Hewlett Packard
Enterprise

Operations Orchestration

Software Version: 10.70
Windows and Linux Operating Systems

Security and Hardening Guide

Document Release Date: April 2017
Software Release Date: November 2016

Legal Notices

Warranty

The only warranties for Hewlett Packard Enterprise 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. Hewlett Packard Enterprise 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 Hewlett Packard Enterprise 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.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

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

Documentation Updates

To check for recent updates or to verify that you are using the most recent edition of a document, go to: https://softwaresupport.hpe.com/.

This site requires that you register for an HP Passport and to sign in. To register for an HP Passport ID, click **Register** on the HPE Software Support site or click **Create an Account** 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 HPE sales representative for details.

Contents

Introduction

Welcome to the *HPE OO Security and Hardening Guide*.

This guide is designed to help IT professionals who deploy and manage HPE Operations Orchestration (HPE OO) instances in a secure manner. Our objective is to help you make well-informed decisions about the various capabilities and features that HPE OO provides to meet modern enterprise security needs.

Security requirements for the enterprise are constantly evolving and this guide should be viewed as HPE's best effort to meet those stringent requirements. If there are additional security requirements that are not covered by this guide, please open a support case with the HPE support team to document them and we will include them in future editions of this guide.

This guide is designed to help IT professionals who deploy and manage HPE Operations Orchestration (HPE OO) instances in a secure manner. Our objective is to help you make well-informed decisions about the various capabilities and features that HPE OO provides to meet modern enterprise security needs.

Security requirements for the enterprise are constantly evolving and this guide should be viewed as HPE's best effort to meet those stringent requirements. If there are additional security requirements that are not covered by this guide, please open a support case with the HPE support team to document them and we will include them in future editions of this guide.

Technical System Landscape

HPE OO is an enterprise-wide application based on Java 2 Enterprise Edition (J2EE) technology. J2EE technology provides a component-based approach to the design, development, assembly, and deployment of enterprise applications.

Security Updates

Between HPE OO 10.20 and 10.50 the following security updates were made:

Between OO version 10.20 and 10.50 the following security updates were made:

- When the **Enable Capture of Logged-in User Credentials** check box is selected in Central, HPE OO will temporarily capture (in a secure manner) the credentials of the logged-in user when this user runs flows in the Remote Debugger. A message warns that credentials may be captured.

- In HPE OO 10.5x, the default is that there is no default role. This gives the administrator better control of user authorization, because users only get roles that are explicitly assigned either to them or to their LDAP group.
- When HPE OO has multiple LDAP configurations, if the administrator flags one of these as default, users who belong to it will not be required to select a domain upon login.
- HPE OO 10.5x secures sensitive data (for example, passwords) during the execution. If a variable is marked as sensitive in Studio, it will be retrieved in an encrypted form when being used in scriptlets.

Between HPE OO 10.10 and 10.20 the following security updates were made:

- It is now possible to grant permissions for system accounts in HPE OO. This enables the administrator to control which users can view which system accounts and run flows that use them. This feature is useful for customers with multiple organizations, who may wish to hide some of the system accounts from some users.

  For more information, see "Content Management Enhancements - Apply Permissions to Multiple Roles" in the *HPE OO10.20 Release Notes*.
- It is now possible to apply permissions to multiple roles in the Edit Permissions dialog box. In previous versions, it was only possible to select one role at a time.

  For more information, see "Content Management Enhancements - Permissions for System Accounts" in the *HPE OO10.20 Release Notes*.
- When you upgrade an HPE OOinstallation from an earlier 10.x version, the SSL truststore is updated to include the up-to-date trusted root certificates, as published by Oracle. This includes deletion of expired certificates, and import of new ones.

  For more information, see "Installation Enhancements - Updated Trusted Root Certificates" in the *HPE OO10.20 Release Notes*.
- HPE OOnow offers the option to audit events, so that you can track security breaches. Auditing lets you track actions that took place on Central, such as logins, triggering flows, creating schedules, editing configurations, and so on.

  Currently, you can retrieve the audit trail only via APIs. For more information, see the *HPE OOAPI Guide*.
- HPE OOnow supports encryption keys that are 2048 bits long (and longer). This aligns our cryptography keys with the FIPS 186-4 standard.
- A new sslEnabledProtocols property has been added to the **server.xml** file (located at **<installation_folder>/central/tomcat/conf/server.xml**):

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

This property ensures that only TLS v1, TLS v1.1 and TLS v1.2 are allowed and that SSL 3.0 is not. This prevents vulnerability to the “POODLE” attack (Padding Oracle On Downgraded Legacy Encryption).

Related Documents

For more information about the security hardening of HPE OO, see the following documents:

- *HPE OONetwork Architecture White Paper*

For more information about HPE OO, see the following documents:

- *HPE OOConcepts Guide*
- *HPE OOAdministrator Guide*
- *HPE OOArchitecture Guide*
- *HPE OODatabase Guide*
- *HPE OOCentral User Guide*
- *HPE OOStudio Authoring Guide*
- *HPE OORelease Notes*
- *HPE OOInstallation, Upgrade, and Configuration Guide*
- *HPE OOSystem Requirements*
- *HPE OOStudio Wizards User Guide*

These and other documents can be found on HPE Live Network (https://hpln.HPE.COM/node/21/otherfiles#).

These and other documents can be found on HPE Live Network (https://hpln.HPE.COM/node/21/otherfiles#).

These and other documents can be found on HPE Live Network (https://hpln.HPE.COM/node/21/otherfiles#).

Security Overview

This section provides an overview of the security models and recommendations for a secure implementation of HPE OO. This includes subjects such as authentication, authorization, encryption, and more. Where relevant, there are references to other HPE OO documents, which describe how to complete security-related tasks.

Security Concepts

HPE OO Glossary

For more information about HPE OO concepts, see the *HPE OO Concepts Guide*.

Role Permission

A permission is a predefined authorization to perform a task. HPE OO Central includes a set of permissions that can be assigned to roles.

For example, the **Schedule** permission grants the ability to view and create run schedules.

Role

A role is a collection of permissions.

For example, the **Flow Administrator** role may be assigned the **View Schedules** permission and the **Manage Schedules** permission.

User

A user is an object associated with a person (or application identity) representing the person and defining their authorization.

Roles are assigned to users, to define the actions they are authorized to perform in Central. For example, the user Joe Smith may be assigned the **Flow Administrator** role.

It is possible to configure different kinds of users:

- **LDAP users** log on to Central using their LDAP user name and password. For example, using their Active Directory user name and password.

- **Internal users** log on to Central using the user name and password that was set up locally in Central.
- **LWSSO** - HPE Lightweight Single Sign On (SSO) is a mechanism in which a single action of user authentication and authorization can permit a user to access all HPE systems that support LWSSO. For example, if users have logged onto another HPE product web client that has LWSSO enabled, they can enter the HPE OO Central application directly, bypassing the HPE OO Central logon screen.

When an internal user and an LDAP user with the same role are logged in, there is no difference between their permissions.

> **Note:** It is recommended to use LDAP users rather than internal users, because LDAP users are secured according to policies implemented by the LDAP provider.

Content Permission

Content permission is permission to view or run individual flows or the flows in a particular folder.

Users who have been assigned a specified role will be able to access the flows according to the content permissions assigned to their role.

For example, users with the **Administrator** role may be entitled to view and run all the flows in the system, while users with the **User** role may be entitled to run certain flows, and have view permission for others.

Common Security Concepts

System Security

The processes and mechanisms by which computer-based equipment, information, and services are protected from unintended or unauthorized access, change, or damage.

Least Privilege

The practice of limiting access to the minimal level that will allow normal functioning. This means giving a user account only those privileges that are essential to that user's work.

Authentication

The process of identifying an individual, usually based on a user name and password, or certificate.

Authorization

Permission to access system objects, based on an individual's identity.

Encryption

A way to enhance the security of a message or file by scrambling the contents so that it can be read only by someone who has the right encryption key to encode it. For example, the TLS protocol encrypts the communication data.

Countermeasure

A way to mitigate the risk of a threat.

Defense in Depth

Layers of protection, so that you don't have to rely on a single security measure alone.

Risk

A possible event that could cause damage. For example, financial loss, damage to the company image, and so on.

Threat

Triggering a risk event that exploits a vulnerability.

Vulnerability

A weakness in a target that can potentially be exploited by a security threat.

Secure Implementation and Deployment

Default Security Settings

In many cases, it is recommended to modify the default security settings that are provided out-of-the-box.

- **Authentication** – By default, authentication is not enabled in Central. It is recommended to enable it, as soon as users have been set up. For more information, see "Enabling Authentication" in the *HPE OO Central User Guide*.
- **Auditing** – By default, auditing is not enabled in Central. It is recommended to enable it. For more information, see "Enabling Auditing" in the *HPE OO Central User Guide*.
- **TLS Encryption** – By default,HPE OO supports three TLS protocols: 1.0, 1.1, 1.2. It is recommended to work with the latest version. For more information, see "Configuring the TLS Protocol" on page 82.
- **TLS Server Certificate** – By default, the user is asked to provide a CA certificate during the installation of the HPE OO server.
- **Client Certificate** – By default, Client Certificate is not enabled. It is recommended to work with Client Certificate to authenticate to Central. For more information, see "Configuring Client Certificate Authentication in Central" on page 58.
- **KeyStore, TrustStore, and Server Certificate Passwords** – By default, Java passwords are provided for the keyStore, trustStore, and Server Certificate. It is recommended to replace these with encrypted passwords. For more information, see Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password.
- **RC4 Cipher** – By default, the RC4 cipher is enabled. It is recommended to disable the RC4 cipher on the JRE level. For more information, see Removing the RC4 Cipher from the SSL-supported Ciphers.
- **Security Banner** – By default, the security banner is not enabled in Central. It is recommended to enable it, with your custom message. For more information, see "Setting Up a Security Banner" in the HPE OO *Central User Guide*.

- **Windows Authentication of the Database** – By default, Windows authentication is not enabled in Central. If you are working in the Windows and SQL server environment, it is recommended to configure HPE OO to work with Windows authentication. See "Configuring HPE OO to Work with Windows Authentication" in the *HPE OO Database Guide*.
- **Default Algorithms** – The **encryption.properties** file contains the default algorithms. If you want to be compliant with FIPS, see "Configuring HPE OO for FIPS 140-2 Level 1 Compliance" on page 72. For more information about the FIPS 140-2 Level 1 defaults, see "Encryption Administration" in "Encryption" on page 27.
- **Java Policy** – By default, the **java.policy** file is not hardened. For information about how to modify the **java.policy** file, see "Preventing Flows from Accessing the Central/RAS Local File System" on page 84.

HPE OO Security Hardening

The "Hardening" chapter provides recommendations for safeguarding your HPE 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.

To comprehensively protect your HPE OO system, it is necessary to secure both HPE OO and the computing environment (for example, the infrastructure and the operating system) upon which the application runs.

The "Hardening" chapter provides recommendations to help secure HPE 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.

Physical Security

HPE Software recommends that HPE OO is protected by physical security controls defined by your organization. The HPE OO server components are installed in a physically secured environment, according to best practice. For example, the server must be in a closed room with access control.

Secure Installation Guidelines

Supported Operating Systems

For the types and versions of supported operating systems, see the *HPE OO System Requirements*.

Operating System Hardening Recommendations

Contact your operating system vendor for recommended best practices for hardening your operating system.

For example:

- Patches should be installed
- Unnecessary services/software should be removed or disabled
- Minimal permissions should be assigned to users
- Auditing should be enabled

Tomcat Hardening

When you install HPE OO Central, Tomcat is partially hardened by default. If you want extra hardening, see the recommendations in the Hardening chapter.

Installation Permissions

The following permissions are required to install and run HPE OO:

Installing	Windows/Linux: Any standard user who is able to run a Java process, and who has

HPE OO	permission to create folders and services
Running HPE OO	• Windows: The Windows service runs as the system user or a specific user (the user must have access to the HPE OO installation directory) • Linux: Any standard user who is able to run a Java process

See also the recommendations in the CIS Apache Tomcat documentation.

Network and Communication Security

The *HPE OO Architecture Guide* describes the basic HPE OO topology, high availability, and load balancer security.

The *HPE OO Network Architecture White Paper* describes the required firewall configuration, and suggests two workarounds that can be applied in cases when, due to policy restrictions, the required firewall configuration cannot be implemented:

- SSH reverse tunneling
- Reverse proxy

Communication Channel Security

Supported Protocols and Configuration

HPE OO supports the TLS protocol.

For more information, see Replacing the Central TLS Server Certificate.

The Central ports are defined by the administrator during the installation.

Channel Security

HPE OO supports the following secure channels:

Channel (Directed)	Supported Secure Protocol
OOSH, browser, Studio Remote Debugger, or RAS → Central	For a secure channel, use the TLS communication for encryption and Client Certificate for authentication.
Central → LDAP Server	To encrypt communication between Central and LDAP, use Secure LDAP, using the TLS protocol.

RAS Security

In a topology with a reverse RAS (which waits for Central to initiate the connection), the following mechanism protects the security of the RAS:

- If there are several consecutive failed connection attempts (because the wrong shared secret was entered), this will result in a delay.

For more information about reverse RASes, see “Setting Up Topology – Workers and RASes” in the *HPE OO Central User Guide*.

Administration Interface Security

Accessing the Administration Interface

There are several ways to control access to the administration interface:

- Credentials
- Client certificate
- SAML

Securing the Administration Interface - Recommendations

1. Authentication must be enabled in Central.

   See "Enabling Authentication" in the *HPE OO Central User Guide*

2. It is recommended to secure the administration interface with the TLS protocol. You should set up TLS between the client and the Central interface for encryption.

   See "Working with Server and Client Certificates" on page 43.

3. It is recommended to work with LDAP users, rather than internal users, because this is more secure.

4. It is recommended to set up authentication to access Central via Client Certificates. This is more secure than user passwords.

   See "Working with Server and Client Certificates" on page 43.

User Management and Authentication

Authentication Model

To enable easy bootstrapping of the authentication mechanism in HPE OO, the product starts with authentication disabled.

You must enable authentication immediately after installation.

For information about how to enable authentication, see "Enabling Authentication" in the *HPE OO Central User Guide*.

There are a number of ways to authenticate access to Central.

Choose the method of identifying users:

- User name and password
- Client Certificate
- SAML token
- Single sign on (HPE LWSSO)

Choose one of two ways to manage the users:

- LDAP users , saved on an LDAP server as Active Directory (recommended)
- Internal users and passwords, saved locally on the Central server (not recommended)

Types of Users

Different types of users can have different permissions assigned for them. For example, flow author, administrator, system administrator, and so on.

For more examples of different types of users, requiring different permissions, see "Major Personas" in the *HPE OO Concepts Guide*.

Authentication Administration and Configuration

Internal or LDAP Users

You can set up internal users with passwords in the Central UI or define the user in the LDAP server and map LDAP groups to Central roles.

> **Note:** Our recommendation is not to use internal users, but to use a more secure alternative such as LDAP users.

For information about configuring internal users, see "Setting Up Security – Internal Users" in the *HPE OO Central User Guide*.

For information about mapping LDAP groups to Central roles, see "Setting Up Security – LDAP Authentication" in the *HPE OO Central User Guide* and "LDAP Configuration" in the *HPE OO API Guide*.

SAML / Client Certificates / LW SSO

For information about configuring Central to work with SAML, see "Setting Up Security – SAML" in the *HPE OO Central User Guide*.

For information about configuring Central to work with Client Certificates, see "Working with Server and Client Certificates" on page 43.

For information about configuring Central to work with LW SSO, see "Setting Up Security – LWSSO" in the *HPE OOCentral User Guide*, "Configuring LWSSO Settings" in the *HPE OO Administration Guide*, and "LW SSO" in the *HPE OO API Guide*

Database Authentication

HPE OO supports four databases: Oracle, MS SQL, MySQL, and Postgres.

We recommend using a strong database password for database authentication and using a strong password policy. For example, blocking after a number of failed attempts.

When using MS SQL, it is possible to work with either database authentication or with OS authentication. Our recommendation is to work with OS authentication, where this is possible. For example, it is possible to use Windows authentication to access Microsoft SQL Server databases.

- For information about setting up OS authentication, see "Configuring HPE OO to Work with Windows Authentication " in the *HPE OO Database Guide*.
- See "Changing the Database Password" in the *HPE OO Administration Guide*.
- See the best practices recommended by the database vendor (if these exist).

Authorization

Authorization Model

User access to HPE OO resources is authorized based on the user's role, and the permissions configured for that role.

See:

- "Setting Up Security – Roles" in the *HPE OO Central User Guide*
- "Assigning Permissions to a System Account" in the *HPE OO Central User Guide*

Minimal Permissions Guidelines

It is recommended to:

- Select appropriate permissions for the role.
- Use minimal permissions when creating new roles.
- Grant minimal permissions and extend the permissions only as needed to avoid unwanted privilege escalation. For example, start with View permissions and add additional permissions individually as needed.

Authorization Configuration

Central is installed with a number of out-of-the box roles, which you can configure and assign to users. By default, the out-of-the box roles are assigned the following permissions:

Role	Default Permissions
Administrator	All
End_user	None
Everybody	None
Promoter	All the **Content** permissions
System_admin	All the **System** permissions

Default Role

It is possible to configure one of the roles with the **Default Role** attribute. If you do so, make sure that this is the role with the least privileges. Remember that when you give permissions to this role, this affects all LDAP users, in addition to those are explicitly associated with the role.

For more information, see "Assign a role to be the default role" under "Setting Up Security – Roles" in the *HPE OO Central User Guide*.

See also:

- "Assigning Permissions to a System Account" in the *HPE OO Central User Guide*
- "Setting Permission for Content" in the *HPE OO Central User Guide*

Access to Workspaces in Studio

When creating multiple workspaces in Studio, we recommend creating a workspace under folders to which only the creating user has read and write permissions.

Workspaces created under public folders might be accessible to all users, making them prone to tampering and disclosure of sensitive information.

Backup

In order to prevent data loss, It is highly recommended to regularly back up your data on the servers onto secure media. This is also helpful for disaster recovery and business continuity.

After installing HPE OO, make sure to back up the **central\var\security** folder and the **central\conf\database.properties** file.

Some data on the database schema is encrypted and the keys for decryption are stored locally on the HPE OO Central server. If these system files become corrupted or deleted, the schema will be useless, because there will be no way to decrypt the data.

> **Note:** The keys are encrypted, so it is important to include them in the backup. The keys are located in the **security** folder.

See:

- "Backing Up HPE OO" in the *HPE OO Administration Guide*
- "Setting up Disaster Recovery" in the *HPE OO Administration Guide*
- "Backing Up and Recovering the Central Security Files" in the *HPE OO Installation, Upgrade, and Configuration Guide*
- "Using a Load Balancer in HPE OO Deployment" in the *HPE OO Architecture Guide*

Encryption

Encryption Model

HPE OO supports encryption and hash algorithms to protect sensitive data. Encryption is designed to prevent the exposure and modification of sensitive data, such as passwords, definitions, and so on, in HPE OO.

It is important to use well-known, standard algorithms without known vulnerabilities, in order to prevent decryption by unauthorized persons.

For example, SSL is not used, because of known vulnerabilities in the SSL protocol.

Static Data

All saved passwords are protected using well known algorithms and none appear in clear text.

For example:

- The system account passwords are encrypted.
- The internal user passwords are hashed.
- The database passwords are encrypted.

Data in-transit

HPE OO uses the Transport Layer Security (TLS) protocol to encrypt the data between components (such as Central and RAS).

Disabling the HTTP port

It is recommended to disable the HTTP port, for security reasons, so that the only communication channel will be on TLS and encrypted. For more information, Changing the HTTP/HTTPS Ports or Disabling the HTTP Port.

Encryption Administration

Recommended Encryption Best Practices

In order to reach higher levels of security and cryptography, it is recommended to configure HPE OO to be compliant with Federal Information Processing Standards (FIPS) 140-2. HPE OO can be set to be compliant with FIPS 140-2 Level 1.

Default Configuration Set

- Symmetric-key algorithm: AES with key size 128
- Hashing algorithm: SHA1

Advanced Settings

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

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

See "Configuring Compliance with FIPS 140-2" on page 75.

Digital Certificates

A digital certificate is an electronic "passport" for a person, server, station, and so on.

- To use encryption between a browser and the Central server, you need to install a digital certificate on the server side.
- To use Client Certificate to authenticate the Central server, you need to install a Client Certificate on the client side (for example, on the browser, RAS, OOSH, Studio, and so on).

HPE OO uses the Java Keytool utility to manage cryptographic keys and trusted certificates. This utility is included in the HPE OO installation folder, in **<installation dir>/java/bin/keytool**.

Certificate Location

Installations of HPE OO Central include two files for the management of certificates using Keytool:

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

Access Control to KeyStore and TrustStore

It is recommended that the TrustStore and KeyStore are stored with read permissions only for the user that runs the Central service.

Replacing the HPE OO Self-signed Certificate

It is recommended to replace the HPE OO self-signed certificate after a new installation of HPE OO or if your current certificate has expired.

Part of the process of replacing the certificate is generating a PKCS12 format certificate, using your CA. Contact your CA for specific details about the certificate process or refer to your corporate policy.

For more information, see Replacing the Central TLS Server Certificate.

Adding Digital Signatures to a Content Pack

If a content pack has a digital signature from a trusted CA, this provides assurance that the content is trusted.

Adding a digital signature is not mandatory.

- HPE OO out-of-the-box content packs contain a digital signature from Verisign.
- HPE OO authors are recommended to add a digital signature to their custom content packs.
- If a signed content pack is breached, the content pack cannot be deployed.
- If the signature is expired, a warning appears before deployment, and the user must select a check box acknowledging that they are ignoring the expired signature.

Pay attention to content packs that are not signed. An unsigned content pack is not trusted, and could contain malicious content. Note also that unsigned content pack could be breached and with the signature removed.

For more information about digital certification of content packs, see "Deploying and Managing Content Packs" in the *HPE OO Central User Guide*.

Sensitive Information in a Content Pack

System Account Passwords

Do not include passwords when creating a content pack. The passwords will be obfuscated inside the content pack, which is not a secure option.

The HPE OO security best practice is to configure the system account passwords in Central. For more information, see "Setting Up System Accounts for a Content Pack" in the *HPE OO Central User Guide*.

Auditing and Log Files

Auditing

Auditing lets you track actions that took place on the Central server, such as logins, triggering flows, creating schedules, editing configurations, and so on. The audit data enables you to track user activity on the Central system, tracking who did what action, when. For example, auditing will show that a user ran a flow, updated a configuration, deleted a schedule, or failed authentication.

The audit data is saved in the database. For more information, see "Auditing" in the *HPE OO API Guide*.

Logs

Logs let you trace errors, warnings, information, and debugging messages.

The logs are saved in the file server, in the following locations:

- Central - **<oo-installation>/central/var/logs**
- Studio - **<user>/.oo/logs**
- RAS - **<oo-installation>/ras/var/logs**.

No Sensitive Data Kept in Audit Records and Log Files

No sensitive data is kept in the audit records or in the log files in HPE OO.

Getting Audit Records

You can get the audit records via API or via a query to the OO_AUDIT table. For more information, see "Auditing" in the HPE OO *API Guide*.

Example of audit data:

```
[
{
“time”:1412312016740, “type”:”AuditConfigurationChange”,
“group”:”AuditManagement”, “subject”:” mydomain\myuser2”, “outcome”:”Success”,
“data”:”{"enabled":false}”
},
{
```

“time”:1412312016722, “type”:”InternalUserDelete”, “group”:”Authentication-Authorization”, “subject”:”mydomain\myuser2”, “outcome”:”Success”, “data”:{"usersNames":["admin"]}"

}

]

APIs and Interfaces

API and Interface Model

You can work with the HPE OO public Application Programming Interfaces (APIs) instead of via the HPE OO Central UI, to perform the same actions. Some actions can only be performed via APIs, such as purging and auditing. The public API is HTTP-based. All APIs are RESTful and use JavaScript Object Notation.

Features and Administration of the API and Interface Security Configuration

It is important to work securely with the APIs. Use the security mechanisms mentioned in this guide (authentication, encryption, and so on) while working with the APIs.

The API interface can work on HTTP or HTTPS.

> **Note:** When you use our APIs to display HTML, it is your responsibility to protect it from XSS attacks.

For more information, see the following chapters in the *HPE OO API Guide*:

- "LDAP Configuration"
- "Users"
- "LW SSO Configuration"
- "Authentication"
- "Roles"

Security Questions and Answers

How can I generate a certificate request that can be signed by an external CA?

Export the certificate request and send it to the external CA for signing. For instructions, see Replacing the Central TLS Server Certificate.

Which TCP/UDP ports does HPE OO use? What is the direction, user, and encryption?

When you install HPE OO, you need to configure at least one available port for the Central Server in the HTTP/HTTPS fields. The default provided values are 8080 and 8443, but you can change these. For more information about secure channels between Central and the other components, see "Network and Communication Security" on page 18

Where and how are the credentials stored (admin accounts, integration users)?

See "User Management and Authentication" on page 21.

How do I configure self-signed SSL certificates for Central/RAS/Studio?

During the installation of HPE OO, if you do not provide a certificate, a self-signed certificate is created by default. However, it is not recommended to use self-signed certificates, for security reasons. HPE recommends working with a certificate from custom-root CA or from a well-known CA.

For more information about configuring certificates for HPE OO, see "Encrypting the Communication Using a Server Certificate" on page 44.

How do I enable or disable any kind of auditing?

By default, auditing is not enabled. For details on how to enable it, see "Enabling Auditing" in the *HPE OO Central User Guide*. For more information about auditing, see "Auditing and Log Files" on page 32.

How much detail is in the logs, and how do I change the amount of logging?

The logs can be set to different levels of granularity. The default level is INFO, but you can adjust this. For details, see "Adjusting the Logging Levels" in the *HPE OO Administration Guide*.

For more information about log files, see "Auditing and Log Files" on page 32.

How is sensitive information encrypted?

See "Encryption" on page 27.

Is the communication between Central and RAS encrypted?

If you use HTTPS, it is encrypted.

Is the communication between HPE OO and other integration components (HPNA, CSA, AD, and so on) encrypted?

This depends on the integration that you are using. If you use HTTPS, it is encrypted.

How can I restrict access to the Flow Library, based on the user roles?

See "Setting Up Security – Roles" in the *HPE OO Central User Guide*.

What authentication mechanism does HPE OO support?

The supported authentication mechanisms are LDAP, SAML and internal users. HPE OO also supports Client Certificate and LWSSO. See "User Management and Authentication" on page 21.

Is HPE OO FIPS 140-2 compliant?

Yes. For more information, see "Configuring Compliance with FIPS 140-2" on page 75.

What are the authentication methods between Central and RAS?

User password or Client Certificate.

Are all passwords stored encrypted or hashed?

Yes. All saved passwords are protected using well known algorithms and none are left in cleartext.

Can I limit the Central user IP address?

No, this is not supported at the moment.

Is HPE OO certified for common criteria?

This is in progress. We are currently "in evaluation". For details, see https://www.cse-cst.gc.ca/en/canadian-common-criteria-scheme/publication/list/evaluation-product.

When I use OOSH, can I pass sensitive data to Central?

Our recommendation is to use a secure channel when connecting to Central. See "Network and Communication Security" on page 18.

Hardening HPE OO

This section describes how to configure security hardening for HPE OO.

> **Note:** For information about other administrative tasks, see the *HPE OO Installation, Upgrade, and Configuration Guide*.

Security Hardening Recommendations

1. Install the latest version of HPE OO. For more information, see the *HPE OO Installation, Upgrade, and Configuration Guide*.
2. (Optional) Configure HPE OO for FIPS 140-2 Compliance. If you choose to do this, it must be configured before you start the Central server. See "Configuring HPE OO for FIPS 140-2 Level 1 Compliance" on page 72.
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 "Working with Server and Client Certificates" on page 43.
4. Harden the HPE OO Central server by removing the HTTP port and replacing the passwords of the KeyStore and TrustStore with strong passwords. See Changing or Closing the HTTP/HTTPS Ports and Changing the KeyStore/TrustStore Password.
5. Harden HPE OO Studio by replacing the KeyStore and TrustStore passwords with strong passwords, and encrypt or obfuscate the passwords in the configuration files. See Changing the KeyStore/TrustStore Password.
6. Remove the RC4 cipher from the SSL-supported ciphers. See Removing the RC4 Cipher from the SSL-supported Ciphers.
7. (Optional) Configure the TLS protocol version. See "Configuring the TLS Protocol" on page 82.
8. Enable authentication in Central. See "Enabling Authentication" in the *HPE 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 *HPE 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 *HPE OO Central User Guide*.

11. In the Windows and SQL server environment, configure HPE OO to work with Windows authentication. See "Configuring HPE OO to Work with Windows Authentication" in the *HPE OO Database Guide*.
12. Make sure that auditing is enabled in Central. For more information, see "Enabling Auditing" in the the *HPE OO Central User Guide*.

Default Security Settings

In many cases, it is recommended to modify the default security settings that are provided out-of-the-box.

- **Authentication** – By default, authentication is not enabled in Central. It is recommended to enable it, as soon as users have been set up. For more information, see "Enabling Authentication" in the *HPE OO Central User Guide*.
- **Auditing** – By default, auditing is not enabled in Central. It is recommended to enable it. For more information, see "Enabling Auditing" in the *HPE OO Central User Guide*.
- **TLS Encryption** – By default,HPE OO supports three TLS protocols: 1.0, 1.1, 1.2. It is recommended to work with the latest version. For more information, see "Configuring the TLS Protocol" on page 82.
- **TLS Server Certificate** – By default, the user is asked to provide a CA certificate during the installation of the HPE OO server.
- **Client Certificate** – By default, Client Certificate is not enabled. It is recommended to work with Client Certificate to authenticate to Central. For more information, see "Configuring Client Certificate Authentication in Central" on page 58.
- **KeyStore, TrustStore, and Server Certificate Passwords** – By default, Java passwords are provided for the keyStore, trustStore, and Server Certificate. It is recommended to replace these with encrypted passwords. For more information, see Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password.
- **RC4 Cipher** – By default, the RC4 cipher is enabled. It is recommended to disable the RC4 cipher on the JRE level. For more information, see Removing the RC4 Cipher from the SSL-supported Ciphers.

- **Security Banner** – By default, the security banner is not enabled in Central. It is recommended to enable it, with your custom message. For more information, see "Setting Up a Security Banner" in the HPE OO *Central User Guide*.
- **Windows Authentication of the Database** – By default, Windows authentication is not enabled in Central. If you are working in the Windows and SQL server environment, it is recommended to configure HPE OO to work with Windows authentication. See "Configuring HPE OO to Work with Windows Authentication" in the *HPE OO Database Guide*.
- **Default Algorithms** – The **encryption.properties** file contains the default algorithms. If you want to be compliant with FIPS, see "Configuring HPE OO for FIPS 140-2 Level 1 Compliance" on page 72. For more information about the FIPS 140-2 Level 1 defaults, see "Encryption Administration" in "Encryption" on page 27.
- **Java Policy** – By default, the **java.policy** file is not hardened. For information about how to modify the **java.policy** file, see "Preventing Flows from Accessing the Central/RAS Local File System" on page 84.

Working with Server and Client Certificates

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.

HPE OO uses the Keytool utility to manage cryptographic keys and trusted certificates. This utility is included in the HPE 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 HPE 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 HPE OO certificate (private key).

> **Note:** If you use client certificates with LDAP, the LDAP must be configured as default in Central. For details, see "Setting Up Security – LDAP Authentication" in the *HPE OO Central User Guide*.

Recommendations:

- It is recommended to replace the HPE OO self-signed certificate after a new installation of HPE OO or if your current certificate is expired.
- It is recommended to store the TrustStore and KeyStore with read permissions only for the user that runs the Central service.
- It is recommended to clear the console after using Keytool or to use the prompt for password inputs.

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**.
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 <keystore password>
   ```

4. If you already have a certificate with a **.pfx** or **.p12** extension, go to the next step. If not, 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:**

> To generate the PKCS12 format certificate, you need to use your CA. As this step may vary according to CA vendor and policy, contact your CA for a detailed explanation of the certificate generation process.

> **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.

   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.

8. Start Central.

Importing a CA Root Certificate to the Central TrustStore

If you are using a custom root certificate for Central, you will need to import the trusted root certificate authority (CA) to the **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, HPE 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**or **<oosh dir>/var/security/client.truststore** for a standalone OOSH.
2. 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). In a standalone OOSH, it is under the main OOSH directory.

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

3. 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, HPE 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 the 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 make sure that the `-Dssl.support-self-signed` value is set 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 make sure that the `-Dssl.verifyHostName` is set 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
   ```

   > **Note:** This property is set to **true** by default.

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 <path to the client.truststore> -file <certificate_name.cer> -storepass <changeit>
   ```

6. Start the RAS.

Importing a CA Root Certificate to the Studio TrustStore

If you are using custom certificates in the Central, SVN, or GIT servers, in order for Studio to be able to work with these, you will need to import the trusted root certificate authority (CA) to the Studio

client.truststore file. 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, HPE 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.

For a fresh **.oo** folder, Studio copies the **client.truststore** file from **<installation.dir>/studio/var/security** to the **<user>/.oo** folder. This is a one time action, in order to ensure that Studio can automatically import certificates (for example, for the Studio Remote Debugger). Studio will use this file as the **client.truststore** if it exists; otherwise, it will use the one from the Studio installation (**<installation.dir>/studio/var/security/client.truststore**).

After an upgrade to 10.5x or later, the truststore location is the **<user>/.oo** folder.

If you want to manually import a certificate, you can import it either to **.oo/client.truststore** or to **client.truststore** in the Studio installation folder.

If you are using multiple workspaces, the changes made to the **client.trustore** file located under the **.oo** folder will apply only to the specific workspace. In order to apply the change to all the newly created workspaces, edit the **client.trustore** file located in the Studio installation folder.

> **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 **<user>/.oo**

   For example, **C:/Users/<username>/.oo**

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

3. Make sure that the `-Dssl.support-self-signed` value is set to **false**. This enables the trusted root certificate authority (CA).

   For example:

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

4. Make sure that `-Dssl.verifyHostName` is set 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). Replace the parameters that are highlighted in <yellow>:

```
keytool -importcert -alias <any_alias> -keystore <path to the
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*.

Checking the Revoke Status of a Certificate

A certificate revocation list (CRL) is a list of certificates (or more specifically, a list of serial numbers for certificates) that have been revoked. Entities that present these (revoked) certificates should no longer be trusted.

From the RAS Side

The RAS can determine when a Central certificate has been revoked in the remote server. When the RAS is started and performs a handshake with Central, it retrieves the Central certificate that contains a link to the location of the CRL file. The RAS then goes to the CRL file and validates the Central certificate against this CRL file.

If a certificate has been revoked, there is no connection between the RAS and the Central. In addition, there will be error messages in the log file on the RAS side.

From the Central Side

On the Central side, the RAS appears with Availability Offline (not connected) in the Topology area.

What do you want to do?

Enable Revoke Status Check

1. Open the RAS wrapper file **ras-wrapper.conf** located under **<RAS_INSTALLATION>/ras\conf**.
2. Add the following lines to the RAS:

```
wrapper.java.additional.<n>=-Dcom.sun.security.enableCRLDP=true
wrapper.java.additional.<n>=-Dcom.sun.net.ssl.checkRevocation=true
```

3. Set the following flag to false:

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

Changing and Encrypting/Obfuscating 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 *HPE OO Administration Guide*.

From OOSH, run the following command:

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

where `<keyName>` is one of 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.

   Use the following keytool command to change the KeyStore password:

   ```
   keytool -storepasswd -keystore <installation_
   folder>/central/var/security/key.store
   ```

   Use the following keytool command to change the server certificate private key entry password:

   ```
   keytool –keypasswd -alias tomcat -keystore <installation_
   folder>/central/var/security/key.store
   ```

   Use the following keytool command to change the TrustStore password:

   ```
   keytool -storepasswd -keystore <installation_
   folder>/central/var/security/client.truststore
   ```

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"/>
   ```

   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.

b. Save the file.

5. Edit the **central-wrapper.conf** file located in **<installation dir> central\conf\central** and replace the password of the TrustStore with new password in encrypted or obfuscated form. Examples:

   ```
   wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword={ENCRYPTED}
   <encrypted_password>
   ```

   ```
   wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword={OBFUSCATED}
   <obfuscated_password>
   ```

   For information about how to encrypt or obfuscate a password, see Encrypting and Obfuscating Passwords.

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 password of the TrustStore.
- **To change the OOSH TrustStore password**: Edit the **oosh.bat** file and change the password of the TrustStore.
- **To change the Studio TrustStore password**: Add the property **client.truststore.password** with the password in obfuscated format to the **Studio.properties** file in the **<user>/.oo** folder.

  ```
  client.truststore.password={OBFUSCATED}6L9+NqBjKYp5heuvMEzg0g==
  ```

  If this property is not defined, Studio will fall back to the system property **javax.net.ssl.trustStorePassword** for the TrustStore password.

For information about how to obfuscate a password, see Encrypting and Obfuscating Passwords.

Encrypting and Obfuscating Passwords

You can encrypt or obfuscate a password using the `encrypt-password` script, which is located in **<installation_folder>/central/bin**.

Our recommendation is to use encryption.

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.

Encrypting Passwords

1. Locate the `encrypt-password` script, in **<installation_folder>/central/bin**.
2. Run the script with the `-e -p <password>` option, where `password` is the password you want to encrypt.

   > **Note:** You can either use `-p` as the flag to encrypt the password or `--password`.

   The encrypted password should appear as follows:

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

Obfuscating Passwords

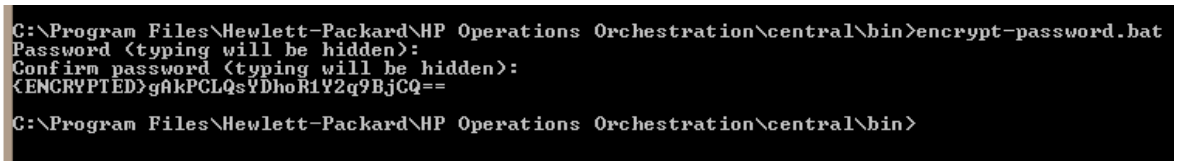
1. Locate the `encrypt-password` script, in **<installation_folder>/central/bin**.
2. Run the script with the `-o <password>` option, where `password` is the password that you want to obfuscate.

   The obfuscated password should appear as follows:

   ```
   {OBFUSCATED}<some_chars>.
   ```

Creating a Prompt for the Password

It is recommended to run the `encrypt-password` script without providing the `-p` argument. For example:



This will create a prompt for the hidden password inputs.

Removing Vulnerable Ciphers from the SSL supported Ciphers

The DES and Triple DES ciphers, as used in the TLS protocol have a birthday bound of approximately four billion blocks, which makes it easier for remote attackers to obtain cleartext data via a birthday attack against a long-duration encrypted session, as demonstrated by an HTTPS session using Triple DES in CBC mode, aka a "Sweet32" attack.

For more information about this attack, see https://sweet32.info/.

To disable RC4, DES and Triple DES ciphers in Operations Orchestration:

1. Open the **$JRE_HOME/lib/security/java.security** file.
2. Remove the comments and change the parameters according to the example below:

   ```
   jdk.certpath.disabledAlgorithms=DES, DESede, RC4, MD2, RSA keySize < 1024
   jdk.tls.disabledAlgorithms=DES, DESede, RC4, MD5, DSA, RSA keySize < 1024
   ```

3. Restart the OO Central server.

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

> After upgrading from an earlier version of HPE OO 10.x, repeat these steps.

Changing the HTTP/HTTPS Ports or Disabling the HTTP Port

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 the HTTP Port

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

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. 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 <path to the 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 the HTTPS Connector

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:\HPE\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:/HPE/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 "Working with Server and Client Certificates" on page 43.

> **Note:** By default, the user certificate is matched against internal users of OO Central. If you want to match the certificate against LDAP users, make sure you set the desired LDAP as default LDAP.

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

Set the `clientAuth` attribute to `want` if you want the TLS stack to request a client certificate. When the attribute is set to want and:

- The certificate is not presented, then the authentication does not fail and the user is redirected to the login page.
- An invalid certificate is presented, then the authentication fails and the user is not redirected to the login page.

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`.

   For example:

   ```
   <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" server="OO" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
   sslProtocol="TLSv1.2" truststoreFile="C:/Program Files/Hewlett-Packard/HP
   Operations Orchestration/central/var/security/client.truststore"
   truststorePass="changeit" truststoreType="JKS"/>
   ```

   > **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.
   > - The OO 9x backwards compatible SOAP/REST APIs are not supported in cases where Client authentication is required.

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. Edit the **central-wrapper.conf** file, located in **<installation dir> central\conf\central**.

   Uncomment the following properties and set the client certificate location and password to a client certificate with an administrator user.

   ```
   #wrapper.java.additional.23=-Djavax.net.ssl.keyStore="%CENTRAL_
   HOME%/var/security/certificate.p12"

   #wrapper.java.additional.24.stripquotes=TRUE

   #wrapper.java.additional.25=-Djavax.net.ssl.keyStorePassword={OBFUSCATED}
   ZUoMreNLw6qIOyzX7g5YKw==

   #wrapper.java.additional.26=-Djavax.net.ssl.keyStoreType=PKCS12
   ```

   For information about how to encrypt or obfuscate a password, see Encrypting and Obfuscating Passwords.

7. 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 HPE 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.

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 Certificate to a RAS TrustStore.

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 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={OBFUSCATED}
   <obfuscated_password>
   ```

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

4. Start the RAS server.

Important Notes! 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 *OO Central User Guide*.

In HPE OO 10.20 and later, the `keyStorePassword` parameter 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. See Encrypting and Obfuscating Passwords.

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 Certificate to Studio Debugger TrustStore.

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 your client certificate:

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

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

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

4. Start Studio.

Notes:

- InHPE OO 10.20 and later, the `keyStorePassword` parameter 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. See Encrypting and Obfuscating Passwords.
- 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 HPE 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 HPE 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 Certificate to the OOSH TrustStore.

1. Stop OOSH.
2. Edit the **oosh.bat** from **<installation dir>/central/bin** (in a standalone OOSH, it is under the main OOSH directory).
3. Change the following according to your client certificate:

   ```
   -Djavax.net.ssl.keyStore="<installation dir>/var/security/certificate.p12"
   -Djavax.net.ssl.keyStorePassword={OBFUSCATED}<obfuscated_password>
   -Djavax.net.ssl.keyStoreType=PKCS12
   ```

4. Start OOSH.

> **Note:**
>
> In HPE OO 10.20 and later, the `keyStorePassword` parameter 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. See Encrypting and Obfuscating Passwords.
>
> 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 HPE 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 HPE OO internal domain.

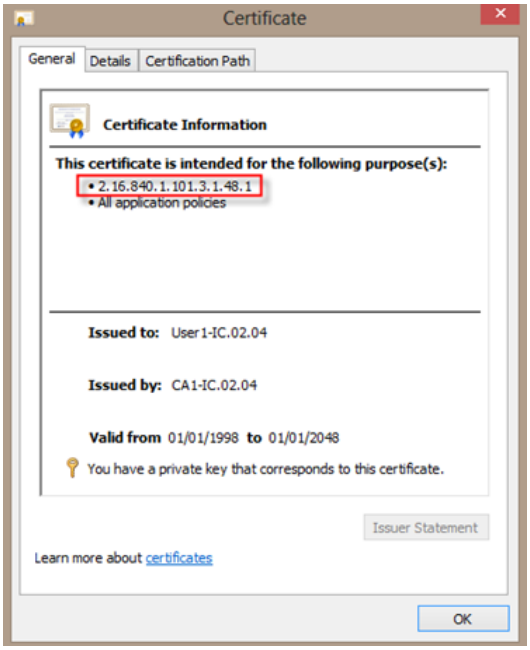
Processing Certificate Policies

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

- You can set the purpose string in the certificate.
- HPE 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).



For more information about how to change OO system properties, see the *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 HPE OO (LDAP or internal user).
- To change the regular expression, change the configuration item: `x509.subject.principal.regex`.

Enabling OO to Read from the Subject Alternative Name Field in a Certificate

You can enable OO to read from the `Subject Alternative Name` field in a certificate, using the configuration item `x509.principal.lookup.field`.

This configuration item controls which certificate field is used to extract the username.

Possible values are:

- `subjectDN` - represents the `Subject` field of the certificate, meaning that OO keeps its default behavior and attempts to extract the username from the **Subject** field. This is the default value.
- `subjectAltNames.otherName.principalName` - represents the `User Principal Name` (OID 1.3.6.1.4.1.311.20.2.3) contained in the `Other Name` entry of the `Subject Alternative Names` certificate extension. For CAC Authentication, it might be required to use the value of the User Principal Name, so you would use this value.

For more information about how to change HPE OO configuration items, see the *HPE OO Shell (OOSH) User Guide*.

Signing OO Content Packs

Code signing is a mechanism whereby publishers of software and content use a certificate-based digital signature to verify their identities to users of the code, thus allowing users to decide whether or not to install it based on whether they trust the publisher.

It is important that Operations Orchestration content packs are code signed with your digital signature for the following security benefits:

- Operations Orchestration administrators can identify your company as the author, publisher or distributor of the content pack, at the time of deploying content packs to Operations Orchestration Central.
- It provides integrity of the content pack, that is it protects the content packs from being tampered with (if the content pack is changed, the digital signature is invalidated).

A digital certificate is an electronic document issued by a Certificate Authority (CA).

It contains the public key for a digital signature and specifies the identity associated with the key, such as the name of an organization.

The certificate is used to confirm that the public key belongs to the specific organization. The CA acts as the guarantor.

Digital certificates must be issued by a trusted authority and are only valid for a specified time. They are required in order to create a digital signature.

A typical option to store the digital signature file is inside a PCKS#12 file that has .pfx extension, for example **digital_signature.pfx**.

See "Working with Content Packs" in the *Studio Authoring Guide* to learn more about content packs.

What do you want to do?

Sign content packs using the jarsigner utility

1. Download Java SE 8u102 from here:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

2. Install it to a suitable location on your local file system and locate your jarsigner utility at the following location <JDK_INSTALLTION>/bin.
3. Use the digital signature to sign the content pack file .jar.

Convert the digital signature file from PCKS#12 format to JKS format used by the jarsigner utility

1. Using the keytool utility located in **<oo_installation>/java/bin/keytool**, run the following command:

```
keytool -importkeystore -srckeystore
<digital_signature.pfx> -srcstoretype pkcs12 -srcalias
<certificate_alias_pfx> -destkeystore
<digital_signature.jks> -deststoretype jks -deststorepass
<password_jks> -destalias <certificate_alias_jks>
```

where:

Name	Description
certificate_alias_pfx	The alias inside the source digital file in pkcs12 format identifying the private key to be used to sign the content pack JAR file, and the key's associated certificate
digital_signature.jks	The name of the output digital file in JKS format
deststorepass	The password for the output digital file in JKS format
certificate_alias_jks	The alias inside the output digital file in JKS format identifying the private key to be used to sign the content pack JAR file, and the key's associated certificate

2. Then, run the following command:

```
jarsigner -keystore <digital_signature.jks>
-signedjar <signed-content-pack.jar> <content_pack.jar>
 <certificate_alias_jks>
```

where:

Name	Description

digital_signature.jks	The name of the output digital file in JKS format
signed-content-pack.jar	The output signed content pack file JAR
content_pack.jar	The JAR file to be used as input for signing
certificate_alias_jks	The alias inside the output digital file in JKS format identifying the private key to be used to sign the content pack JAR file, and the key's associated certificate.

Verifying the Integrity of the Studio Tools JARs

There are JAR files located under the <OO_installation_folder>/studio/tools/lib folder that provide the executable code for OOSHA, wizards and hpln-index-generator tools.

In order to verify the integrity of the JAR files, to ensure that the JARs have not been tampered with since the Operations Orchestration installation, you can verify their digital signature.

A digital certificate is an electronic document issued by a Certificate Authority (CA).

It contains the public key for a digital signature and specifies the identity associated with the key, such as the name of an organization.

The certificate is used to confirm that the public key belongs to the specific organization. The CA acts as the guarantor.

Digital certificates must be issued by a trusted authority and are only valid for a specified time. They are required in order to create a digital signature.

See the *HPE OO Studio Wizards Guide* for more information.

What do you want to do?

Verify the digital signature of a Studio tools JAR

1. Download Java SE 8u102 from here:

   http://www.oracle.com/technetwork/java/javase/downloads/index.html

2. Install it in a suitable location on your local file system and locate your jarsigner utility under:

   <JDK_INSTALLATION>/bin

3. Run the following command from the command line

   ```
   jarsigner -verify -keystore
   <oo_installation>/central/var/security/internal.truststore
   ```

```
-strict -verbose -certs <jar_file>
```

where <jar_file> is any one of the following jars:

- hpln-index-generator.jar
- oosha-jar-with-dependencies.jar
- ps-wizard-jar-with-dependencies.jar
- rest-wizard-jar-with-dependencies.jar
- shell-wizard-jar-with-dependencies.jar
- third-party-cp-wizard-jar-with-dependencies.jar
- ws-wizard-jar-with-dependencies.jar

The output of the command is expected to end in "jar verified". Additionally, all JAR file entries listed in the output must be accompanied by the "smk" label.

Configuring HPE OO for FIPS 140-2 Level 1 Compliance

The section below explains how to configure HPE OO 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 HPE OO for FIPS 140-2 compliance, HPE OO uses the following security algorithm:

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

HPE OO uses the security provider: RSA BSAFE Crypto software version 6.2.1. This is the only supported security provider for FIPS 140-2.

> **Note:** Once you have configured HPE OO to be compliant with FIPS 140-2, you can only revert back to the standard configuration if you re-install HPE OO.

Prerequisites

Note for upgraders:

If you are upgrading from an installation of HPE OO 10.10 (and later) that was already configured with FIPS, see Prerequisite Steps for Upgraders.

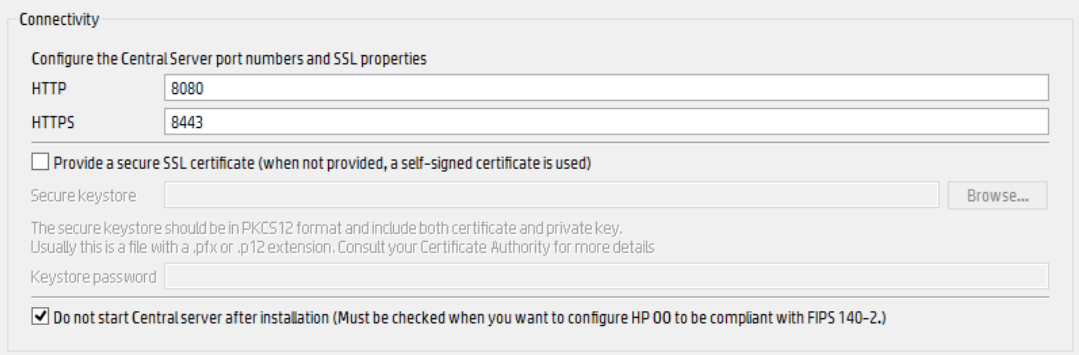
Before configuring HPE OO to be compliant with FIPS 140-2, perform the following steps:

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

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

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

2. Verify that when HPE 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.



3. Back up the following directories:
   - **<installation dir>\central\tomcat\webapps\oo.war**
   - **<installation dir>\central\tomcat\webapps\PAS.war**
   - **<installation dir>\central\conf**
   - **<installation dir>\java** (the entire **java** folder should be backed-up)
4. Download **Server Oracle JRE 8** from http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.html, and replace the **OpenJDK (Zulu) JRE** with the **Server Oracle JRE**.
   a. Delete everything inside the **<installation dir>\java** folder.
   b. Extract the downloaded archive.
   c. Copy the **JRE** folder content to **<installation dir>\java**.
5. 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/server-jre8-downloads-2133154.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 HPE OO.

6. Install the RSA BSAFE Crypto software files. On the system on which HPE OO is installed, copy

the following to **<oo_jre>\lib\ext** (where **<oo_jre>** is the directory in which the JRE used by HPE OO is installed. By default, this is **<installation dir\java**).

- **<installation dir>\central\lib\cryptojce-6.2.1.jar**
- **<installation dir>\central\lib\cryptojcommon-6.2.1.jar**
- **<installation dir>\central\lib\jcmFIPS-6.2.1.jar**

Prerequisite Steps for Upgraders

1. Download the Server Oracle JRE 8 and replace the OpenJDK (Zulu) JRE with the Server Oracle JRE.
   a. Delete everything inside the **<upgrade dir>\JAVA folder**.
   b. Extract the downloaded archive.
   c. Copy the **JRE** folder content to **<upgrade dir>\JAVA**.

   http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.html
2. 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/jce8-download-2133166.html

   See the **ReadMe.txt** file from the downloaded content for information on how to deploy the files and upgrade the JRE used by HPE OO.
3. Install the RSA BSAFE Crypto software files. On the system on which HPE OO is installed, copy the following files to **<oo_jre>\lib\ext**:

   (where **<oo_jre>** is the directory in which the JRE that is used by the HPE OO upgrader. By default, this is **<upgrade dir\java**).
   - <installation dir>\central\lib\cryptojce-6.2.1.jar
   - <installation dir>\central\lib\cryptojcommon-6.2.1.jar
   - <installation dir>\central\lib\jcmFIPS-6.2.1.jar

Next, follow the steps in the "Configure the Properties in the Java Security File" section in "Configuring Compliance with FIPS 140-2" on the next page.

Configuring Compliance with FIPS 140-2

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

1. Configure the Properties in the Java Security File.
2. Configure the encryption.properties File and Enable FIPS Mode.
3. Create FIPS-CompliantHPE OO Encryption.
4. Re-encrypt the database password with the new encryption.
5. Start HPE OO.

Step 1: 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 HPE OO 10.x completely replaces the installed JRE files. Therefore, if you are upgrading to 10.x, you must complete the following steps.

> **Note:** If you are upgrading from an installation of HPE OO 10.10 and later that was already configured with FIPS, you must follow the "Prerequisite Steps for Upgraders" section in "Configuring HPE OO for FIPS 140-2 Level 1 Compliance" on page 72, and then follow the steps here, where **<oo_jre>** is the JRE included in the upgrade (in the location **<upgrade dir>\JAVA**).
>
> Make sure to make all the changes in the **java** folder inside the extracted **upgrade** folder.

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.

Step 2: Configure the encryption.properties File and Enable FIPS Mode

The HPE 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.

Step 3: Create FIPS-Compliant Encryption

To create or replace the HPE OO encryption store file, so that it is FIPS-compliant, see "Replacing the FIPS Encryption" on page 79.

> **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

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

> **Note:** If you work with Client Certificate, the certificate should be generated with the FIPS-compliant RSA JCE provider, and with the secure hash algorithms that are supported in FIPS, as listed above.

Step 4: Re-encrypt the database password with the new encryption

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

Step 5: Start HPE OO

Replacing the FIPS Encryption

HPE 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 HPE OO, 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

Use the **generate-keys.bat/sh** file to replace the FIPS encryption key in the encryption repository.

> **Note:** This process backs up the **encryption_repository** file, so you must have the relevant write permissions.

1. Go to **<Central installation folder>/var/security**.
2. Back up the **encryption_repository** file and delete it from the **<Central installation folder>/var/security** folder.
3. Go to **<Central installation folder>/bin**.
4. Run the **generate-keys** script.
5. Press the **Y** key to proceed.

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

> **Note:** If you prefer to run the **generate-keys** script without pausing for the user to type **Y** or **N**, use the silent mode flag **–s** when running the script.

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 HPE OO for FIPS 140-2 Level 1 Compliance" on page 72.
2. Complete all the steps in the "Configure the Properties in the Java Security File" in "Configuring Compliance with FIPS 140-2" on page 75.
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 Compliance with FIPS 140-2" on page 75.
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 HPE OO to define the supported TLS protocol version. By default, HPE 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 **<installation_folder>/central/tomcat/conf/server.xml** file .
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.

Customizing Access to Networks from Inside Java Operations

In order to allow operations code access to a network via sockets you must add SocketPermission rules to the **<installation_folder>/java/lib/security/java.policy** file.

Examples:

```
grant codebase "file:${oo.home}/var/cache/-" {
permission java.net.SocketPermission "example.com:7777", "connect,accept";
permission java.net.SocketPermission "localhost:1024-", "connect";
...
}
```

First rule allows Java operations code (from `${oo.home}/var/cache/`) to connect to port 7777 on `example.com` domain and accept connections on that port.

Second rule allows Java operations code (from `${oo.home}/var/cache/`) to connect to any port between 1024 and 65535 on the localhost.

It is recommended to selectively grant access to networks via specific IP addresses for specific ports using specific actions (connect, listen, accept, resolve) and not set generic rules that allow more access than necessary.

For more information about how to use socket permission rules, see http://download.java.net/jdk7/archive/b123/docs/api/java/net/SocketPermission.html.

Preventing Flows from Accessing the Central/RAS Local File System

You should modify the wrapper configuration and java.policy files of Central or RAS, in order to prevent flows from accessing the local file system of Central or RAS, and gaining access to sensitive resources.

> **Note:** In order to exploit this scenario, a user would need to have both deployment and triggering permissions, in addition to entitlement on flows or the ability to entitle flows. Users with such permissions are likely to be trustworthy users.

To protect from this scenario:

1. In the wrapper configuration file of Central or RAS (**<installation_folder>/<ras/central>/conf/<central/ras>-wrapper.conf**), add the `wrapper.java.additional.<nn>` parameter as follows:

   `wrapper.java.additional.<nn>=-Djava.security.manager`

   Replace <nn> with the number after the last number.

2. In the **java.policy** file (located at **<installation_folder>/java/lib/security/java.policy**), add the following. This allows access to the minimum resources that are required by HPE OO and prevents access to the local file system of Central/RAS that contains sensitive data.

```
grant codebase "file:${oo.home}/bin/-" {
      permission java.security.AllPermission;
  };
grant codebase "file:${oo.home}/lib/-" {
      permission java.security.AllPermission;
  };
grant codebase "file:${oo.home}/tomcat/-" {
      permission java.security.AllPermission;
  };

grant codebase "file:${oo.home}/var/cache/-" {
   permission java.lang.RuntimePermission "getClassLoader";
   permission java.io.FilePermission "${oo.home}/var/cache/-",
     "read, write";
   permission java.io.FilePermission "${oo.home}/var/logs",
     "read, write";
```

```
};
```

Note: You may need to add more permissions depending on the content you run.

To allow the flow to access to resources in the local file system of Central/RAS, you should specify this in the java.policy. For example:

```
grant codebase "file:${oo.home}/var/cache/-" {
                permission java.io.FilePermission
"C:\\users\\cathy\\foo.bat", "read, write, execute, delete";
                permission java.io.FilePermission "C:\\users\\cathy\\-",
"read,write,execute,delete"; // Recursive Example
                permission java.io.FilePermission "C:\\users\\cathy\\*",
"read,write,execute,delete"; // Flat Example
                   .....
};
```

Adding Java Security Manager

To avoid a possible scenario where user creates a flow which shuts down the localhost server, apply the specifications from Preventing Flows from Accessing the Central/RAS Local File System.

The following sections describe how to set up the Java policy file for standalone RAS and Central embedded worker.

Configuring RAS to Add Java Security Manager

To activate security manager on a standalone RAS:

1. In the **%RAS_HOME% \conf** folder, create a text file named `ras.policy`, add the following content:

```
grant codeBase "file:${oo.home}/lib/score-worker-manager-impl.jar"
{
permission java.lang.RuntimePermission "exitVM.75";
}
```

2. In the **ras-wrapper.conf** file located in the **%RAS_HOME%\conf** folder, add the following wrapper:

```
wrapper.java.additional.<next_index>=-Djava.security.policy="%RAS_
HOME%/conf/ras.policy"
```

3. Restart the RAS.

Configuring Central Embedded Worker to Add Java Security Manager

To activate security manager on a Central embedded worker:

1. Add the following content before **System Code Permissions** in the `%CENTRAL_HOME%/tomcat/conf/catalina.policy` file.

```
grantcodeBase "file:${oo.home}/tomcat/webapps/oo/WEB-
INF/lib/score-worker-manager-impl.jar"
{
permission java.lang.RuntimePermission "exitVM.75";
}
```

Hewlett Packard
Enterprise