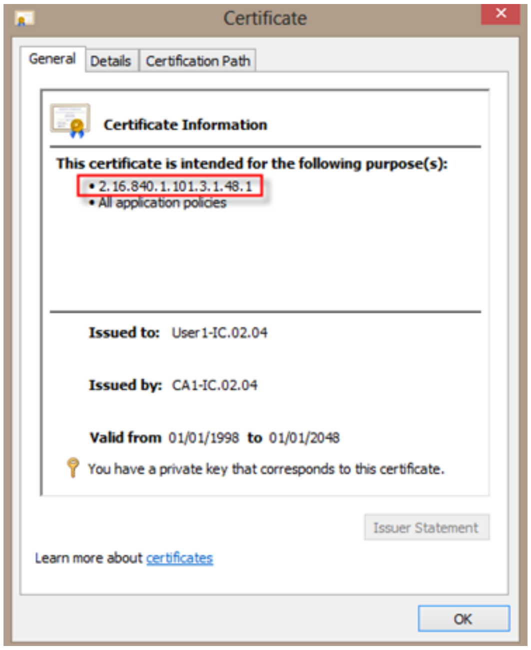
```
-Djavax.net.ssl.keyStore="<installation dir>/var/security/certificate.p12"

-Djavax.net.ssl.keyStorePassword=changeit

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

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

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`.

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-passwo
rd" keystoreType="PKCS12" maxThreads="200" port="8443" protocol="org.apache.coyo
te.http11.Http11NioProtocol" scheme="https" secure="true" sslProtocol="TLS"/>
```

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

Configuring High Availability

In previous versions of HP OO, a clustering solution called Terracotta was provided as part of the application. In HP OO 10.x, this is no longer the case, and you can use any clustering solution that you wish.

For information about how to install the load balancer, see the documentation provided by your load balancer vendor.

Load Balancer Requirements

We recommend to configure the load balancer with two separate virtual IPs for the user interface and for RASes:

- For the HP OO user interface and customer portals, the virtual IP should use a **sticky session** policy. The sticky session ensures that all subsequent requests will be sent to the server who handled the first login request. This means that users will only need to log in to the HP OO interface once.

- For RASes, the virtual IP should use a **round robin** policy, to distribute the load across the different servers.

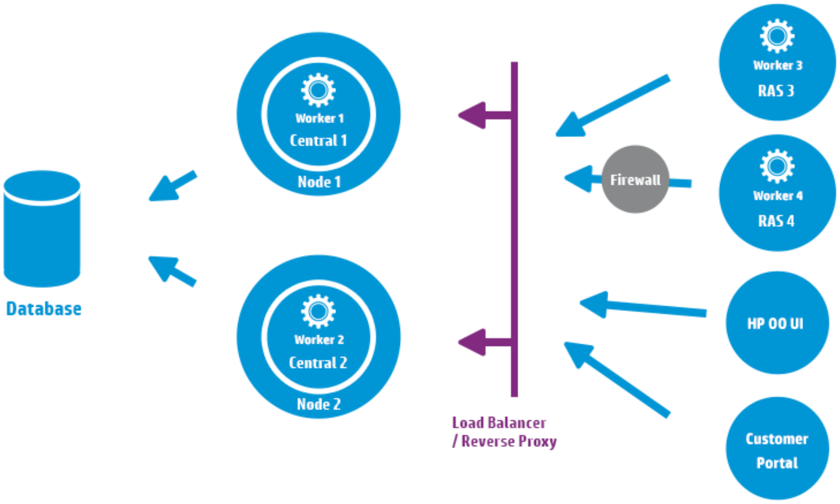
> **Note:** If you have a different configuration that satisfies these requirements, it is okay to use it. For example, if you have a load balancer that supports JSESSION, you can use the JSESSIONID parameter to set up a single virtual IP with a sticky session policy for all sources. Since RAS requests are stateless (no JSESSIONID), this will provide a round robin policy for RASes.

Central uses the following URL to check which server is live: http://<IP>/oo/hello.html

Load Balancer Security

In a hardened high availability environment, the load balancer should be configured for SSL. For information about how to configure SSL, see "Server and Client Certificate Authentication" on page 5.

Communication between the HP OO interface and the load balancer can use HTTPS. We recommend to install the SSL certificate on the load balancer so that this is the termination point for the encryption. Beyond the load balancer, communication will continue in HTTP, at a faster rate.



Configuring the Load Balancer and HP OO Centrals for SSL Offloading

If a load balancer is used to access the Central servers, it is recommended to configure the load balancer for SSL offloading.

1. Edit the Tomcat **server.xml** file, to include the following:

```
<Engine name="Catalina" defaultHost= "localhost" >

. . .

<Valve className="org.apache.catalina.valves.RemoteIpValve"protocolHeader
="X-Forwarded-Proto"   />

. . .

</Engine>
```

2. Configure the load balancer to add a new header to all the clients' requests.

   The header name is configurable and should match the Tomcat configuration specified above. In this example, the name is "X-Forwarded-Proto".

   In the F5 load balancer, the configuration would look like this:

```
when HTTP_REQUEST {
```

```
HTTP::header insert "X-Forwarded-Proto" "https";

}
```

Federal Information Processing Standard (FIPS)

Configuring HP OO for FIPS 140-2 Compliance

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

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: AES
- Hashing algorithm: SHA1

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.

Prerequisite

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 10.02 or higher to be compliant with FIPS 140-2. You cannot configure an upgraded installation of HP OO version 10.01 or an installation of HP OO version 10.01 that is in use.

2. Backup 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 the <oo_jre> is the directory in which the JRE used by HP OO is installed. By default this is**<installation dir>\java**)

3. 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:** Refer to the **ReadMe.txt** file from the downloaded content for information on how to deploy the files and upgrade the JRE used by HP OO.

4. 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**

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 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
- Replace data source password
- 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 10.02 completely replaces the installed JRE files. Therefore, the following steps must be done after upgrading to 10.02.

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 one. For example, change a provider entry from:

```
security.provider.1=sun.security.provider.Sun to security.provider.2=sun.sec
urity.provider.Sun.
```

2. For every provider listed, in the format **security.provider.<nn>=<provider_name>**, increment the preference order number `<nn>` by one. For example, change a provider entry from:

   ```
   security.provider.1=sun.security.provider.Sun to security.provider.2=sun.sec
   urity.provider.Sun.
   ```

3. 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
   ```

4. Add the following entry to ensure **RSA BSAFE** is used in FIPS 140-2 compliant mode:

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

5. As the default DRBG algorithm ECDRBG128 is not safe (according to NIST), set the security property com.rsa.crypto.default to HMACDRBG as follows:

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

6. Exit and save 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. Open **the encryption.properties** file from **<installation dir>\central\var\security\encryption.properties** in a text editor. For example, edit the following file:

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

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

3. Locate `secureHashAlgorithm=SHA1` and replace it with `secureHashAlgorithm= SHA-256`.

4. 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.

5. 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.

> **Note:** AES has three approved key lengths: 128/192/256 by NIST SP800-131A publication.
>
> These secure hash algorithm are supported in FIPS: SHA-1, SHA-256, SHA-384, SHA-512.

Replace the Database Password

Replace the database password described earlier.

Start HP OO

To start the HO operations orchestrations services:

1. On the server that hosts HP OO,, navigate to **Control Panel** > **Administrative Tools** > **Services**.

2. Start the **HP Operations Orchestration Central** service.

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 10.00, you have the option to change the FIPS encryption algorithm.

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

To modify the FIPS encryption algorithm:

1. In the Central UI, go to **System Workspace** >**Topology** >**Workers**.
2. Disable all Central embedded workers.
3. Stop the Central Service.
4. Go to **<Central installation folder>/var/security**.
5. Back up and edit the **encryption.properties** file with the supported algorithms.

6. Back up the **encryption_repository** file.

7. Back up and delete the **credentials.store** file.

   The embedded worker will create a new **credentials.store** file when Central starts.

8. Go to **<Central installation folder>/bin/**.

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

   A new master key is generated.

10. Run the **encrypt-password** script with the **-e -p <password> option**, where **password** is the database password.

11. Copy the result.

    It should look like `${ENCRYPTED}<some_chars>`.

12. Go to **<Central installation folder>/conf**.

13. Open the **database.properties** file and change the **db.password** value to the one that you copied.

14. Start Central.

Changing the RAS Encryption Properties

To change the RAS encryption properties:

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

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

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

3. Save the changes.

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

5. Run **oosh.bat**.

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

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

7. Restart the RAS service.

Configuring LWSSO Settings

When you install HP OO 10.00, if you choose to upgrade the LWSSO settings from HP OO 9.x, these LWSSO settings will be migrated, but LWSSO will be disabled in HP OO 10.00, even if it was previously enabled in HP OO 9.x.

When you enable LWSSO afterward, you may receive warnings under certain scenarios. To clear the warnings from the log, follow the steps below to set the management URL property using the fully qualified domain name.

- When Central and a RAS are installed on the same machine, and the LWSSO settings are enabled, you must set the management URL property using the fully qualified domain name.

  a. Stop the RAS process.

  b. In the **ras/conf/ras-wrapper.conf** file, change

  ```
  wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://localhost:<port>/oo
  ```

  to

  ```
  wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://<FullyQualifiedDomainN
  ame>:<port>/oo
  ```

  c. Start the RAS process.

- When the RAS is installed on a different machine from the Central and LWSSO settings are enabled, you must specify the management URL of the Central with the fully qualified domain name during the RAS installation, instead of the IP address.

- When connecting another application to Central through LWSSO, you must specify the management URL of the Central with the fully qualified domain name.

  a. Stop the Central process.

  b. In the **central/conf/central-wrapper.conf** file, change

  ```
  wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://localhost:<port>/oo
  ```

  to

  ```
  wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://<FullyQualifiedDomainN
  ```

```
ame>:<port>/oo
```

c. Start the Central process.

Configuring the XSS Policy

HP OO has XSS protection with AntiSamy. The default protection policy is slashdot, which allows for `<b>`, `<u>`, `<i>`, `<a>`, and `<blockquote>` without CSS.

This policy is configurable to one of the supported policies by AntiSamy. For more information, see

https://www.owasp.org/index.php/Category:OWASP_AntiSamy_Project#Stage_2_-_Choosing_a_base_policy_file

The policy is configurable via a system configuration property called `xss.policy`. Possible values include: slashdot (the default), ebay, myspace, tinymce, and anythinggoes.

To check which policy is configured, go to **https://host/oo/reports/sysinfo**and look for the parameter `xss.policy` in the **system configuration** section.

The simplest way to change the default Slashdot policy is using the HP Operations Orchestration Shell utility.

1. Double-click the `oosh.bat` batch file, to start the OOSH utility.

2. In the command line, type, for example:

   ```
   ssc --url https://host/oo --key  xss.policy --value anythinggoes
   ```

For more information about the HP Operations Orchestration Shell utility, see the the *HP Operations Orchestration Shell User Guide*.

Configuring Localization

Configuring the Database for Localization

If your HP OO system is localized and you are using MS SQL Server, you need to set the database collation to the relevant collation name, in according to your required language:

- English: `SQL_Latin1_General_CP1_CS_AS`
- Japanese: `Japanese_Unicode_CS_AS`
- Simplified Chinese: `Chinese_Simplified_Stroke_Order_100_CS_AS`
- German: `SQL_Latin1_General_CP1_CS_AS`

- French: `French_100_CS_AS`
- Spanish: `SQL_Latin1_General_CP1_CS_AS`

If you already have a database installed, HP OO creates the tables using the database-specific collation. It is important to note that using other collations can cause characters to appear in gibberish in the user interface for localized installations. In addition, other collations are not officially supported in MS SQL for localized installations.

Setting the System Locale in Central-wrapper.conf

If your HP OO system is localized , you will need to set the following properties to reflect the system locale, in the **central-wrapper.conf** file:

```
set.LANG=

set.LC_ALL=

set.LANGUAGE=

wrapper.java.additional.16=-Duser.language=

wrapper.java.additional.17=-Duser.country=
```

For example, for Japanese: `set.LANG=ja_JP` and `set.LC_ALL=ja_JP`

Configuring the System

Changing the Database Password

1. Stop the Central Service.
2. Run the encrypt-password script with the -e -p <password> option, where password is the database password.
3. Copy the result it should appear as follows:

   ```
   ${ENCRYPTED}<some_chars>.
   ```

4. Go to the folder **<Central installation folder>/conf**, and open the **database.properties** file.
5. Change the `db.password` value to the one that you copied.

Changing the Database IP

This section is relevant when you need to configure HP OO to work with another database instance. All the database parameter such as database credentials, schema name, tables, and so on, should be identical.

1. Edit the file **\HP Operations Orchestration\central\conf\database.properties.**
2. Look for the `jdbc.url` parameter. For example:

   ```
   jdbc.url=jdbc\:jtds\:sqlserver\://16.60.185.109\:1433/schemaName;sendStringP
   arametersAsUnicode\=true
   ```

3. Change the IP address\FQDN of the database server.
4. Save the file.
5. Restart Central.

Adjusting the Logging Levels

It is possible to adjust the granularity of the information that is provided in the log, separately for regular logging, deployment, and execution.

The granularity options are:

- `INFO` - Default logging information
- `DEBUG` - More logging information

- ERROR/WARNING - Less logging information

To adjust the granularity in the logging:

1. Open the **log4j.properties** file (under **/<oo-installation>/central/conf/log4j.properties**).
2. Replace INFO with DEBUG or ERROR/WARNING in the following place in the **log4j.properties** file.

   For example:

   ```
   log.level=INFO
   execution.log.level=DEBUG
   deployment.log.level=DEBUG
   ```

Adjusting the Timing of Quartz Jobs

In the HP OO system, quartz jobs run periodically for maintenance of the system.

Each job runs for a set amount of time, this time is the repeated at intervals. Following are examples of job triggers:

Trigger Name	Current Repeat Interval	What Happens
onRolling:OO_EXECUTION_EVENTS_Trigger	12 hours	Rolling the events table for purging
onRolling:OO_EXECUTION_STATES_Trigger	4.5 minutes	Rolling the states table for purging
queueCleanerTrigger	1 minute	Purging the queue tables
queueRecoveryTrigger	2 minutes	Checks if the system needs recovery
recoveryVersionTrigger	0.5 minute	Version counter to be used for the recovery
splitJoinTrigger	1 second	Joins finished splits

If you want to tweak these jobs timings in order to improve performance, perform the following:

Note: Any change to the timings can have a major affect on the system, consult with your HP service representative before making any changes to these triggers.

1. Enter the Jminixm page using the url: **{OO_HOST}:{OO_PORT}/oo/jminix/**

   > **Note:** You need **Manage System Settings** permission in order to enter **jminix**.

2. Open the OO tab, under **MBeans** there is an operation named **jobTriggersMBean**.
3. Use this operation, enter the values on the right tab, using the name of the trigger you want to change. Use the exact same name as the table, with the new value of repeat interval.

This changes the triggering times of the job.

Java™
COMPATIBLE