Operations Orchestration

Version : 2018.09
PDF Generated on : 12 Dec 2018

MICRO FOCUS

Table of Contents

Administer 1
Initial administration tasks 2
Configure and enable authentication 3
Set the system locale in central-wrapper 4
Manage OO licenses 5
Change the MS SQL collation language or the central-wrapper.conf language for content 8
Adjusting the logging levels 9
Configure LW-SSO settings 11
Change the service name 13
Change the timeout limit for the OO web interface 14
Schedule database purging 15
Capture the credentials of the logged-in user 17
Specify communication protocols 18
Initial administration tasks for OO Workflow Designer 19
Displaying system information 20
Configure a proxy for OO Workflow Designer 21
Recurring administration tasks 22
Encrypt and obfuscate passwords 23
Changing the database password 24
Change the database IP 25
Change the hostname IP 26
Adjust the timing of Quartz jobs 27
Reverse the direction of the RAS - Central communication 28
Change the URL of a load balancer/Central on the RAS side after its installation 29
Back up and recover Operations Orchestration 30
Set up disaster recovery 31
System tuning 32
Database maintenance 33
Recurring administration tasks for OO Workflow Designer 34
Changing the database password 35
Change the database IP 36
Change the hostname IP 37
Encrypting passwords 38
Set the system localization 39
Change the language in wrapper file 40
Create a Git repository 41
Adjust the logging levels 42
Change the service name and description 43
Changing the Timeout Limit for the OO Workflow Designer Web Interface 44
Configure content pack import size 45
Back up OO Workflow Designer 46
Set up disaster recovery 47

System tuning 48
Database maintenance 49
Set up database environment 50
Prepare the database environment 51
Deploy and maintain a Microsoft SQL server database 56
Workflow for Microsoft SQL Server Deployment 57
System Requirements for Microsoft SQL Server 58
Configure SQL server 61
Manually create an OO database on Microsoft SQL server 63
Microsoft SQL server database maintenance 65
Deploy and maintain an Oracle database 71
Workflow for Oracle deployment 72
System requirements for Oracle 73
Configure an Oracle database 76
Manually create an Operations Orchestration database on an Oracle instance 78
Oracle database maintenance 80
Deploy and maintain a MySQL database 84
Workflow for MySQL deployment 85
System requirements for MySQL 86
Configure MySQL 88
Manually create an Operations Orchestration database on MySQL 90
MySQL database maintenance 93
Deploy and maintain a Postgres database 95
Workflow for Postgres deployment 96
System Requirements for Postgres 97
Configuring Postgres 99
Manually create an Operations Orchestration database on Postgres 100
Postgres database maintenance 102
Database purging 105
Additional guidelines for Microsoft SQL server 107
Additional guidelines for Oracle 110
Installation Wizard 'Other database' option 113
Set up database environment for OO Workflow Designer 118
Prepare the database environment 119
Use database clusters 121
Database security 122
Database sizing 123
Hardware requirements 124
Deploy and maintain a Microsoft SQL server database 125
Workflow for Microsoft SQL Server Deployment 126
System Requirements for Microsoft SQL Server 127
Configure SQL server 129
Manually create an OO database on Microsoft SQL server 131
Microsoft SQL server database maintenance 133

Additional guidelines for Microsoft SQL server 136
Deploy and maintain an Oracle database 138
Workflow for Oracle deployment 139
System requirements for Oracle 140
Configure an Oracle database 142
Manually create an Operations Orchestration database on an Oracle instance 144
Oracle database maintenance 146
Additional guidelines for Oracle 147
Deploy and maintain a MySQL database 150
Workflow for MySQL deployment 151
System requirements for MySQL 152
Configure MySQL 154
Manually create an Operations Orchestration database on MySQL 156
MySQL database maintenance 158
Deploy and maintain a Postgres database 160
Workflow for Postgres deployment 161
System Requirements for Postgres 162
Configuring Postgres 164
Manually create an Operations Orchestration database on Postgres 165
Postgres database maintenance 167
Installation Wizard 'Other database' option 169
Set up security and hardening 172
Introduction 173
Security overview 175
Security concepts 176
Secure implementation and deployment 178
Secure installation guidelines 179
Network and communication security 181
Administration interface security 183
User management and authentication 184
Authorization 186
Configure OO Central in IDM mode 188
Configure LDAP in IDM mode 192
Backup 194
Encryption 195
Hardening Operations Orchestration 198
Hardening recommendation 199
Working with server and client certificates 201
Encrypting the communication using a server certificate 202
Replacing the Central TLS server certificate 203
Import a CA root certificate to the Central TrustStore 205
Import a CA root certificate to a RAS TrustStore 206
Import a CA root certificate to the OOSH TrustStore 207
Import a CA root certificate to the Studio TrustStore 208

Check the revoke status of a certificate 210
Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password 211
Encrypt and obfuscate passwords 214
Remove vulnerable Ciphers from the SSL supported Ciphers 216
Change or disable the HTTP/HTTPS ports 217
Troubleshoot the HTTPS connector 219
Client certificate manual authentication 220
Configure client certificate authentication in Central 221
Signing OO content packs 227
Verify the integrity of the Studio tools JARs 229
Configure Operations Orchestration for FIPS 140-2 level 1 compliance 230
Configure compliance with FIPS 140-2 233
Replace the FIPS encryption 236
Configure the TLS protocol 238
Prevent flows from accessing the Central/RAS local file system 239
Adding Java security manager 241
Enable redirection to HTTPS 243
Disable CSRF protection 244
Configure the database for localization 245
Setting the system locale in Central-wrapper.conf 246
Set up security and hardening for OO Workflow Designer 247
Security overview 248
Security concepts 249
Secure implementation and deployment 250
Secure installation guidelines 252
Security protection in OO Workflow Designer 254
Network and communication security 256
User management and authentication 257
Backup 259
Encryption 260
Log files 263
Security questions and answers 264
Hardening Operations Orchestration 265
Security hardening recommendations 266
Working with server and client certificates 268
Encrypt the communication using a server certificate 269
Replace the OO Workflow Designer TLS server certificate 270
Import a CA root certificate to the OO Workflow Designer TrustStore 272
Change and encrypt obfuscate the KeyStore TrustStore password 273
Remove the RC4 cipher from the SSL-supported ciphers 275
Change or disable the HTTP/HTTPS ports 276
Configure the secured properties 278
Set SSL configuration properties 279
Troubleshoot the HTTPS connector 280

Troubleshoot multiple deployment requests 281
Configure OO Workflow Designer for FIPS 140-2 level 1 compliance 282
Configure compliance with FIPS 140-2 284
Replace the FIPS encryption| 287

Administer

This section documents relevant information regarding the following Operations Orchestration (OO) configuration tasks.

- Completing advanced configuration tasks.
- Configure four different types of database, to be used with OO.
- Deploying and managing OO instances in a secure manner, and configuring security hardening for OO.

Initial administration tasks
Recurring administration tasks
Set up database environment
Set up security and hardening

Initial administration tasks

Initial administration tasks

This section includes administration tasks that you will generally perform once, while setting up Operations Orchestration after installation.

Configure and enable authentication
Set the system locale in central-wrapper
Manage OO licenses
Change the MS SQL collation language or the central-wrapper.conf language for content
Adjusting the logging levels
Configure LW-SSO settings
Change the service name
Change the timeout limit for the OO web interface
Schedule database purging
Specify communication protocols

Configure and enable authentication

Configure and enable authentication
After installing Operations Orchestration, you must configure the authentication mechanism and enable authentication.

Configure authentication

Select the authentication mechanism to be used in Operations Orchestration, for example, LDAP, SSO, internal users, etc.

Enable authentication

This is done in Central by marking the **Enable Authentication** check box.
Note The **Enable Authentication** check box is only available if there are existing internal or LDAP users with permission to disable the authentication in the future.

Related topics

Setting up the system configuration
Enable authentication

Set the system locale in central-wrapper

Set the system locale in central-wrapper.conf

If your Operations Orchestration (OO) system is localized, you will need to set the properties to reflect the system locale in the **central-wrapper.conf** file.

1. Open the **central-wrapper.conf** file in a text editor (located under <**installation_folder>/central/conf**).
2. Edit the following properties:set.LANG= set.LC_ALL= set.LANGUAGE= wrapper.java.additional.<x>=-Duser.language= wrapper.java.additional.<x>=-Duser.country= For example, for Japanese: `set.LANG=ja_JP set.LC_ALL=ja_JP`
3. Save the **central-wrapper.conf** file.

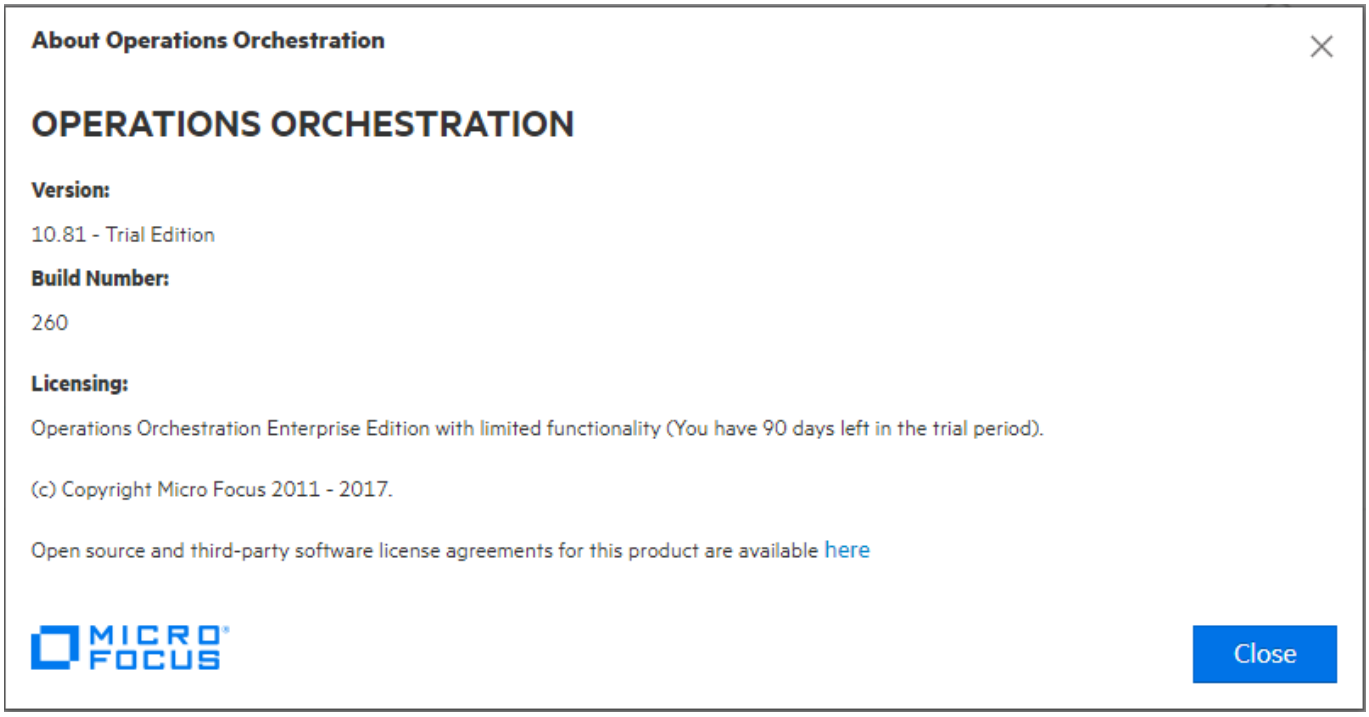
Manage OO licenses

When Operations Orchestration (OO) is first installed, this installs the Trial version license. You will need to install an Enterprise Edition license within 90 days.
OO offers the following options for licensing:

- **OO Enterprise Edition Trial**: A temporary license that runs for 90 days, and includes all out-of-the-box content packs. After the trial license is expired, you will need to purchase the OO Enterprise Edition license to use the content packs. This the default license that comes with the OO installer.
- **OO Enterprise Edition**: A full license, unlimited in time, which includes all out-of-the-box content packs. This license unlocks the OO Community Edition and OO Enterprise Edition Trial licenses.
- **OO Community Edition**: A partial license, which includes the Base content pack and the CE content packs, and enables 500 runs per month.
- **OO Enterprise Edition With Limited Functionality**: A partial license that limits all new functionalities that were introduced in OO version 10.80 and later. All features introduced in previous versions of OO are accessible as under OO Enterprise Edition License.

View license information

You can view the license information by clicking the **Info** button in the top right corner in Central, to display the **About** pop-up window. This window displays the current status of the license. If the license is limited in time or number of runs, it shows how much is left.



Install a new license

To install a new license, got to the Central **License** tab under **System Configuration** > **System Settings**.

Purchase a license

Option 1
Download the license from the OO web site at:
https://www.microfocus.com/licensing
Option 2
Issue a license using the License Management system:

- For a standalone Central installation, issue the license with the IP address of the Central server.
- If you have CSA and OO installed on the same machine, issue one license for that IP address.
- If you have CSA and OO installed on separate machines, issue two licenses for the two IP addresses.
- For a cluster, choose one of the nodes and issue a license for the IP address of this node. Note Make sure to open the Central UI of the selected node (and not via the Load Balancer IP) when you install the license.

- If you have CSA and an OO cluster, issue two licenses: one for CSA and one for a selected node. Note Make sure to open the Central UI of the selected node (and not via the Load Balancer IP) when you install the license for OO.

For more information about licensing details, contact your Sales team or Account Manager.

License a cluster

For a cluster, make sure that you issue the license on one of the cluster IPs (and not on the load balancer IP). Apply it directly on the specific node. This sets the license for the entire cluster.

Change the MS SQL collation language or the central-wrapper.conf language for content

When Operations Orchestration (OO) Central was installed, there was an option to select a supported language in addition to English. This language is used as the MS SQL collation language, if relevant, and for content. This language support is required if, for example, you need to ping a server that is configured in Japanese. You can change this language by editing the **central-wrapper.conf** file.

1. Open the **central-wrapper.conf** file in a text editor (located under <**installation_folder**>**/central/conf**).
2. Edit the language property. For example, change `wrapper.lang=en_US` to `wrapper.lang=ja_JP`.
3. Save the **central-wrapper.conf** file.

Related topics

Language support

Adjusting the logging levels

Because logging activity can slow the performance of Operations Orchestration (OO) and create very large log files, it is important for OO to run with appropriate logging levels. The default logging level is set to INFO. This level provides the necessary information without impacting performance.
In addition, you can adjust the granularity of the information in the log, separately for regular logging, deployment, and execution.
The granularity options are:

- INFO - Default logging information
- DEBUG - More logging information
- WARN - Less logging information
- ERROR - Less logging information

To adjust the granularity in the logging:

1. Open the **log4j.properties** file (under **/<installation_folder>/central/conf/log4j.properties**).

Replace INFO with DEBUG, WARN or ERROR in the relevant places in the **log4j.properties** file.
For example:

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

Note For Ticket: 5309109246The Tomcat server keeps access logs under **/<installation_folder>/central/var/logs**. To prevent these log files from growing in size, it is recommended that you monitor the size of the files named **localhost_access_log.yyyy-mm-dd.txt**, and delete old files as necessary. Use established best practices for your operating system, whether Windows or Linux.
For example - To delete files older than 10 days:
Unix

```
find <OO_HOME>/central/var/logs/ -name " localhost_access*.txt"
-type f -mtime +10 -exec rm -f {} \;
```

Windows

```
C:\Users\Administrator>forfiles /p "<OO_HOME>\central\var\logs" /s /m
localhost_access*.txt /d -10 /c "cmd /c del @PATH"
```

where <OO_HOME> refers to the location of the OO installation folder

You can set this up using a schedule in OO or use Cron or Windows Scheduler to run these commands the background.

Configure LW-SSO settings

When you install OO 10.10 or later, if you choose to upgrade the LW-SSO settings from version 9.x, these LW-SSO settings will be migrated, but LW-SSO will be disabled in OO 10.x, even if it was previously enabled in version 9.x.

When you enable LW-SSO afterward, you may receive warnings under certain scenarios. To clear the warnings from the log, follow the below steps to set the management URL property using the Fully Qualified Domain Name(FQDN).

Central and RAS installed on the same machine

When Central and a RAS are installed on the same machine, and the LW-SSO settings are enabled, you must set the management URL property using the Fully Qualified Domain Name.

1. Stop the RAS process.
2. In the **ras/conf/ras-wrapper.conf** file, changewrapper.java.additional.<x>=-Dmgmt.url=<protocol>://localhost:<port>/oo to wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://<FullyQualifiedDomainName>:<port>/oo
3. Start the RAS process.

RAS installed on a different machine

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

Connecting Another Application to Central Through LW-SSO

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

1. Stop the Central process.
2. In the **central/conf/central-wrapper.conf** file, change wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://localhost:<port>/oo to wrapper.java.additional.<x>=-Dmgmt.url=<protocol>://<FullyQualifiedDomainName>:<port>/oo
3. Start the Central process.

Related Topics

Set up security using LW-SSO

Change the service name

If you want to change the service name and description from the default values, you can edit these in the **central-wrapper.conf** file.

Note This procedure can be applied to a RAS as well. Just replace the word "Central" with "RAS". For example, use **ras-wrapper.conf**, **ras\conf**, and so on.

1. Stop the Central service.
2. Navigate to the **central\bin** directory. The default is **C:\Program Files\Micro Focus\Operations Orchestration**.
3. Run **uninstall-central-service.bat**.
4. Verify that the service is uninstalled.
5. Navigate to the **central\conf** directory.
6. Make a copy of the **central-wrapper.conf** file. Note This step is an optional step used for saving the original configuration, so that you have a backup if a rollback is required or if the modified file becomes unusable in any way.
7. Edit the **central-wrapper.conf** file, by modifying the following lines to suit your needs: wrapper.name wrapper.displayname wrapper.description For example: # Name of the service wrapper.name=OOCentral # Display name of the service wrapper.displayname= Operations Orchestration Central # Description of the service wrapper.description=Operations Orchestration Central is the run time environment of Operations Orchestration. It is used for running flows, monitoring the various runs, and generating reports.
8. Run **install-central-as-service.bat**.
9. Verify that the service is installed with the new names and description.
10. Start the new Central service.

Change the timeout limit for the OO web interface

You can change the default session timeout (30 minutes) of the Central server by configuring the **tomcat web.xml** file.

1. Open the **<installation_folder>/central/tomcat/conf/web.xml** file with any XML editor.
2. Search for the following:

```
<!-- ==================== Default Session Configuration ================= -->
<!-- You can set the default session timeout (in minutes) for all newly   -->
<!-- created sessions by modifying the value below.                       -->

  <session-config>
      <session-timeout>30</session-timeout>
  </session-config>
```

3. Change the **session-timeout** value (in minutes).
4. Restart the Central server.

Schedule database purging

In order to keep the Operations Orchestration database size under control, it is important to schedule database purging operations from day one.

Purging flows

The purging flows are available in the **Micro Focus Solutions content pack** (available on ITOM Marketplace). It is recommended to deploy this content pack, configure the flows with your required settings, and schedule them in Central.
The following purging flows are located under **Library** > **Integrations** > **Micro Focus** > **Operations Orchestration** > **10.x** > **Database**:

- **Purge Execution Summary** - Purges run data. It is recommended to schedule this flow as soon as you start to run flows.
- **Purge Debug Events** - Purges Studio Remote Debugger event data. If you use the Studio Remote Debugger, schedule this flow.
- **Purge Audit Records** - Purges old auditing records if auditing is enabled. If you have enabled security auditing, schedule this flow.
- **Purge Rerun Info** - Purges rerun data. If you have defined rerun points in your flows, schedule this flow.

For more information about these purging flows, see the flow descriptions in Central.

Purging APIs

The purge APIs enable the system administrator to purge:

- Run data such as flow inputs, outputs, and step logs. This run data should be purged regularly, because running large numbers of runs can cause the database to reach the maximum table size. Note This only affects the data of completed runs.

- Debugger events - when using remote-debugging sessions from Studio.
- Audit records - if auditing is enabled.

By using the purge APIs, you can purge the data manually as required, or by scheduling recurrent flows that incorporate these APIs.

As an alternative to using the purging flows, you can perform purging functions via API.
The following APIs are available:

- **DELETE /audit/records**: Purges old auditing records if auditing is enabled.
- **DELETE /debugger-events**: Purges Studio Remote Debugger event data.
- **DELETE /executions**: Purges run data such as bound inputs, outputs and step log events. This run data needs to be purged regularly, because running large numbers of runs can cause the database to reach the maximum table size. Note This only affects the data of completed runs.
- **DELETE/executions/rerun**: Purges the rerun data from the database.
- **DELETE /steps-log**: Purges step data according to time and number of executions to purge.

By using the purge APIs, you can purge the data manually as required, or by scheduling recurrent flows that incorporate these APIs.

Capture the credentials of the logged-in user

An administrator can enable OO to capture the credentials of the logged-in user. This makes it possible to run flows automatically, when they require logged-in user credentials as input parameters in flow steps.

When this option is enabled, when a user schedules, resumes, or starts a flow that contains logged-in user credentials as an input parameter, the credentials of that logged-in user are captured in the flow run context (in encrypted form).

During log in, a warning message appears, telling the user that their credentials may be captured. For a remote Central, the warning appears in the Remote Debugging login page.

Note By default this option is not enabled and the flow will prompt for logged-in user credentials at the relevant step.

To enable this option in the Central UI:

Select the **Enable the capture of logged-in user credentials** check box in the Central **Security > Settings** tab.

To enable this option using the OOSH utility:

1. Connect to Central using OOSH.
2. In the command line, type: ssc --key record.loggedin.user.credentials --value true
3. Press the **Enter** key.

Specify communication protocols

To specify the communication protocols that must be used by the system notification feature, add the `system.notifications.smtp.enabled.protocols` property in the **central-wrapper.conf** file. To do this:

1. Open the **central-wrapper.conf** file located in the `<installation_directory>\central\conf` directory.
2. Add the `system.notifications.smtp.enabled`.protocols property as follows:
   The `system.notifications.smtp.enabled.protocols` property is optional and its default value is `TLSv1.2 TLSv1.1 TLSv1 SSLv3`. You can overwrite the default value and specify the protocol that you want as a list of space-separated values. For example, to specify more than one protocol:
   `wrapper.java.additional.34=Dsystem.notifications.smtp.enabled.protocols=<protocol 1> <protocol 2> <protocol 3>`

Initial administration tasks for OO Workflow Designer

This topic includes administration tasks that you will generally perform once, while setting up OO Workflow Designer after installation.

Related topics

- Displaying system information
- Configuring a Proxy for OO Workflow Designer

Displaying system information

You can view a system information report, which displays version and static system information, database connection information, and general system information.

To view the system information report, enter the following text into your browser address line:

`http://<OO designer URL>:<port number>/oo-designer/reports/sysinfo`

For example:

Version and static system information

Buid Id	Build Job Name	Build Time	Revision	Version	Time
7	2018.08-NG-designer-product	2018-07-30_14-04-50	44f6faf3167327864e17c5a02d6a1ac40b5ffed1	2018.08.0	10-08-2018 00:20

Database connection information

URL	Database Name	Database Version	Username
jdbc:postgresql://localhost:5432/dsr201808demo	PostgreSQL	9.4.9	dsr201808demo

General system information

Key Name	
java.vendor	Azul Systems, Inc.
cloudslang.maven.repo.local	/usr/local/microfocus/oo_designer/designer/var/repo
sun.java.launcher	SUN_STANDARD
catalina.base	/usr/local/microfocus/oo_designer/designer/tomcat
sun.management.compiler	HotSpot 64-Bit Tiered Compilers
catalina.useNaming	true
run.executions.for.debugger.data.overflow	TRUE
os.name	Linux
sun.boot.class.path	/usr/local/microfocus/oo_designer/java/lib/resources.jar:/usr/local/microfocus/oo_designer/java/lib/rt.jar
host.name	ef42a4a0e0d5
mgmt.url	https://localhost:9501/oo-designer
java.vm.specification.vendor	Oracle Corporation
java.runtime.version	1.8.0_162-b01
wrapper.native_library	wrapper
javax.net.ssl.keyStore	/usr/local/microfocus/oo_designer/designer/var/security/key.store
wrapper.key	eNvukTXAZudlZtxL8hwOwdj3Wkx0s23R
user.name	root

Configure a proxy for OO Workflow Designer

If you need to configure a proxy for OO Workflow Designer, add or modify the value of the following properties located in the <installation_folder>/designer/conf/designer-wrapper.conf file:

- **proxy.mode (enabled/disabled)**
- **proxy.host**
- **proxy.port**
- **proxy.user**
- **proxy.password**

Recurring administration tasks

This section includes administration tasks that you may need to perform periodically or in response to a situation that occurs.

Encrypt and obfuscate passwords
Changing the database password
Change the database IP
Change the hostname IP
Adjust the timing of Quartz jobs
Reverse the direction of the RAS - Central communication
Change the URL of a load balancer/Central on the RAS side after its installation
Back up and recover Operations Orchestration
Set up disaster recovery
System tuning
Database maintenance

Encrypt and obfuscate 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.

Encrypt 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>.

Obfuscate 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>.

Create a prompt for the password

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

```
C:\Program Files\Micro Focus\Operations Orchestration\central\bin>encrypt-password.bat
Password (typing will be hidden):
Confirm password (typing will be hidden):
{ENCRYPTED}gAkPCLQsYDhoR1Y2q9bjCQ==

C:\Program Files\Micro Focus\Operations Orchestration\central\bin>
```

This will create a prompt for the hidden password inputs.

Changing the database password

1. If Central is up and running, stop the Central Service.
2. Run the encrypt-password script with the `--password <password>` option, where `<password>` is the database password. Note You can either use `-p` as the flag to encrypt the password or `--password`. For example: <install-dir>/central/bin/encrypt-password --password <plain-text-pass>
3. Copy the result. It should appear as follows: ${ENCRYPTED}<some_chars>.
4. Go to the folder **<installation_folder>/central/conf**, and open the **database.properties** file.

Change the `db.password` value to the one that you copied. In a cluster, make sure to do this on each node.

Note The password is stored encrypted, so make sure you include the `{ENCRYPTED}` prefix.The = character needs to be escaped. For example, db.password=GFDGDS12\=\=.

encrypting password section to show the option to prompt for a password as an option vs having it as a separate section.

The database change documentation needs to add a step after copying in the encrypted password to the database.properties file the equal "=" sign in the encrypted text must be escaped by which I mean a backspace in front of it.

example:

result from encrypt-password.bat => DSFUIEKEQ7==

After copying into the database.properties it would need to look like this db.password=DSFUIEKEQ7\=\=

Change the database IP

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

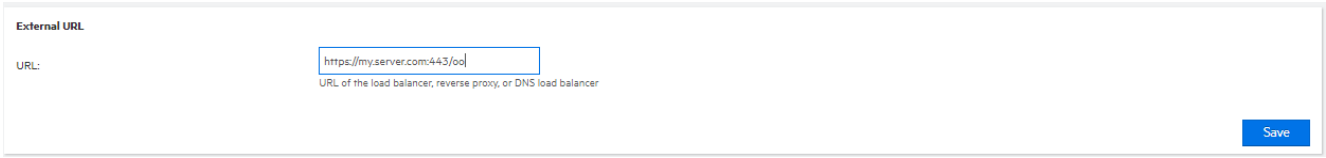
1. Edit the file **<installation_folder>/central/conf/database.properties**.
2. Look for the `jdbc.url` parameter. For example:
   jdbc.url=jdbc\:jtds\:sqlserver\://16.60.185.109\:1433/schemaName;sendStringParametersAsUnicode\=true
3. Change the IP address\FQDN of the database server.
4. Save the file.
5. Restart Central.

Change the hostname IP

If the Central hostname or IP has changed:

If the hostname/IP has changed in a Central, you will need to manually update this hostname/IP address in a number of places.

1. Edit the file **<installation_folder>/central/conf/central-wrapper.conf**.
2. Look for the `-Dmgmt.url` parameter and update it with the correct hostname or IP address. For example: -Dmgmt.url=http://[machine_name or IP]:8080/oo
3. Update the same `-Dmgmt.url` parameter in the file **<installation_folder>/ras/conf/ras-wrapper.conf**. You need to perform this step for each RAS that communicates with the Central.
4. If you are using a Central certificate, generate a new Central certificate with the new FQDN. It is not recommended to use the IP. You will also need to replace the certificate in the Central keystore.
5. If there is a load balancer link configured, update this link in Central.



6. Restart Central and all affected RASes.
7. If you have configured a remote debugging session that uses the Central hostname or IP address, you will need to modify the connection URL accordingly in Studio.

If the RAS IP has changed:

If the RAS IP is changed, this change affects the presentation in the Central URL, under **System Configuration** > **Topology** > **Workers**.

- If the RAS is a “reverse RAS”, it will become disabled in the **Topology** > **Workers** tab. You need to edit the row, and replace the previous IP with the new one.
- If the RAS is a “standard RAS”, the change will not affect functionality, because it is the RAS that connects to the Central.

Related topics

Debug a remote Central with Studio

Adjust the timing of Quartz jobs

In the Operations Orchestration system, quartz jobs run periodically for maintenance of the system. Each job runs for a set amount of time, and is repeated at set intervals. The following are examples of job triggers:

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

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

Note Any change to the timings can have a major effect on the system. Consult with your service representative before making any changes to these triggers.

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

This changes the triggering times of the job.

Note The events persistency mechanism is deprecated (see **onRolling:OO_EXECUTION_EVENTS_Trigger**). You can configure this job if you use the Remote Debugger.

Reverse the direction of the RAS - Central communication

When installing a RAS, you chose whether the RAS will initiate the connection to Central (Standard RAS) or wait for Central to initiate the connection (Reverse RAS).
After the installation, it is possible to change this communication direction.

1. Edit the file **<installation_folder>/central/conf/central-wrapper.conf**.
2. Look for the `-register.ras` parameter:
   - If `-register.ras=true`, this means that the RAS will initiate the connection to Central (standard RAS).
   - If `-register.ras=false`, this means that the RAS will wait for Central initiate the connection (reverse RAS).
3. Update the same `-register.ras` parameter in the file **<installation_folder>/ras/conf/ras-wrapper.conf**. You need to perform this step for each RAS that communicates with the Central.
4. Restart Central and all affected RASes.
   - After a standard RAS restarts, the RAS will initiate communication to Central and Central will update the RAS registration accordingly.
   - For a reverse RAS, you will also need to configure Central to register the RAS . You do this in the Central UI, in the **Topology** > **Workers** tab.

Change the URL of a load balancer/Central on the RAS side after its installation

While the best practice is to configure the URL of a Central/load balancer via the installer, if you need to make changes to the URL after the RAS was already installed, you can do this by editing the **ras-wrapper.conf** file.

For example, this would be needed if you installed a RAS against a Central/load balancer and the Central/load balancer's FQDN changed. You will need to change Central/load balancer's URL stored at the RAS level, so that the RAS is able to communicate again with the Central/load balancer.

1. Stop the RAS.
2. Open the **ras-wrapper.conf** file, located under **<installation folder>/ras/conf**.
3. Edit the URL in the following line: wrapper.java.additional.<x>=-Dmgmt.url=<http or https>://<FQDN>:<port>/oo
4. Restart the RAS.

Back up and recover Operations Orchestration

You can back up and recover Operations Orchestration as described below.

Back up

Back up the entire content of the **<installation_folder>central\var\security** folder and the **<installation_folder>central\conf\database.properties** file.

Recover

1. Install a new Central with an existing schema. The installation will fail on the **Start Central** step.
2. Stop the Central service and verify that Central is not running.
3. Open the **central/var/security** folder.
4. Delete the **credentials.store** file, if it exists.
5. Override the **encryption_repository** and **encryption.properties** files with the backed up versions.
6. Override the **central/conf/database.properties** file with the backed up file.
7. Start the Central service.

Set up disaster recovery

Follow the below instructions to set up a successful disaster recovery.

Plan for disaster recovery

To enable disaster recovery, create a secondary datacenter by replicating the following:

- Database configuration
- Schema
- OO configuration and security files: **<installation_folder>\central\var\security** and **<installation_folder>\central\conf**

The replication can be ongoing, using whatever means you choose, for example, disk replication or database replication.

Perform disaster recovery

OO supports cold disaster recovery, which requires a manual process to switch over from the primary datacenter to the secondary datacenter. This may be required in the case of a full or partial failure in the primary datacenter.

1. Restore the OO server(s) by reinstalling all nodes (Central and RAS). Use the database configuration, schema, and OO configuration and security files that you replicated for the secondary datacenter. Note You can configure the database (by editing the **database.properties** file). See the "*Secure and harden Operations Orchestration*" topic for more information about configuring the database connection details.
2. Delete the old worker nodes in the database, which no longer exist.
3. Turn off OO in the primary datacenter.
4. Switch over to the secondary datacenter.

Related Topics

Secure and harden Operations Orchestration

System tuning

You can configure one or more of the following parameters for each Central node. The JVM and worker threads are also applicable to each RAS node.

Note Changing any of the parameters below requires a restart of the configured node.

- **JVM heap size** - You can increase the initial and maximum size of the Central/RAS heap by configuring the **central-wrapper.conf** and **ras-wrapper.conf** files (located under **<installation_folder>/<central or ras>/conf/)**. wrapper.java.initmemory=<value in MB> wrapper.java.maxmemory=<value in MB> We recommend configuring Central to 4GB and RAS to 2GB.
- **Worker threads** - By default, each node has 20 worker threads. If your flows have a large number of parallel or multi-instance lanes, we recommend gradually increasing this number, over several iterations. You can configure the number of worker threads in the **central-wrapper.conf** and **ras-wrapper.conf** files (located under **<installation_folder>/<central or ras>/conf/)**. Open the file and edit the `-Dcloudslang.worker.numberOfExecutionThreads` property. This is a new property, which was not supported in versions earlier than 10.22. Therefore, it should be added, if it doesn't exist, as follows: wrapper.java.additional.<next available number>= -Dcloudslang.worker.numberOfExecutionThreads=<new value> You can configure the number of worker threads in the **engine-webapp.jar** file (located under **<installation_folder>/central/tomcat/webapps/oo/WEB-INF/lib/**). Open the JAR file and edit the `numberOfExecutionThreads` property in the **spring/engineContext.xml** file.

- **Database connections** - You can increase the number of database connections in the **database.properties** file (located under **<installation_folder>/central/conf/**). Edit the `db.pool.maxPoolSize` property. We recommend changing it to 100 connections.

Related topics

Performance and sizing information

Database maintenance

Database maintenance is important in order to maintain the efficiency and throughput of OO.
For detailed information on database housekeeping (index rebuilding, table statistics updated, and so on), see the Set up database environment section.

Related topics

Set up database environment

Recurring administration tasks for OO Workflow Designer

This section includes administration tasks that you may need to perform periodically or in response to a situation that occurs.

- Change the database password
- Change the database IP
- Change the host name or IP
- Encrypting passwords
- Set the system localization
- Change the language in wrapper file
- Create a Git repository
- Adjust the logging levels
- Change the service name and description
- Changing the Timeout Limit for the OO Workflow Designer Web Interface
- Configure content pack import size
- Back up OO Workflow Designer
- Set up disaster recovery
- System tuning
- Database maintenance

Changing the database password

1. If OO Designer is up and running, stop the OO Workflow Designer Service.
2. Run the encrypt-password script with the --password <password> option, where <password> is the database password. Note You can either use -p as the flag to encrypt the password or --password.
   For example:
   <install-dir>/designer/bin/encrypt-password --password <plain-text-pass>
3. Copy the result. It should appear as follows: `{ENCRYPTED}<some_chars>.`
4. Go to the folder <installation_folder>/designer/conf, and open the database.properties file.
5. Change the db.password value to the one that you copied. In a cluster, make sure to do this on each node. Note The password is stored encrypted, so make sure you include the {ENCRYPTED} prefix.

Related topics

- Encrypting passwords

Change the database IP

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

1. Edit the file **<installation_folder>/designer/conf/database.properties**.
2. Look for the `jdbc.url` parameter. For example:
   `jdbc.url=jdbc\:jtds\:sqlserver\://16.60.185.109\:1433/schemaName;sendStringParametersAsUnicode\=true`
3. Change the IP address/FQDN of the database server.
4. Save the file.
5. Start OO Workflow Designer

Change the hostname IP

If the hostname/IP has changed in OO Workflow Designer, you must manually update this hostname/IP address.

1. If you are using an OO Workflow Designer certificate, generate a new certificate with the new FQDN. It is not recommended to use the IP address. You must also replace the certificate in the OO Workflow Designer keystore.
2. Restart OO Workflow Designer. Related topics Replace the OO Workflow Designer TLS server certificate

Encrypting passwords

You can encrypt a password using the encrypt-password script, which is located in <installation_folder>/designer/bin.

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.

Encrypt passwords

1. Locate the encrypt-password script, in <installation_folder>/designer/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>.`

Create a prompt for the password

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

```
C:\Program Files\Micro Focus\Operations Orchestration
Designer\designer\bin>encrypt-password.bat
Password <typing will be hidden>:
Confirm password <typing will be hidden>:
<ENCRYPTED>sdf874sgs787fds09ewff==
C:\Program Files\Micro Focus\Operations Orchestration
Designer\designer\bin>
```

This will create a prompt for the hidden password inputs.

Set the system localization

If your OO Workflow Designer system is localized, you will need to set the properties to reflect the system locale, in the designer-wrapper.conf file.

1. Open the designer-wrapper.conf file in a text editor (located under <installation_folder>/designer/conf).
2. Edit the following properties:
   set.LANG=
   set.LC_ALL=
   set.LANGUAGE=
   wrapper.java.additional.<x>=-Duser.language=
   wrapper.java.additional.<x>=-Duser.country=
   For example, for Japanese:
   set.LANG=ja_JP
   set.LC_ALL=ja_JP
3. Save the designer-wrapper.conf file.

Change the language in wrapper file

When OO Workflow Designer was installed, there was an option to select a supported language in addition to English. This language support is required if, for example, you need to ping a server that is configured in Japanese. You can change this language by editing the **designer-wrapper.conf** file.

1. Open the **designer-wrapper.conf** file in a text editor (located under **<installation_folder>/designer/conf**).
2. Edit the language property. For example, change wrapper.lang=en_US to wrapper.lang=ja_JP.
3. Save the **designer-wrapper.conf** file.

Create a Git repository

If you do not have a Git server account, you can create a local Git repository. OO Workflow Designer includes a batch file for creating this repository: git-create-repo.bat.

1. Double-click the git-create-repo.bat file in the <installation path>/bin folder.
2. In the command line dialog box, enter the path to an existing folder on the current machine or on a network location, where you want to create the repository. For example: \\xxx.net\shared\GIT\GitRepo. If the repository is on a shared file system, make sure that the user for whom the OO Workflow Designer server will run has read and write permission for this file system.
3. You are now prompted to enter a name for the new repository that will be created under the path you provided earlier. Add any name. You do not need to add the suffix ".git" to the name, as this is added by the tool. Note The repository name must not contain characters that are forbidden by Windows or Linux, for example, *, <, >, or % and must not begin and end with white spaces.
4. The URL of the newly created Git repository is displayed. Make a note of this URL, and press any key to close the command-line dialog box. Note The URL should be in the following format: file://<path>/<repo>.git

Adjust the logging levels

Because logging activity can slow the performance of OO Workflow Designer and create very large log files, it is important for OO Workflow Designer to run with appropriate logging levels. The default logging level is set to INFO. This level provides the necessary information without impacting performance. In addition, you can adjust the granularity of the information in the log, separately for regular logging, deployment, and execution. The granularity options are:

- INFO - Default logging information
- DEBUG - More logging information
- WARN - Less logging information
- ERROR - Less logging information

To adjust the granularity in the logging:

1. Open the **log4j.properties** file (under **/<installation_folder>/designer/conf/log4j.properties**). Replace INFO with DEBUG, WARN or ERROR in the relevant places in the **log4j.properties** file.
   For example:
   log.level=WARN
   import.log.level=INFO
   project.log.level=INFO
   scm.log.level=INFO
   Note The Tomcat server keeps access logs under **/<installation_folder>/designer/var/logs**. To prevent these log files from growing in size, it is recommended that you monitor the size of the files named **localhost_access_log.yyyy-mm-dd.txt**, and delete old files as necessary. Use established best practices for your operating system, whether Windows or Linux. For example, to delete files older than 10 days: **Unix** find <installation_folder>/designer/var/logs/ -name " localhost_access*.txt" -type f -mtime +10 -exec rm -f {} \; **Windows** C:\Users\Administrator>forfiles /p "<installation_folder>\designer\var\logs" /s /m localhost_access*.txt /d -10 /c "cmd /c del @PATH" where <installation_folder> refers to the location of the OO Workflow Designer installation folder You can set this up using a schedule in OO Central or use Cron or Windows Scheduler to run these commands the background.

Change the service name and description

If you want to change the service name and description from the default values, you can edit these in the **designer-wrapper.conf** file.

1. Stop the OO Workflow Designer service.
2. Navigate to the **designer\bin** directory. The default is **C:\Program Files\Micro Focus\CSA\workflow-designer\designer\bin**.
3. Run **uninstall.exe**.
4. Verify that the service is uninstalled.
5. Navigate to the **designer\conf** directory.
6. Make a copy of the **designer-wrapper.conf** file. Note This step is an optional step used for saving the original configuration, so that you have a backup if a rollback is required or if the modified file becomes unusable in any way.
7. Edit the **designer-wrapper.conf** file, by modifying the following lines to suit your needs:
   wrapper.name wrapper.displayname wrapper.description
   For example:
   # Name of the service
   wrapper.name=MICROFOCUSOODC
   # Display name of the service
   wrapper.displayname=Micro Focus Operations Orchestration Workflow Designer
8. Run **install-designer-as-service.bat**.
9. Verify that the service is installed with the new names and description.
10. Start the new OO Workflow Designer service.

Changing the Timeout Limit for the OO Workflow Designer Web Interface

You can change the default session timeout (30 minutes) of the OO Workflow Designer server by configuring the tomcat web.xml file.

1. Open the <installation_folder>/designer/tomcat/conf/web.xml file with any XML editor.
2. Search for the following:

```
<!-- ==================== Default Session Configuration ================= -->
<!-- You can set the default session timeout (in minutes) for all newly   -->
<!-- created sessions by modifying the value below.                       -->

  <session-config>
      <session-timeout>30</session-timeout>
  </session-config>
```

3. Change the session-timeout value (in minutes).
4. Restart the OO Workflow Designer server.

Configure content pack import size

The default size limit of content packs that can be uploaded to OO Workflow Designer is 200 MB (the exact size limit is 209715200 bytes). However, you can configure this size limit. For example, you may want to adjust the size limit to protect against DOS (Denial of Service).

1. Open the **secured.properties** file, located under **<installation_dir>/designer/var/security/**.
2. Update the value for the **upload.max.fileSize.limit** property.
   For example:
   upload.max.fileSize.limit=209715200

3. Restart OO Workflow Designer. For more information about the **secured.properties file**, see "Configure the secured properties" topic. Related topics Configure the secured properties

Back up OO Workflow Designer

Backing up

Back up the entire content of the **<installation_folder>designer\var\security** folder and the **<installation_folder>designer\conf\database.properties** file.

Recovery

1. Install a new OO Workflow Designer with an existing schema. The installation will fail on the **Start OO Workflow Designer** step.
2. Stop the OO Designer service and verify that OO Workflow Designer is not running.
3. Open the **designer/var/security** folder.
4. Delete the **credentials.store** file, if it exists.
5. Override the **encryption_repository**, **encryption.properties** and **secured.properties** files with the backed up versions.
6. Override the **designer/conf/database.properties** file with the backed up file.
7. Start the OO Workflow Designer service.

Set up disaster recovery

Plan for disaster recovery

To enable disaster recovery, create a secondary datacenter by replicating the following:

- Database configuration
- Schema
- OO Workflow Designer configuration and security files: <installation_folder>\designer\var\security and <installation_folder>\designer\conf

The replication can be ongoing, using whatever means you choose, for example, disk replication or database replication.

Perform disaster recovery

OO Workflow Designer supports cold disaster recovery, which requires a manual process to switch over from the primary datacenter to the secondary datacenter. This may be required in the case of a full or partial failure in the primary datacenter.

1. Restore the OO Workflow Designer server(s) by reinstalling all nodes. Use the database configuration, schema, and OO Workflow Designer configuration and security files that you replicated for the secondary datacenter. Note You can configure the database (by editing the database.properties file).
2. Delete the old worker nodes in the database, which no longer exist.
3. Turn off OO Workflow Designer in the primary datacenter.
4. Switch over to the secondary datacenter. Related topics
   - Set up database environment for OO Workflow Designer

System tuning

You can configure one or more of the following parameters for each OO Workflow Designer node.

Note Changing any of the parameters below requires a restart of the configured node.

- **JVM heap size** - You can increase the initial and maximum size of the OO Workflow Designer heap by configuring the **designer-wrapper.conf** files (located under **<installation_folder>/designer/conf/**). `wrapper.java.initmemory=<value in MB>` `wrapper.java.maxmemory=<value in MB>` We recommend configuring OO Workflow Designer to 4GB.
- **Database connections** - You can increase the number of database connections in the **database.properties** file (located under **<installation_folder>/designer/conf/**).

Edit the `db.pool.maxPoolSize` property. We recommend changing it to 100 connections.

Database maintenance

Database maintenance is important in order to maintain the efficiency and throughput of OO Workflow Designer.

For detailed information on database housekeeping (index rebuilding, table statistics updated, and so on), see *Prepare the database environment* section.

Related topics

Prepare the database environment

Set up database environment

This section describes the database administrator how to configure four different types of database, to be used with OO.

Prepare the database environment
Deploy and maintain a Microsoft SQL server database
Deploy and maintain an Oracle database
Deploy and maintain a MySQL database
Deploy and maintain a Postgres database
Database purging
Additional guidelines for Microsoft SQL server
Additional guidelines for Oracle
Installation Wizard 'Other database' option

Prepare the database environment

This section contains information about the types of databases used with Operations Orchestration.

Overview

The term “database” may be interpreted in several ways, depending on the database vendor/technology used. In Oracle, the term “database” relates to a collection of files containing data and metadata. A single Oracle database may contain one or more schemas (and users). A Microsoft SQL Server “database” is closer in definition to Oracle’s "schema" than to Oracle’s "database".
In order to avoid confusion, this section will use the following terms:

- Instance/server – the software and memory structures providing RDBMS services
- Database – the entity containing tables, views, indexes, and so on

Operations Orchestration requires a single database to be created. This database may co-exist with other databases contained in a database server.
You can set up an OO database on one of the following database server types:

- Microsoft SQL Server Standard/Enterprise (2008 R2/2012/2014/2016)
- Oracle 11gR2 Standard/Enterprise Server, including RAC environment
- Oracle 12cR1 Standard/Enterprise Server - regular instance (non-CDB) including RAC environment and Oracle 12cR2 Standard/Enterprise Server - regular instance (CDB).
- Postgres (9.3/9.4/9.5/9.6)
- MySQL Community/Standard/Enterprise Server (5.5/5.6/5.7)

Language Support
OO can be installed and used in any supported language (English, French, German, Spanish, Japanese, and Simplified Chinese). Databases and database servers should be properly configured in order to support the desired language.
If you are using OO in a multilingual environment, it is preferable that you configure your database to use the Unicode character set. See the relevant deployment topic for detailed instructions.
If you have user input in two languages apart from English (for example, German and Chinese) then MS SQL should not be used. You should use an alternative database such as Oracle, MySQL, or Postgres with the recommended Unicode configuration for OO.

- This section is intended for trained database administrators. If you are not familiar with the type

of database you wish to use, or you feel that you lack the relevant knowledge required in order to create and configure an OO database, see the database vendor's documentation and make sure you fully understand each action you perform as described in this section.

- OO database connectivity relies on Java JDBC. If your environment requires specific adaptations or security measures, see the JDBC documentation (or database vendor documentation) to find out exactly how the JDBC connection URL should be formatted.
- This section describes the required database settings for OO. Settings that are not specified in this document can be left with default values or configured by your organization's DBA.

Using database clusters

Database clusters protects the database from several types of failures and makes Operation Orchestration more robust.

OO does not provide database cluster-related features of its own, such as database connection failover. It relies on the functionality of the JDBC package being used and the database cluster environment capabilities, such as SCAN/AG listeners.

You can install OO in conjunction with any type of database cluster environment that satisfies the following conditions:

- Connection pooling is supported
- A single, valid database URL can be provided
- It provides reliable transaction handling during failover (a single, complete transaction must either completely fail or succeed)

The OO installer does not provide database cluster-specific installation options. During installation, you need to provide the installer a simplistic / "regular" form of database connection. Database URL adaptations usually have to be made after the system is already installed in the **database.properties** configuration file.

Database security

The OO database is at the heart of the OO system, containing the OO system configuration and possibly corporate-sensitive data. It is highly recommended that you enforce strict security settings on your database by following these guidelines.

Harden your database server by following the recommendations from the database vendor and the operating system vendor. Server hardening may include, among other items to:

- Restrict SSH access on Linux servers to a well-controlled set of OS users

- Enforce a strict OS user password policy – password length, complexity, lockout-policy, and so on
- Set up an audit system to detect and report break-in attempts

Harden your database user/login accounts to:

- Enforce the password policy – password length, complexity, lockout-policy, and so on
- Restrict access to administration-level accounts
- Set up an audit system to detect and report break-in attempts

Database encryption at file-system level is supported as long as it does not compromise OO's performance, and completely transparent to OO. For example, Oracle TDE may be used as long as it does not cause any performance degradation or prevent LOB free-space reclaim.

Database schema

Operation Orchestration databases are always created using the default database schema. For SQL Server, OO uses the **dbo** schema, and for PostgreSQL - the **public** schema. Currently, no other schemas are supported by OO.

OO database connection pooling

A connection pool is a cache of database connections maintained so that the connections can be reused when future requests to the database are required. Connection pools are used to enhance the performance of executing commands on a database. In connection pooling, after a connection is created, it is placed in the pool and used again so that a new connection does not have to be established. If all the connections are being used, a new connection is made and is added to the pool. Connection pooling reduces the time required to establish a connection to the database.

Configure the database connection pooling in OO Central

- Locate the file **central/conf/database.properties** in an existing Central installation.
- Add or edit the property named `db.connection.pool.strategy` to one of the values such as `hikari` or `tomcat-dbcp2`. In case the value is left empty, the default strategy `tomcat-dbcp2` will be used.Example:`db.connection.pool.strategy=hikari`
- Restart the OO Central service for the changes to take effect.

OO Database sizing

OO 10.x keeps all flow and step execution data in the database by default, enabling comprehensive debugging of previous flow runs. As a result, the database size grows in accordance to the system throughput and flow complexity.

In version 10.22 and later, the amount of data kept in the database can also be controlled by setting the system persistency level. See the *OO Tuning Guide*the "*Performance and sizing information*" section for more details.

This section will help you prepare for installing OO. By estimating your system scale (small/standard/enterprise), you will be able to derive the amount of disk space required, derive the amount of memory (RAM) used by the database, and determine additional database installation parameters.

Estimate the system scale according to complexity:

System Criteria\Scale	Small	Standard	Enterprise
Average number of steps per flow	50 or less	100 or more	1000 or more
Average flow duration	Seconds to minutes	1 hour or more	Up to 24 hours
Average payload size per flow [1]	1 KB or less	1 MB or more	4 MB or more

Estimate the system scale according to concurrency/frequency:

System Criteria\Scale	Small	Standard	Enterprise
Average number of flows per day	Less than 100	300 or more	1000 or more

[1] – The term "payload" refers to flow/step data of considerable size. For example, a CSV or XML1. file used as flow input, a large JSON object / REST API call data.

The following table provides disk space and memory requirements for different OO deployment scales:

System Scale\Parameter	OO Database Disk Size Requirements	Memory
Small	50 GB	4 GB
Standard	500 GB	8 GB
Enterprise	Between 500 GB and 2 TB	12 GB

Notes:

- Disk space and memory values are estimates. Actual disk space and memory consumption vary, depending on the database vendor and database server configuration.
- Memory (RAM) reflects recommended database memory, not the overall amount of memory

available on the database machine (virtual or physical server).
- Disk space reflects the amount of disk space required for day-to-day operation of the OO system and reasonable historical data retention—not including long-term database backups. It is highly recommended to regularly purge historical data using the stored procedure published on ITOM Marketplace and the database purge flows provided in the Micro Focus Solutions content pack, in order to keep the database size within your environment limitations.
- The amount of additional disk space required for keeping OO database backups depends on the backup policy (frequency and retention period).

Hardware requirements

The following table describes the hardware (CPU and memory) requirements recommended for the each of the database servers.

Note The memory values reflect database memory consumption as part of the entire machine's memory.

Database\Scale	Small/Standard				Enterprise			
	CPUs		RAM		CPUs		RAM	
	Small	Standard	Min	Rec	Min	Rec	Min	Rec
SQL Server	2	4	4 GB	8 GB	4	12	8 GB	12 GB
Oracle	2	4	4 GB	8 GB	4	12	8 GB	12 GB
MySQL	2	4	4 GB	8 GB	4	12	8 GB	12 GB
Postgres	2	4	4 GB	8 GB	4	12	8 GB	12 GB

Min = Minimal value, **Rec** = Recommended value In addition to the generalized hardware requirements above, refer to the appropriate hardware requirements and software requirements sections per database.

Related topics

- Deploy and maintain a Microsoft SQL server database
- Deploy and maintain an Oracle database
- Deploy and maintain a MySQL database
- Deploy and maintain a Postgres database
- Performance and sizing information

Deploy and maintain a Microsoft SQL server database

In order to deploy OO using Microsoft SQL Server, you must have an existing SQL Server database service. If you need to create a new database service, see the relevant documentation provided by Microsoft, because this information is not included within this section. However, this section contains recommendations for the SQL Server configuration.

It is also recommended to use the SQL Server Agent service in order to comfortably schedule data purging and index maintenance jobs.

This section includes the following topics:

Workflow for Microsoft SQL Server Deployment
System Requirements for Microsoft SQL Server
Configure SQL server
Manually create an OO database on Microsoft SQL server
Microsoft SQL server database maintenance

Workflow for Microsoft SQL Server Deployment

To deploy OO using Microsoft SQL Server, perform the following steps:

1. **Review sizing guidelines**. For details, see "*OO Database Sizing*" topic in "*Prepare the database environment*" section.
2. **Review Hardware and Software Requirements**. For details, see "*System requirements for Microsoft SQL server*" section.
3. **Configure Microsoft SQL Server**. For details, see "*Configure MySQL server*" section.
4. **Create an OO database on Microsoft SQL Server**. For details, see "*Manually Creating an OO Database on Microsoft SQL Server*" section.
5. (Optional) **Set up Windows authentication**. For details, see "*Using Windows Authentication to Access Microsoft SQL Server Databases*" in "*Additional Guidelines for Microsoft SQL Server*" section. This step is only relevant if you are using Windows authentication instead of SQL Server authentication.

Related topics

- Prepare the database environment
- System requirements for Microsoft SQL server
- Configure SQL server
- Manually create an OO database on Microsoft SQL server
- Additional guidelines for Microsoft SQL server

System Requirements for Microsoft SQL Server

This section describes the system requirements for working with Microsoft SQL Server in conjunction with OO.

Hardware requirements

For OO database sizing guidelines and hardware requirements, see "OO *Database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" topic.
For Microsoft SQL Server hardware requirements, see the relevant installation section for your Microsoft SQL Server release and operating system.

Software requirements

The following table lists the Microsoft SQL Server releases supported by OO:

Microsoft SQL Server Database Releases			
Version	**Type**	**32/64Bit**	**Service Pack**
2016	Standard	64 Bit	1
	Enterprise	64 Bit	1
2014	Standard	64 Bit	1
	Enterprise	64 Bit	1
2012	Standard	64 Bit	2
	Enterprise	64 Bit	2
2008 R2	Standard	x64	3
		x86	3
	Enterprise	x64	3
		x86	3

Only the listed service packs should be installed. Newer service packs are also supported unless stated otherwise in the *OO Release notes*.
See the Microsoft documentation for supported platforms.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by quality assurance personnel.

Database Release			Operating System
Version	32/64Bit	Service Pack	
Microsoft SQL Server 2016 Enterprise Edition	64 Bit	0	Windows 2012 R2 Standard Edition 64 Bit
Microsoft SQL Server 2014 Enterprise Edition	64 Bit	1	Windows 2012 Standard Edition 64 Bit
Microsoft SQL Server 2012 Enterprise Edition	64 Bit	2	Windows 2012 Standard Edition 64 Bit
Microsoft SQL Server 2008 R2 Enterprise Edition	64 Bit	3	Windows 2012 Standard Edition 64 Bit

Language support

In Microsoft SQL Server, unlike other databases, OO database does not use Unicode collation. Use one of the following collations depending on your OO installation language:

Language	Database Collation
English	SQL_Latin1_General_CP1_CS_AS
Japanese	Japanese_Unicode_CS_AS
Simplified Chinese	Chinese_Simplified_Stroke_Order_100_CS_AS
German	SQL_Latin1_General_CP1_CS_AS
French	French_100_CS_AS
Spanish	SQL_Latin1_General_CP1_CS_AS

Note When an MS SQL database is created with a case-sensitive collation, object names such as tables, keys, and so on become case-sensitive as well.
For example, for the OO_STEP_LOG_BINDINGS table, a command such as `SELECT * FROM oo_step_log_bindings` is seen as using an invalid object name.

If you are currently using a different collation, it is highly recommended that you change your OO

database collation to one of the collations above, in order to support future OO versions.
The following procedure is an example of how to change an existing database collation:

1. Connect to your database server using an administrative login (for example, “sa”).
2. Disconnect all existing sessions to the specific OO database that you intend to change.
   Important! The command will fail unless this database has exactly 0 sessions/connections.
3. Run the following code (change `my_database` to the actual name of the database): USE [master]
   GO
   ALTER DATABASE [my_database] COLLATE Japanese_Unicode_CS_AS
   GO

Note This change does not alter existing column collations. Any new column or table will use the new collation by default from this point on. The new collation sorting rules are applied immediately. In other words, the new collation affects sorting behavior and future data, rather than existing data.

If you have user input in two languages apart from English (for example, German and Chinese) then MS SQL should not be used. You should use an alternative database such as Oracle, MySQL, or Postgres with Unicode configuration.

Related topics

Prepare the database environment

Configure SQL server

This section contains information on Microsoft SQL Server and database configuration settings. You can install an OO database in any SQL Server environment including clustered environments.
Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Microsoft SQL Server 2008R2, 2012, 2014, and 2016	
Server Options/Features	
Configuration Item	**Supported Configuration Options**
Server Configuration Options	Defaults, unless instructed otherwise
Instances	Default, Single
Authentication Mode	Mixed, Windows 1
Full-Text Search	(Not required for OO)

[1] At the moment, the OO 10.x and 2018.0x installer supports only SQL authentication. Windows authentication can be configured later.

Microsoft SQL Server 2008R2, 2012, 2014, and 2016			
Instance/Server Options			
	Mandatory	**Recommended**	**Supported**
Server Collation		**SQL_Latin1_General_CP1_CS_AS**	Any Collation
Network Libraries	Server: **TCP/IP** Client: **TCP/IP**		
Concurrent Connections	**>= 800**	**0** *(unlimited)*	
Max Server Memory	**> 4GB**	**2,147,483,647** *(default, unlimited)* *Allocate 4-12 GB depending on system scale according to the Performance and Sizing Information topic of OO Install section.*	

Microsoft SQL Server 2008R2, 2012, 2014 and 2016
Database Options

	Mandatory	Recommended	Supported
Collation	**Any collation listed in the "Language Support" section in System requirements for Microsoft SQL server topic.**		
Recovery Model		**Full**	Simple, Full
Allow Snapshot Isolation	**True**		
Is Read Committed Snapshot On	**True**		
Auto Shrink	**False**		
Auto Create Statistics	**True**		

Note SQL Server uses a READ-COMMITTED transaction isolation without snapshot isolation by default. It is mandatory to set the **Allow Snapshot Isolation** and **Is Read Committed Snapshot On** flags. OO fails to operate properly using any other type of transaction isolation.

Related topics

Performance and sizing information
System requirements for Microsoft SQL server

Manually create an OO database on Microsoft SQL server

During OO setup, a new database can be created automatically by the OO installer or an existing database can be used.

If during installation, you are authorized to connect to the database server as **sysadmin** (that is, connect as “sa”), use the installer's “Create the database/schema” option, and you can skip this section.

This section describes the procedure for manually creating an OO database, login, and user on Microsoft SQL Server.

Note Only the database, login and user are created at this point; objects such as tables and indexes are not created yet. These objects are created when OO starts for the first time.

This section is relevant for you if, for example, due to security restrictions, you do not wish to use login/user credentials with elevated privileges during the OO installation. In such a case, you (or your organization’s DBA) should create the database, login, and user first, and then let the OO installer connect to the pre-existing database using “lower” privileges.

To create a database, you must connect to the SQL Server instance using a login that has **CREATE DATABASE** permission.

- Members of the sysadmin server roles automatically have **CREATE DATABASE** permission, and are also mapped to dbo in all databases.
- Perform the following procedures only if you are an experienced Microsoft SQL Server database administrator.
- If you prefer to use the database creation wizard/GUI, make sure you select all options that correspond with the T-SQL code presented below. For example, make sure you set **Allow Snapshot Isolation** to TRUE under the **Options** page/**Other Options** pane/**Miscellaneous** tab in the New Database dialog box.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

In order to use Microsoft SQL Server in the most efficient way with OO, it is required that language-specific database collations are used. This enables using the **varchar** data-type for textual columns instead of the **nvarchar** data-type. Using the **varchar** data type is more efficient and reduces overall database size.

Selecting a specific collation also means that an OO system that uses SQL Server is limited to the set of languages supported by that specific collation. For example, if the **SQL_Latin1_General_CP1_CS_AS** collation is used, English, German, and Spanish characters may be used, but Japanese characters may not. If **Japanese_Unicode_CS_AS** is used, French accent

characters will not be presented properly. For the complete specification of each collation, see the Microsoft SQL Server documentation.

To create a database:

1. Log in to Microsoft SQL Server as “sa” or any other login with a **sysadmin** role or the **CREATE DATABASE** permission.
2. Run **<installation folder>/central/bin/sql/MS-SQL/mssql_create_oo_db.sql** and verify that the database, login, and user were created successfully. Adapt the script parameters to match your environment.
3. (Optional) In order to verify that database objects can be created by the new login and user, connect to the database server using OO’s newly created login role credentials and run **<installation folder>/central/bin/sql/MS-SQL/mssql_create_test_table.sql**. Verify that the script ran correctly and that no errors are shown.

Manually create database objects

Once the database, login, and user are in place, the database objects (tables, indexes, and so on) are created when the OO service starts and connects to the database for the first time.

You could create the database objects manually if the user created for OO does not have permissions to create or modify database objects (limited to DML operation only).

It is highly recommended to grant the OO database user DDL-related privileges, in order to allow OO to perform upgrade operations without manual intervention, as these operations sometimes require modifications to the database structure.

To manually create the database objects:

1. Extract the **mssql.sql** files from the OO installation zip file under **docs\sql**.
2. Log in to Microsoft SQL Server as the relevant user, who is permitted to create and modify database objects in the OO database.
3. Run the **mssql.sql** file and verify that no errors occurred.

Microsoft SQL server database maintenance

This section describes the various maintenance tasks that are recommended for OO databases created on Microsoft SQL server, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO database

Microsoft SQL Server databases are either configured for the **Full** recovery model, or the **Simple** recovery model. You can back up an OO database using either one of these models. As OO keeps all configuration and operational history in a single database, always backup the complete database.
Consider the following guidelines when you create your backup plan for OO:

Backup method

The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements.
If your organization is more tolerant to data loss, you can use the simple recovery model and perform a daily or weekly full backup.

Backup frequency

Daily backup is recommended, especially if you are using/modifying OO on a daily basis.
You should back up once a month at the very least.

Timing

Schedule backup for the time when OO is least active.

Retention

Retention depends on your business guidelines and regulations.

Create a maintenance plan

Maintaining an OO database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO database in good shape.

Supplied Utilities for OO Database Housekeeping

OO provides a set of scripts for index maintenance, statistics maintenance, and history purging. These scripts create stored procedures that you can tune and schedule to run periodically.

Note SQL scripts are available in the Central server installation under the **central**/**bin**/**sql** folder.

It is recommended to use these procedures, but you can also use other methods in accordance with company policy, as long as indexes and statistics are kept in good shape.

Note that index rebuilding online (without OO system downtime) requires an enterprise grade database. Make sure you are running an enterprise version of Microsoft SQL Server before attempting online index rebuilding.

Also, note that maintenance activity usually consumes additional resources from the database server. This is why it is important to schedule maintenance activity at the times when OO is least active.

Utility for Maintaining Indexes and Statistics

Download the latest **HPE_OO_DB_Maintenance.zip** pack from ITOM Marketplace and unpack it to a temporary folder.

To install and use the OO maintenance stored procedures:

1. Log in to Microsoft SQL Server as “sa” or any member of the **sysadmin** role and execute:**<installation folder>/central/bin/sql/MS-SQL/mssql_grant_view_server_state.sql** in order to give the OO user the ability to access **dm_os_performance_counters** dynamic management view (DMV).
2. Edit the **OOIndexMaintenance.sql** and (optionally) **OO_DB_MAINTENANCE_LOG.sql** scripts under **<temporary folder>/HPE_OO_DB_Maintenance/MS-SQL/10.60_to_10.80/** and replace “`USE <your_db_name_here>`” at the beginning of the script with your actual OO database name. For example, if your database name is “OOPROD”, replace it with “`USE OOPROD`”. Do not skip this step; otherwise, the set of procedures will not be created in the correct database.
3. Log in to Microsoft SQL Server as the OO user.
4. Run the **OOIndexMaintenance.sql** and (optionally) OO_DB_MAINTENANCE_LOG.sql T-SQL

scripts in the specified order and verify that the new objects were created successfully.
5. Tune your stored procedure according to the explanations below. **<installation folder>/central/bin/sql/MS-SQL/mssql_execute_index_maintenance.sql** provides an example of how the index maintenance procedure can be used.

- **@FragmentationXXX** sets the script's fragmentation level sensitivity and course of action in each case. These threshold levels and subsequent actions are recommended by Microsoft's documentation. Tune these values with caution.
- **@SortInTempdb** (once set to "Y") lets you perform sorting operations during index reorganization/rebuild in tempdb rather than in memory, for better performance. If you choose to use this option, make sure you have sufficient free space in **tempdb**.
- **@Indexes** is a filter for including/excluding indexes in the maintenance operation. It is recommended to keep this filter as is, to analyze all indexes.
- **@TimeLimit** is the timeout in seconds for the maintenance operation to complete. Set it in accordance with your maintenance window boundaries, if applicable.
- **@LockTimeout** is the timeout in seconds to wait for object lock. Once expired, the specific operation fails and the procedure continues to the next object.
- **@LogToTable** determines whether maintenance operation results should be logged to a table. This lets you keep track of the maintenance operations and helps procedure debugging. Note that it requires creating the table using the **OO_DB_MAINTENANCE_LOG.sql** script.
- **@Execute** determines whether actual operations (such as index rebuild) are performed or not. If this parameter is set to 'N', the procedure performs a "dry run" and shows an analysis of the relevant objects.

Utility for Purging Historical Data

To install and use the OO history purging stored procedure:

1. Edit the **OOPurgeHistory.sql** script under **<temporary folder>/HPE_OO_DB_Maintenance/MS-SQL/10.60_to_10.80/** and replace "`USE <your_db_name_here>`" at the beginning of the script with your actual OO database name. For example, if your database name is "OOPROD", replace it with "`USE OOPROD`". Do not skip this step; otherwise, the procedure will not be created in the correct database.
2. Log in to Microsoft SQL Server as the OO user.
3. Run the **OOPurgeHistory.sql** T-SQL script and verify that the new objects were created successfully.
4. Tune your stored procedures according to the explanations below. **<installation folder>/central/bin/sql/MS-SQL/mssql_execute_purge_stored_procedure.sql** provides an example of how the index maintenance procedure can be used.

- **@PurgeExecutionsOlderThan** determines how many days are kept, (protected) relative to the time the procedure starts running. By default, 90 days are kept. Older data is deleted, starting with the oldest records.
- **@PurgeExecutionsInBatchesOf** determines how many flows are handled together. Smaller

values imply smaller, more frequent transactions, and higher values imply less frequent, larger transactions. 1000 is recommended for most systems.

- **@ShouldPurgeExecutionSummary** determines if the **OO_EXECUTION_SUMMARY** table should be purged. The default value is "0" (do not purge this table). It is recommended to keep data in this table as it does not consume a lot of space. Use “1” only if you want to completely remove any reference to the relevant flows.
- **@verbose** determines the verbosity level. "0" corresponds to “quiet” output, "1" corresponds to normal output, and "2" prints out detailed information.
- **@StopPurgingProcessAfter** is the timeout in hours for the operation to complete. Set it in accordance with your maintenance window boundaries if applicable.
- **@DeepClean** determines whether deep cleansing is performed. For example, searching for “orphan” records that may bloat the database unnecessarily. The default is “0” (off). Note that setting this flag to “1” prolongs the procedure run time, but the timeout limit is still imposed.
- **@DisableIndexes** determines whether certain database indexes are disabled for the duration of the purge operation. These indexes are rebuilt at the end of the purge operation. Using this feature speeds up the purge on the expense of flow drill-down (step-level debugging).

Upgrade/Rollback instructions

Prior to any upgrade or rollback of OO, take the following steps:

1. Purge non-essential data from the database. Upgrade and rollback operations convert flow execution data. Less data means a shorter upgrade/rollback. If all flow execution data is essential, perform a complete database backup first and then purge.
2. Back up your database, preferably after the database was purged.
3. Schedule the operation to the database server’s quiet period.
4. Verify that no shared resources are exhausted – temporary space, transaction log, and similar resources.

Following any upgrade or rollback of OO, take the following steps:

1. Install relevant versions of maintenance and purge stored procedures. Stored procedures and purge flows are not upgraded/rolled back by the OO upgrade utility.
2. Reschedule database maintenance and purge routines (stored procedures and/or purge flows).

Database server upgrade from 2012 or 2008R2 to

server 2014 or server 2016

If the OO 10.60x or higher database server is being upgraded from 2012 or 2008R2 to 2014, you must modify the OO Central server configuration in order to use a different JDBC connector jar.

SQL Server 2008R2 and 2012 are interfaced by OO using JTDS 1.3.0 connector, while 2014 or 2016 is interfaced using Microsoft JDBC 4.2.

If your SQL Server database server is being upgraded to 2014 or 2016:

1. Edit central server **database.properties** file located under the **central\conf** folder.
2. Modify the **jdbc.driver_class** value as follows: From: net.sourceforge.jtds.jdbc.Driver To: com.microsoft.sqlserver.jdbc.SQLServerDriver
3. Modify the **jdbc.url** value as follows: From: jdbc:jtds:sqlserver://db_host:db_port/db_name;sendStringParametersAsUnicode=true To: jdbc:sqlserver://db_host:db_port;databaseName=db_name ;sendStringParametersAsUnicode=true If you have edited your JDBC URL and used additional options, use Microsoft's JDBC 4.2 document in order to format the new URL correctly.
4. Repeat all above steps for each central server in the OO system.
5. Restart the OO Central servers.

If OO is being upgraded to 10.6x or higher and the SQL Server database is being upgraded to 2014 simultaneously, follow these steps:

1. Complete the OO upgrade procedure using the current database (2008R2 or 2012). Follow all the instructions in the sections above as usual – purge beforehand, backup, etc. It is highly recommended to perform a sanity test for the OO system using the current database.
2. Make a backup of the OO database (full, copy-only backup is preferred). Backup OO's login role, if necessary.
3. Upgrade the SQL Server database server to 2014 or 2016.
4. Restore OO's database from the backup if necessary (optional). If OO's database was upgraded along with the database server, there is no need to restore it. Reconfigure the login role authentication and database ownership if necessary.

At this point, you must configure Central servers to use Microsoft JDBC instead of JTDS.

Microsoft SQL Server 2008R2 and 2012 "Always On" is supported by OO as a legacy client[1] to implement a high availability and/or disaster recovery solution.

[1] OO does not support the "MultiSubnetFailover" feature for SQL Server 2008R2 and 2012, and requires that **RegisterAllProvidersIP** is set to 0 in the cluster/listener configuration.

Connection to SQL Server 2014 and 2016 "Always On" is supported including the "MultiSubnetFailover" feature, but has not been certified for OO.

High Availability Setup

As OO requires a single point of contact, it is essential that an Availability Group Listener (AG

listener) exist in the high availability setup. OO’s database connection is defined by a single URL - containing a single hostname.
Normally, OO interfaces with the AG listener in order to connect to the availability group primary replica (a READ-WRITE capable instance).
In the case of a database failover scenario, while database connectivity is lost, OO will repeatedly attempt to connect to the database until a new READ-WRITE capable instance accepts the connection. The redirection of the configured database hostname to a different IP address should be done by the AG Listener and cluster environment.

Note OO has no use for a read-only database (thus, has no use for read-only routing).

Disaster Recovery Solution
In a disaster recovery setup, OO should be halted, reconfigured (edit the **database.properties** file), and then started again once the new database instance is available.

Related topics

Additional guidelines for Microsoft SQL server
Back up and recover Operations Orchestration

Deploy and maintain an Oracle database

In order to deploy Operations Orchestration using Oracle, you must have an existing Oracle database service. If you need to create a new database instance/service, see the relevant documentation provided by Oracle, because this information is not included within this section. However, this section contains recommendations for the Oracle instance configuration.
This section includes the following topics:

Workflow for Oracle deployment
System requirements for Oracle
Configure an Oracle database
Manually create an Operations Orchestration database on an Oracle instance
Oracle database maintenance

Workflow for Oracle deployment

To deploy OO using Oracle, perform the following steps:

1. **Review sizing guidelines**. For details, see "*OO database sizing*" in "*Prepare the database environment*" section.
2. **Review Hardware and Software Requirements**. For details, see "*System requirements for Oracle*" section.
3. **Configure an Oracle Database**. For details, see "*Configure an Oracle database*" section.
4. **Create a Database**. For details, see "*Manually create an OO database on an Oracle instance*" section.
5. (Optional) **Connect OO to an Oracle RAC environment**. This step is only relevant if you are using OO in an Oracle RAC environment.

Related topics

- Prepare the database environment
- System requirements for Oracle
- Configure an Oracle database
- Manually create an OO database on Microsoft SQL server

System requirements for Oracle

This section describes the system requirements for working with Oracle in conjunction with Operations Orchestration.

Hardware requirements

For OO database sizing guidelines and hardware requirements, see "*OO database sizing*" section and "*Hardware requirements*" in "*Prepare the database environment*" topic.
For Oracle hardware requirements, see the relevant installation section for your Oracle release and operating system.

Software requirements

The following table lists the Oracle releases supported by OO:

Oracle Releases			
Version	**Type**	**32/64Bit**	**Patch Set**
11g R2	Standard	64 Bit	11.2.0.1 – 11.2.0.4 [1]
	Enterprise	64 Bit	11.2.0.1 – 11.2.0.4 [1]
12c R1 Regular instance, non-CDB	Standard	64 Bit	12.1.0.1 - 12.1.0.2
	Enterprise	64 Bit	12.1.0.1 - 12.1.0.2
12c R2 Regular instance, CDB	Standard	64 Bit	12.2.0.1
	Enterprise	64 Bit	12.2.0.1

Only the listed patch sets should be installed. Newer patch sets are also supported unless stated otherwise in the OO *Release Notes*.

[1] It is highly recommended to apply Oracle's "DATABASE PATCH SET UPDATE 11.2.0.4.6", which solves an issue with rollback to previous OO versions.

See the Oracle documentation for supported platforms.

Oracle 12c R1 is only supported in its regular, backward-compatible instance form. OO 10.x does not support connecting to an Oracle 12c container database (CDB).

The Oracle 12c RAC environment is supported in backward-compatible form. New features such as multiple cluster subnets for SCAN listener are not supported.

OO is compatible with Oracle Data Guard protected databases. For OO to work with a database protected by Oracle Data Guard, ensure that the Data Guard protection mode is either set to “Maximum Availability" or "Maximum protection”. The Data Guard environment must be configured for Fast-Start Failover to limit the down time of the OO service during failovers. The JDBC URL requirements are similar as those for Oracle RAC.

During failover, OO may function in read only mode (that is, no new flow executions can occur) for a short period of time while the database change occurs. Also, during failover, historical data may be out of sync. This will be resolved when the failover completes.

Oracle connector

The `jdbc connector.jar` file is not provided by default. When installing provide the connector to the installer. If upgrading from a previous version of OO the connector will not be deleted.
The `jdbc connector .jar` fie is not provided with the OO installation. You must download it and place it under each `Central <installation folder>/central/lib` folder.
Verify that the connector version you are using is fully compatible with your database server version. The recommended driver is the Oracle JDBC driver version 7-12.1.0.2

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by quality assurance personnel.

Database Release			Operating System
Version	32/64Bit	Patch Set	
Oracle 11g R2 Enterprise Edition	64 Bit	11.2.0.4.6	Red Hat Enterprise Linux 6.5 64 Bit
Oracle 11g R2 Enterprise Edition	64 Bit	11.2.0.4.0	Windows 2012 Standard Edition 64 Bit
Oracle 12c R1 Enterprise Edition	64 Bit	12.1.0.2	Red Hat Enterprise Linux 6.5 64 Bit
Oracle 12c R2 Enterprise Edition	64 Bit	12.2.0.2	Oracle Enterprise linux 7.3 64 Bit

Language support

The Oracle instance character set should be set to **AL32UTF8**. This enables using any Unicode character (and practically all common characters in all languages).

Related topics

Prepare the database environment

Configure an Oracle database

This section contains information on the Oracle instance and database configuration settings. You can install an OO database in an Oracle clustered environment (Oracle RAC or other).
Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Oracle Database 11gR2 / 12cR1 / 12cR2			
Instance/Server Options			
Instance configuration options	**Defaults, unless instructed otherwise**		
	Mandatory	**Recommended**	**Supported**
PROCESSES	**>= 500**		
SESSIONS	**>= 555**		
TIMED_ STATISTICS		**TRUE**	TRUE, FALSE
OPEN_CURSORS	**>= 900**		
Shared/Dedicated Server		'*Dedicated*'	Dedicated, Shared
UNDO_ MANAGEMENT		**AUTO**	Automatic, Manual
Undo size	**>= 4GB**	**6GB – 10GB**	
Memory Management		**ASMM**	AMM, ASMM
MEMORY_TARGET		**0** (disabled)	>= 5G *(for AMM)*
SGA_TARGET		**8G – 12G**	>= 4G *(for ASMM)*
SGA_MAX_SIZE		**8G – 12G**	>= 4G *(for ASMM)*
PGA_AGGREGATE_ TARGET		**1G – 2G**	>= 500M *(for ASMM)*

All values reflect resources required by Operations Orchestration. If OO shares the Oracle instance with other users, these values should be added to whatever is currently consumed by others.

Oracle Database 11gR2 / 12cR1 / 12cR2			
Instance/Server Options			
	Mandatory	**Recommended**	**Supported**
File system			ASM, Any

Storage options		**Locally managed tablespace**	
		Automatic segment space management (ASSM)	
		Automatic local extent management	
ARCHIVELOG mode		**ARCHIVELOG**	ARCHIVELOG, NOARCHIVELOG
Redo Log total size	**>= 600MB**	**1GB**	

All values reflect resources required by OO. If OO shares the Oracle instance with other users, these values should be added to whatever is currently consumed by others.

- See the "*Performance and Sizing Information*" *section in OO Administer Guide*"*Performance and sizing information*" section to determine values displayed as range.

Related topics

Performance and sizing information

Manually create an Operations Orchestration database on an Oracle instance

During OO setup, a new database can be created automatically by the OO installer or a pre-existing database can be used.
If during installation, you are authorized to connect to the database server as dba (connect as “SYSTEM”), use the “create the database/schema” option, and you can skip this section.

Note In some cases, the term "database" is used but in the case of Oracle, it should be interpreted as "user".

This section describes the procedure for manually creating an OO database in an Oracle instance.

Note Only the database is created at this point; objects such as tables and indexes are not created yet. These objects are created once OO starts for the first time.

This section is relevant for you if, for example, due to security restrictions, you do not wish to use user credentials with elevated privileges during the OO installation. In such a case, you (or your organization’s DBA) should create the user (database) first, and then let the OO installer connect to the pre-existing database using basic privileges.
To create a database, you must connect to the Oracle instance using a login that has **CREATE USER** system privilege—for example, system user.

- Any user with the DBA role has sufficient privileges to create the new user.
- Perform the following procedures only if you are an experienced Oracle database administrator.
- If you prefer to use the database creation wizard/GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to the oracle as “system” or any other user with a DBA role.
2. Edit the **<installation folder>/central/bin/sql/Oracle/oracle_create_oo_db.sql** SQL script, run it, and verify that the database was created successfully.
3. (Optional) In order to verify that database objects can be created by the new user, connect to the Oracle instance as OO user, edit and run: **<installation folder>/central/bin/sql/Oracle/oracle_create_test_table.sql** Verify that the script ran successfully, and no errors or warnings were shown.

Manually create database objects

Once the database is in place, the database objects (tables, indexes, and so on) are created when the OO service starts and connects to the database for the first time.

You can create the database objects manually if the user created for OO does not have permissions to create or modify database objects (limited to DML operation only).

It is highly recommended to grant the OO database user DDL-related privileges, in order to allow OO to perform upgrade operations without manual intervention, as these operations sometimes require modifications to database structure.

To manually create the database objects:

1. Extract the **oracle.sql** file from the OO installation zip file under **docs\sql**.
2. Edit the **oracle.sql** file and prefix each object with the OO user, to make sure the objects are created under the OO user.
3. Connect to the OO database as the relevant user, who is permitted to create and modify database objects in the OO database.
4. Run the **oracle.sql** file and verify that no errors occurred, and that all objects are created and owned by the OO user.

Connect to Oracle using SID or service name

Connecting to an Oracle database server requires you to specify either the SID (system ID) or Service Name. In version 10.20 and later, OO supports specifying the service name during installation (using either the Installation Wizard or a silent installation).

The following examples show how to format JDBC URL for each option, as it should appear in the Central **database.properties** file under the **central\conf** folder.

Connecting to an Oracle instance using the SID:

```
jdbc.url=jdbc\:oracle\:thin\:@DB_HOSTNAME_OR_IP\:PORT\:SID
```

Connecting to an Oracle instance using the Service Name:

```
jdbc.url=jdbc\:oracle\:thin\:@//DB_HOSTNAME_OR_IP\:PORT/SERVICE_NAME
```

Oracle database maintenance

This section describes the various maintenance tasks that are recommended for OO databases created on Oracle, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the Operations Orchestration database

Oracle databases can be backed up using several tools, such as **expdp** and **RMAN**. An OO database can be backed up using any type of method/tool as long as the complete database is backed up.
Consider the following guidelines when you create your backup plan for OO:

Backup method

The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every "n" hours depending on your business requirements.

Backup frequency

Daily backup is recommended, especially if you are using/modifying OO on a daily basis.
You should back up once a month at the very least.

Timing

Schedule backup for the time when OO is least active.

Retention

Retention depends on your business guidelines and regulations.

Create a maintenance plan

Maintaining an OO database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO database in good shape.
Supplied Utilities for OO Database Housekeeping
OO provides a set of scripts for index maintenance, statistics maintenance, and history purging. These scripts create a package containing stored procedures that you can tune and schedule to run periodically.
It is recommended to use these procedures, but you can also use other methods in accordance with

company policy, as long as indexes and statistics are kept in good shape.
The index rebuilding online (without OO system downtime) requires an enterprise grade database. Make sure you are running an enterprise version of Oracle before attempting online index rebuilding.
The maintenance activity usually consumes additional resources from the database. This is why it is important to schedule maintenance activity at the times when OO is least active.

Utility for Maintaining Indexes and Statistics

Download the latest **HPE_OO_DB_Maintenance.zip** pack from ITOM Marketplace and unpack it to a temporary folder.

To install and use the OO maintenance stored procedures:

1. Log in to Oracle as “system” or any other user with a DBA role, edit, and execute **<installation folder>/central/bin/sql/Oracle/oracle_privs_for_index_maintenance.sql**. These system privileges are required in order to verify that the stored procedure created in the following steps has the explicit (not role-based) privileges to execute the index analysis and rebuild.
2. Open the **<temporary folder>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80/** folder, log in to Oracle database as the user created for OO, and run the **HPE_OO_DB_MAINT.sql** script. Verify that the new package and procedures were created successfully.In the same folder, **oracle_execute_IndexMaintenance.sql** provides an example of how this procedure can be used.

- **pMaxHeight (IN)** - The minimal index height threshold for index rebuilding. The Oracle documentation recommends 3. Smaller values may result in unnecessary rebuilding operations.
- **pMaxLeafsDeleted (IN)** - The minimal deleted leaves threshold for index rebuilding. The Oracle documentation recommends 15. Smaller values may result in unnecessary rebuilding operations.
- **pRebuild (IN)** - Should indexes be rebuilt (1) or only perform a dry-run (0). A dry-run will show only recommendations for index rebuilding.
- **pReturnValue (OUT)** - The number of rebuilt indexes.

Note ONLINE index rebuilding should only be performed when the enterprise edition is used. Otherwise, the index rebuilding operation may lock tables and indexes and may interfere with the operation of OO.

Utility for Purging Historical Data

To install and use the OO history purging stored procedure:

1. Open the **<temporary folder>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80** folder, log in to Oracle database as the user created for OO, and run **HPE_OO_DB_MAINT.sql** script. Verify that the new package and procedures were created successfully.In the same folder, **oracle_execute_PurgeHistory.sql** provides an example of how this procedure can be used.
   - **pPurgeExecutionsOlderThan** determines how many days are kept (protected) relative to the time the procedure starts running. Older data is deleted, starting with the oldest records. This parameter has no default value and must be specified.
   - **pPurgeExecutionsInBatchesOf** determines the maximum number of flows to handle in each batch. The default value is 10000.

- **pShouldPurgeExecutionSummary** determines if the **OO_EXECUTION_SUMMARY** table should be purged. The default value is "0" (do not purge this table). It is recommended to keep data in this table as it does not consume a lot of space. Use “1” only if you want to completely remove any reference to the relevant flows.
- **pVerbose** determines verbosity level. "0" corresponds to “quiet” output, "1" corresponds to normal output, and "2" prints out detailed information. Set it in accordance with your maintenance window boundaries if applicable.
- **pStopPurgingProcessAfter** is the timeout in hours for the operation to complete.
- **pDeepClean** determines whether deep cleansing is performed. For example, searching for “orphan” records that may bloat the database unnecessarily. The default is “0” (off). Note that setting this flag to “1” prolongs the procedure run time, but the timeout limit is still imposed.
- **@DisableIndexes** determines whether certain database indexes are disabled for the duration of the purge operation. These indexes are rebuilt at the end of the purge operation. Using this feature speeds up the purge at the expense of flow drill-down (step-level debugging).

Utility for Reclaiming LOB Space

Oracle databases are slow when it comes to freeing space occupied by LOB segments. Depending on the configuration of the database server, LOB segment space may not be released immediately after purge operations. As LOB space may have a significant impact on the database size, you can add a post-purge operation to force freeing of LOB segments.

The general structure of the command is:

ALTER TABLE FOO MODIFY LOB (LOB_COLUMN_NAME) (shrink space);

The list of tables and columns involved may vary from one OO version to another.

To install and use the OO LOB space reclaim stored procedure:

Open the **<temporary folder>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80** folder, log in to Oracle database as the user created for OO, and run **HPE_OO_DB_MAINT.sql** script. Verify that the new package and procedures were created successfully.

In the same folder, **oracle_execute_ForceLobShrink.sql** provides an example of how this procedure can be used.

Notes:

Several conditions must be met in order for the shrink operation to succeed:

- TABLESPACE and database tables must not be compressed.
- ROW MOVEMENT must be enabled for all tables.
- TABLESPACE must be configured as AUTO segment space management.
- LOBs storage must not be SECUREFILE.

See the Oracle documentation for the complete set of conditions.

Shrinking LOB segments of considerable size (tenths or hundreds of Giga-Bytes) may take some time to complete and may consume a considerable amount of database resources—CPU and I/O. Therefore, it is recommended to execute this process when the database server is least active.

Upgrade/Rollback Instructions

Rolling back OO 10.5x and later to previous versions requires patching Oracle 11gR2 database servers to 11.2.0.4.6. Apply Oracle's "DATABASE PATCH SET UPDATE 11.2.0.4.6" prior to rollback operation.

Prior to any upgrade or rollback of Operations Orchestration, take the following steps:

1. Purge non-essential data from the database.Upgrade and rollback operations convert flow execution data. Less data means a shorter upgrade/rollback. If all flow execution data is essential, perform a complete database backup first and then purge.
2. Back up your database, preferably after the database was purged.
3. Schedule the operation to the database server's quiet period.
4. Verify that no shared resources are exhausted – temporary space, UNDO/REDO, and similar resources.

Following any upgrade or rollback of OO, take the following steps:

1. Install relevant versions of maintenance and purge stored procedures.Stored procedures and purge flows are not upgraded/rolled back by the OO upgrade utility.
2. Reschedule database maintenance and purge routines (stored procedures and/or purge flows).

Deploy and maintain a MySQL database

In order to deploy Operations Orchestration using MySQL, you must have an existing MySQL database. If you need to create a new database service, see the relevant documentation provided by MySQL, because this information is not included within this section. However, this section contains recommendations for the MySQL configuration.

This section includes the following topics:

Workflow for MySQL deployment
System requirements for MySQL
Configure MySQL
Manually create an Operations Orchestration database on MySQL
MySQL database maintenance

Workflow for MySQL deployment

To deploy Operations Orchestration (OO) using MySQL, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure MySQL** .
4. **Create an OO database on MySQL** .

Related topics

Prepare the database environment
System requirements for MySQL
Configure MySQL
Manually create an OO database on MySQL

System requirements for MySQL

This section describes the system requirements for working with MySQL in conjunction with OO.

Hardware requirements

For OO database sizing guidelines and hardware requirements, see "*OO database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" topic.
For MySQL hardware requirements, see the relevant installation section for your MySQL release and operating system.

Software requirements

The following table lists the MySQL releases supported by OO:

MySQL Database Releases		
Version	**Type**	**32/64Bit**
5.5	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit
5.6	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit
5.7	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit

Note
For MySQL 5.6.20 and 5.6.21, the requirements for the **innodb_log_file_size** have increased significantly.
For MySQL 5.6.1 - 19, the recommendation is 256M, but for MySQL 5.6.20 - 21, the recommendation is 2GB.

See the MySQL documentation for supported platforms.

MySQL connector

The MySQL connector jar is not provided with the OO installation. Download and place it under each Central **<installation folder>/central/lib** folder.
Verify that the connector version you are using is fully compatible with your database server version. Currently, the only supported version is 5.1.35.
Upgrade Note: Because prior to version 10.60, MySQL JDBC 5.1.21 was in use, make sure to download and use 5.1.35 during the upgrade process.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by quality assurance personnel.

Database Release			Operating System
Version	32/64Bit	Patch	
MySQL Server 5.6.19 Community Edition	64 Bit		Windows 2012 Standard Edition 64 Bit
MySQL Server 5.6.12 Community Edition	64 Bit		Red Hat Enterprise Linux 6.3 64 Bit

Language support

The MySQL Server character set should be set to `utf8`. This lets you use any Unicode character (and practically all common characters in all languages). Note that the OO database uses the `utf8_bin` collation.

Related topics

Prepare the database environment

Configure MySQL

This section contains information on MySQL and database configuration settings.
Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

MySQL 5.5, 5.6 and 5.7			
Instance/Server Options			
Server Configuration Options	Defaults, unless instructed otherwise		
[mysqld]	**Mandatory**	**Recommended**	**Supported**
character-set-server	utf8		
collation-server	utf8_bin		
transaction-isolation	READ-COMMITTED		
max-allowed-packet	250M		
max-connections	>= 1000		
default-storage-engine	INNODB		
innodb_log_file_size (for MySQL 5.6.1 - 19)	256M		
innodb_log_file_size (for MySQL 5.6.20 - 21)	2GB		
innodb_log_file_size (for MySQL 5.6.22 - 5.7)	256M		
explicit-defaults-for-timestamp	0		
sql-mode	See below		
explicit_defaults_for_timestamp	0 (off/inactive) Deprecated in some releases		
innodb_file_per_table		1	
innodb_thread_concurrency		0	
table_open_cache		1000	
sort_buffer_size		2M	
read_buffer_size		2M	
tmp_table_size		400M	
max_heap_table_size		400M	
innodb_buffer_pool_size		4096M	
innodb_additional_mem_pool_size		20M	
binlog_format		row	
innodb_flush_log_at_trx_commit		2	

innodb_flush_method		O_DIRECT **(Linux Only)**	
innodb_doublewrite		0	
MySQL 5.5, 5.6 and 5.7			
Other Options			
Server Configuration Options	Defaults, unless instructed otherwise		
	Mandatory	**Recommended**	**Supported**
[client]			
default-character-set	**utf8**		
[mysql]			
default-character-set	**utf8**		
[mysqldump]			
max_allowed_packet	**250M**		

- MySQL options may be written using underscore or hyphen in different contexts. Use the correct form according to your server version and usage context.

Manually create an Operations Orchestration database on MySQL

During OO setup, a new database can be created automatically by the OO installer or a pre-existing database can be used.

If during installation, you are authorized to connect to the database server using the DBA role (connect as “root”), use the “create the database/schema” option, and you can skip this section.

This section describes the procedure for manually creating an OO database on MySQL.

Note Only the database and user are created at this point; objects such as tables and indexes are not created yet. These objects are created once OO starts for the first time.

This section is relevant for you if, for example, due to security restrictions, you do not wish to use login credentials with elevated privileges during theOO installation. In such a case, you (or your organization’s DBA) should create the database first, and then let the OO installer connect to the pre-existing database using basic privileges.

To create a database, you must connect to the SQL Server instance using a user that has **CREATE** permission (at the very least).

- **root** has all privileges. Any member of the DBA role will also be able to create the user and database.
- Perform the following procedures only if you are an experienced MySQL database administrator.
- If you prefer to use the MySQL Workbench GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to the MySQL as “root” or any other member of the DBA role.
2. Edit and run **<installation folder>/central/bin/sql/MySQL/mysql_create_oo_db.sql** and verify that the database and user were created successfully. Adapt the script parameters to match your environment.

```
SET @OODB='OODB';
SET @OOUSER='OOUSER';
SET @OOPASS='OOPa55WorD';

SET @SQL1 = CONCAT('CREATE DATABASE IF NOT EXISTS `',@OODB,`` COLLATE utf8_
bin');
SET @SQL2 = CONCAT('CREATE USER ''',@OOUSER,'''@''%'' IDENTIFIED BY
''',@OOPASS,'''');
SET @SQL3 = CONCAT('GRANT ALL PRIVILEGES ON `',@OODB,`.* TO
''',@OOUSER,'''');
PREPARE stmt1 FROM @SQL1;
PREPARE stmt2 FROM @SQL2;
PREPARE stmt3 FROM @SQL3;
EXECUTE stmt1;
EXECUTE stmt2;
EXECUTE stmt3;
DEALLOCATE PREPARE stmt1;
DEALLOCATE PREPARE stmt2;
DEALLOCATE PREPARE stmt3;
FLUSH PRIVILEGES;
```

3. Test your newly created connection to the database and verify that you are able to log in successfully.
4. (Optional) In order to verify that database objects can be created by the new login and user, connect to the database server using OO's newly created login role credentials and run:<installation folder>/central/bin/sql/MySQL/mysql_create_test_table.sqlVerify that the script ran correctly and that no errors are shown.

```
USE OO;

CREATE TABLE TEST_TABLE(
       TEST_COLUMN int NULL
);

INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 1 );
INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 2 );
```

Manually create database objects

Once the database and user are in place, the database objects (tables, indexes, and so on) are created when theOO service starts and connects to the database for the first time.

You can create the database objects manually if the user created for OO does not have permissions to create or modify database objects (limited to DML operation only).

It is highly recommended to grant the OO database user DDL-related privileges, in order to allow OO to perform upgrade operations without manual intervention, as these operations sometimes require modifications to database structure.

To manually create the database objects:

1. Extract the **mysql.sql** file from the OO installation zip file under **docs\sql**.
2. Connect to the OO database as the relevant user, permitted to create and modify database objects in the OO database.
3. Run the **mysql.sql** file and verify that no errors occurred.

MySQL database maintenance

This section describes the various maintenance tasks that are recommended for OO databases created on MySQL, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO database

You can back up MySQL database using several tools, such as **mysqldump** or **mysqlbackup**. You can back up the OO database using any type of method/tool as long as the complete database is backed up.

Consider the following guidelines when you create your backup plan for OO:

- **Backup method:**The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements.
- **Backup frequency:**Daily backup is recommended, especially if you are using/modifying OO on a daily basis. You should back up once a month at the very least.
- **Timing:**Schedule backup for the time when OO is least active.
- **Retention:**Retention depends on your business guidelines and regulations.

Creating a Maintenance Plan

Maintaining an OO database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO database in good shape.

Recommended Utility for Database Maintenance

In order to keep the OO database in good shape, it is recommended to schedule **mysqlcheck** utility to run during a system maintenance window.

Important! This operation locks tables. Only perform this action during a maintenance window when the OO system is not operating.

Below is an example of how to run this utility:

```
mysqlcheck -uoouser –p????? -os --auto-repair OO
```

Replace "oouser" and "OO" with the actual OO user name and database name, respectively.

It is recommended not to provide the password explicitly. See the MySQL documentation for recommendations on how to secure database passwords.

Upgrade/Rollback instructions

Prior to any upgrade or rollback of OO, take the following steps:

1. Purge non-essential data from the database.Upgrade and rollback operations convert flow execution data. Less data means a shorter upgrade/rollback. If all flow execution data is essential, perform a complete database backup first and then purge.
2. Back up your database, preferably after the database was purged.
3. Schedule the operation to the database server’s quiet period.
4. Verify that no shared resources are exhausted – temporary space, transaction log, and similar resources.
5. Make a backup copy of your MySQL connector jar file. Consider using a newer one for upgrade scenarios.

Following any upgrade or rollback of OO, take the following steps:

1. Install relevant versions of maintenance and purge stored procedures.Stored procedures and purge flows are not upgraded/rolled back by the OO upgrade utility.
2. Reschedule database maintenance and purge routines (stored procedures and/or purge flows).

Deploy and maintain a Postgres database

In order to deploy (OO) using Postgres, you must have an existing Postgres database service. If you need to create a new database service, see the relevant documentation provided by Postgres, because this information is not included within this section. However, this section contains recommendations for the Postgres configuration.

This section includes the following topics:

Workflow for Postgres deployment
System Requirements for Postgres
Configuring Postgres
Manually create an Operations Orchestration database on Postgres
Postgres database maintenance

Workflow for Postgres deployment

Workflow for Postgres deployment
To deploy OO using Postgres, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure Postgres**.
4. **CreateOOdatabase on Postgres**.

Related topics

Prepare the database environment
Configure Postgres
Manually create an OO database on Postgres

System Requirements for Postgres

This section describes the system requirements for working with Postgres in conjunction with OO.

Hardware requirements

For OO database sizing guidelines and hardware requirements, see "*OO database sizing*" and "*Hardware Requirements*" in *Prepare the database environment* topic.
For Postgres hardware requirements, see the relevant installation section for your Postgres release and operating system.

Software requirements

The following table lists the Postgres releases supported by OO:

Postgres Database Releases	
Version	**Type**
9.3	x86 32-bit
	x86 64-bit
9.4	x86 32-bit
	x86 64-bit
9.5	x86 32-bit
	x86 64-bit
9.6	x86 32-bit
	x86 32-bit

Only supported versions should be used.
See the Postgres documentation for supported platforms.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by the quality assurance personnel.

Database Release		Operating System
Version	**32/64Bit**	
Postgres 9.3.2	64 Bit	Red Hat Enterprise Linux 6.3 64 Bit
Postgres 9.4.6	64 Bit	Red Hat Enterprise Linux 7 64 Bit
Postgres 9.5.1	64 Bit	Red Hat Enterprise Linux 7 64 Bit
Postgres 9.6.3	64 Bit	Windows 2012 R2

Language support

Postgres determines character set and collation at the database level. The OO database uses Unicode (utf8) encoding and collation. This lets you use any Unicode character (and practically all common characters in all languages).

Related topics

Prepare the database environment

Configuring Postgres

This section contains information on Postgres configuration settings.
Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Postgres 9.3 - 9.6			
Instance/Server Options			
Instance Configuration Options	Defaults, unless instructed otherwise		
	Mandatory	Recommended	Supported
max_connections	>= 1000		
default_transaction_isolation	'read committed'		
autovacuum	on		
track_counts	on		
shared_buffers	>=512MB 1		
effective_cache_size	>=2048MB 1		
work_mem	>=1MB 1		
maintenance_work_mem	**>=32MB** 1		
lc_messages		'en_US.UTF-8'	Any
lc_monetary		'en_US.UTF-8'	Any

[1] - Minimal values. See the Postgres documentation on how to tune these values in accordance with your environment.

Manually create an Operations Orchestration database on Postgres

Manually create an OO database on Postgres

During OO setup, a new database can be created automatically by the OO installer or a pre-existing database can be used.

If during installation, you are authorized to connect to the database server as a privileged user (connect as “postgres”), use the “create the database/schema” option, and you can skip this section.

This section describes the procedure for manually creating an OO database on Postgres.

Note Only the database and role are created at this point; objects such as tables and indexes are not created yet. These objects are created once OO starts for the first time.

This section is relevant for you if, for example, due to security restrictions, you do not wish to use login/user credentials with elevated privileges during the OO installation. In such a case, you (or your organization’s DBA) should create the database, login, and user first, and then let the OO installer connect to the pre-existing database using basic privileges.

To create a database, you must connect to the Postgres instance using a login that has **CREATEUSER** and **CREATEDB** privileges at the very least.

- The **postgres** built-in user has all the required privileges.
- Perform the following procedures only if you are an experienced Postgres database administrator.
- If you prefer to use the PgAdmin GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to Postgres as “postgres” or any other login role with **CREATEUSER** and **CREATEDB** privileges.
2. Edit and run:<installation folder>/central/bin/sql/PostgreSQL/postgres_create_oo_db_linux.sqlor: postgres_create_oo_db_windows.sqlVerify that the database, login, and user were created successfully. Adapt the script parameters to match your environment.

```
CREATE ROLE "oouser" LOGIN
UNENCRYPTED PASSWORD '???????'
NOSUPERUSER INHERIT NOCREATEDB NOCREATEROLE NOREPLICATION;

CREATE DATABASE "OO"
WITH OWNER = "oouser"
ENCODING = 'UTF8'
TABLESPACE =
LC_COLLATE =
LC_CTYPE =
CONNECTION LIMIT = 1000;
```

3. (Optional) In order to verify that database objects can be created by the new login and user, connect to the database server using OO's newly created login role credentials and run:<installation folder>/central/bin/sql/PostgreSQL/postgres_create_test_table.sqlVerify that the script ran correctly and that no errors are shown.

```
CREATE TABLE TEST_TABLE(
       TEST_COLUMN int NULL
);

INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 1 );
INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 2 );
```

Manually create database objects

Once the database and role are in place, the database objects (tables, indexes, and so on) are created when the OO service starts and connects to the database for the first time.

To manually create the database objects (instead of the OO service):

1. Extract the **postgres.sql** file from the OO installation zip file under **docs\sql**.
2. Connect to the OO database as the OO database user.
3. Run the **postgres.sql** file and verify that no errors occurred.

Postgres database maintenance

Postgres database maintenance

This section describes the various maintenance tasks that are recommended for OO databases created on Postgres, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO database

You can back up a Postgres database using several tools, such as the **pg_dump** or **pg_backup** script. You can back up the OO database using any type of method/tool as long as the complete database is backed up.

Consider the following guidelines when you create your backup plan for OO:

- **Backup method:**The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements. If your organization is more tolerant to data loss, you can use the simple recovery model and perform a daily or weekly full backup.
- **Backup frequency:** Daily backup is recommended, especially if you are using/modifying OO on a daily basis. You should back up once a month at the very least.
- **Timing:** Schedule backup for the time when OO is least active.
- **Retention:** Retention depends on your business guidelines and regulations.

Create a maintenance plan

OO Postgres database maintenance mainly includes table REINDEX, as **autovacuum** needs to be activated. Use the example below, in order to keep the OO database in good shape.

Recommended Utility for Database Maintenance

In order to keep the OO database in good shape, it is recommended to run the REINDEX action during a system maintenance window.

This operation locks tables. Perform this action only during the maintenance window when the OO system is not operating.

Here is an example of how to REINDEX a complete database using the **reindexdb** utility:

```
reindexdb -d OO –U oouser –W ?????
```

Replace “OO” and “oouser” with actual OO database and user names and "?????" with OO DB user's password.

It is recommended not to provide the password explicitly. See the Postgres documentation for recommendations on how to secure database passwords.

Utility for purging historical data

OO 10.x keeps all flow and step execution data in the database by default, enabling comprehensive debugging of previous flow runs. As a result, the database size grows in accordance to the system throughput and flow complexity. It is highly recommended to track your database size and make sure that old, irrelevant information is periodically purged.

Reclaiming free space in Posgtres database requires two phases:

1. Space is marked as deleted – following a DELETE command.
2. A background “vacuum” process runs to allow free space to be reused.

After the execution of the purge procedure described below, the vacuum process will usually spring into action and start “vacuuming” tables and indices.

You must allow the vacuum process to complete successfully before starting another purge. If the purge and vacuum overlap, they stall each other as they compete for object locks.

“Vacuum Full” Operation

PostgreSQL database is only able to reclaim Large Objects (LOBs) space when the **Vacuum Full** operation is run. A considerable amount of the OO database space is stored as LOBs. Therefore, it is important to schedule a routine **Vacuum Full** operation - at times in which OO is least active. The **Vacuum Full** operation requires *exclusive table locks*, and therefore may interfere with activity of OO.

To install and use the OO history purging stored procedure:

1. Download the latest **HPE_OO_DB_Maintenance.zip** pack from ITOM Marketplace and unpack it to a temporary folder. Follow the instructions in the *OOAdministration Guide*"*Set up database environment*" section to install and execute a purge stored procedure for PostgreSQL.

```
SELECT OOPurgeHistory(90,10000,0,1,4,0);
```

Upgrade/Rollback instructions

Prior to any upgrade or rollback of OO, take the following steps:

1. Purge non-essential data from the database.Upgrade and rollback operations convert flow execution data. Less data means a shorter upgrade/rollback. If all flow execution data is essential, perform a complete database backup first and then purge.
2. Verify that you have sufficient disk space to support a “vacuum full" operation, and initiate a “vacuum full” operation.
3. Back up your database, preferably after the database was purged.
4. Schedule the operation to the database server’s quiet period.
5. Verify that sufficient free disk space is available for transaction log files.

Following any upgrade or rollback of OO, take the following steps:

1. Install the relevant version of the purging stored procedure.Stored procedures and purge flows are not upgraded/rolled back by the OO upgrade utility.
2. Reschedule the database purge routines (stored procedure and/or purge flows).
3. Reschedule the routine database “vacuum full” operations.

Related topics

Set up database environment

Database purging

Database purging

Purge flows

Purge flows are provided as content in the (Undefined variable: _HPb_Branding_Variables._HP_Company_Acronym) Solutions content pack, and are the preferred method for purging the OO database. As long as purge flows provide sufficient “throughput” (are able to delete a sufficient amount of rows in a given amount of time), it is recommended to use them instead of the purge stored procedures.

Purge flows have several advantages over stored procedures:

- Manageability – The OO administrator can easily schedule and track purge operations.
- Database privileges – No need for assistance from the DBA in installing and scheduling purge operations.
- Purge functionality upgrades – When OO is upgraded, and a new table requires purging, this capability will be added to an existing flow or a new one will be created for that purpose.New stored procedures, however, must be manually downloaded and installed.Note For each purge type (rerun, steplog, and so on), you can only run one purge flow at a time.

Purge best practices

Relating to purge operations, it is important to distinguish between two separate criteria:

- Retention – the age of the data you wish to keep
- Purge frequency – how often the purge operations are executed

These two criteria are completely unconnected. You can keep one years worth of data, while purging every 30 minutes.

As the retention period is often determined as a tradeoff between database size and business requirements, the purge frequency can vary between hourly, daily, or even weekly. The recommended purge frequency is between “hourly” and “daily”.

Purging smaller amounts of data frequently proves (in most cases) to be a better strategy than purging large amounts of data during a weekly or monthly maintenance window.

Here is a suggested workflow for optimizing purge frequency:

- Schedule a recurring purge operation for every four hours and monitor it:
  - How long does it take for the purge operation to complete (average and max. duration)?
  - What was the effect on database performance?
  - What was the effect on Operations Orchestration’s performance (user interface and flow execution)?
- As long as purge operations complete in under 30 minutes and no dramatic effect was registered

by the DBA or the database, keep this frequency.You may also try frequencies from hourly to daily and see which works best.Note that some effect on the database and OO is considered normal.

- Otherwise, if OO’s activity changes considerably throughout the day, and consistent “quiet periods” exist, schedule purge operations to run during these periods.

Additional guidelines for Microsoft SQL server

This appendix contains additional guidelines relevant for a DBA’s deployment on Microsoft SQL Server.

Use Windows authentication to access Microsoft SQL server databases

Unless configured otherwise, DBAs use Microsoft SQL Server authentication to access Microsoft SQL Server databases. Note that the OO installer currently does not support using Windows authentication during the OO installation. However, Windows authentication can be used once OO is installed.

This appendix describes how to enable OO to use Windows authentication to access Microsoft SQL Server databases.

Configure OO to work with Windows authentication

You can enable OO to use Windows authentication instead of Microsoft SQL Server authentication to access OO databases.

To enable OO to use Windows authentication to access a Microsoft SQL database:

1. Encrypt the Windows user password using the **encrypt-password.bat** utility located under **<OO installation>/central/bin** by running:encrypt-password.bat --encrypt --password <password>Save the generated string in order to use it in the next step.
2. Back up your current **database.properties** file located under **<OO installation>/central/conf** if you have an existing (usable) database connection.
3. Edit the **database.properties** file located under **<OO installation>/central/conf**, and change only the relevant parameter syntax to match the following example.Note When copying scripts, note that you may need to remove redundant line breaks from the copied version.For SQL Server 2008R2 and 2012:db.username=[USERNAME]
   db.password=[the string generated by encrypt-password.bat]
   jdbc.url=jdbc\:jtds\:sqlserver\://[DB_HOSTNAME]\:[PORT]/[DB_NAME]
   ;sendStringParametersAsUnicode\=true
   ;domain\=[DOMAIN NAME]
   For SQL Server 2014 and 2016: db.username=[USERNAME] db.password=[the string generated by encrypt-password.bat]
   jdbc.url=jdbc\:sqlserver\://[DB_HOSTNAME]\:[PORT];databaseName\=[DB_NAME];sendStringParametersAsUnicode\=true;integratedSecurity\=true; For SQL server 2014 and 2016, there might be a need to copy **sqljdbc_auth.dll** to the system path of the central server machine (i.e. `<central installation>\java\bin`). For more details, see the Microsoft JDBC documentation. `sqljdbc_auth.dll` is located in `<central installation>\java\bin`. Replace all the highlighted items with the correct values that match your environment. Note that

the **jdbc.url** parameter is broken in this document into several lines for readability. When copying this example onto your **database.properties** file, remove all line breaks so the **jdbc.url** parameter is composed of a single line containing no white spaces.

Configure OO to work with Always On

For SQL Server 2008R2 and 2012

OO connects to SQL Server 2008R2 and 2012 databases using the jtds connector. Thus, it is only able to register a single IP address for the Availability Group Listener. In addition, OO does not support Multi-Subnet Failover.

Cluster Configuration

Configure your cluster environment as follows:

1. Set the cluster option **RegisterAllProvidersIP** to 0 for the network name associated with the Availability Group Listener.
2. Optionally, decrease the **HostRecordTTL** value for the same cluster resource to reduce database reconnection time.

See the Microsoft SQL Server Documentation (“Create or Configure an Availability Group Listener”) for a complete description of configuration options and examples.

Installing Operations Orchestration using Availability Group Listener

Prior to installing OO, verify that the Availability Group Listener network name resolves to an accessible IP address from all Central machines.

1. In the OO installation wizard **Database Connection Configuration** page, enter the Availability Group Listener network name in the **Hostname or IP address** field.
2. Enter the rest of the details and click the **Test Connection** button.

Providing the Availability Group Listener network name guarantees that the OO Central server(s) will be able to reconnect to the database in the case of a failure.

For silent installation, use:

```
db.url=jdbc:jtds:sqlserver://[AG-NET-NAME]:[PORT]/[DB-
NAME];sendStringParametersAsUnicode=true
```

For SQL Server 2014 and 2016

OO connects to SQL Server 2014 and 2016 databases using the Microsoft JDBC 4.2 connector.

Here is an example on how to format JDBC URL in order to connect to an Always On cluster:

```
db.url=jdbc:sqlserver://[AG-NET-NAME];instanceName=[NAMED-INST-NAME];
```

```
databaseName=[DB-
NAME];multiSubnetFailover=true;applicationIntent=ReadWrite;
sendStringParametersAsUnicode=true
```

multiSubnetFailover and **applicationIntent** are added in order to allow more complex behaviors.
Multi-Subnet Failover scenarios in conjunction with SQL Server 2014 Always On cluster were not certified with OO.
For additional information and connection options, refer to "*JDBC Driver Support for High Availability, Disaster Recovery*" in the Microsoft documentation and "*Setting the Connection Properties*" in the Microsoft JDBC documentation.

T-SQL Scripts and Stored Procedures

Use the following T-SQL scripts in order to create the OO maintenance-related stored procedures.
OO_DB_MAINTENANCE_LOG.sql
Run the following script in order to create the **OO_DB_MAINTENANCE_LOG.** table. This table will log all maintenance action results.
Creating this table is optional. You can run the **OOIndexMaintenance** stored procedure with `LogToTable = 'N'` in order to prevent logging to this table.
OOIndexMaintenance.sql
In order for this procedure to operate properly, an additional preceding step is required.
Log in to Microsoft SQL Server as "sa" or any member of the sysadmin role and run the following code in order to give the OO user the ability to access **dm_os_performance_counters dynamic management** view (DMV):

```
USE [master]
GO
GRANT VIEW SERVER STATE TO oouser
GO
```

Replace "oouser" with the actual user created for OO.
Run the script in order to create the **OOIndexMaintenance** stored procedure.
OOPurgeHistory.sql
Run this script in order to create the **OOPurgeHistory** stored procedure.

Additional guidelines for Oracle

Additional guidelines for Oracle

This appendix contains the configuration that needs to be done for OO to work with Oracle 11gR2 and 12cR1 Real Application Cluster (RAC) environments. This information is for advanced users only.

Oracle RAC

A cluster is a collection of interconnected servers that appear as one server to the end user and to applications. Oracle Real Application Cluster (RAC) is Oracle's solution for high availability, scalability, and fault tolerance. It is based on clustered servers that share the same storage.

Oracle RAC is a single Oracle database installed on a cluster of hardware servers. Each server runs an instance of the database and all the instances share the same database files.

For more details about Oracle RAC, see the Oracle Clusterware Guide and the Oracle Real Application Clusters Administration and Deployment Guide in the Oracle documentation set of your release.

In this appendix, the following Oracle RAC example is used:

- Oracle RAC Cluster name: OORAC
- Service Name: ORCL.MY.DOMAIN
- Machine names: Server1, Server2
- On each machine, there is an Oracle instance of OORAC:
  - SID on Server1: OORAC1
  - SID on Server2: OORAC2
- On each machine, there is a virtual IP (Server1-Vip and Server2-Vip):
  - Server1-Vip is assigned to Server1
  - Server2-Vip is assigned to Server2

  The virtual IP is in addition to the static IP assigned to the machine.
- SCAN listener uses a virtual IP, usually published using DNS/GNS:SCAN-Vip
- The local listeners on both servers listen on the default port 1521 and support the database service OORAC. SCAN listener(s) reside on one of the cluster nodes, and may failover along with their virtual IP addresses in the case of a failure.Note Although 12c SCAN allows configuring multiple SCAN listeners (per subnet), OO only supports connecting to a single 12c SCAN listener.

Single Client Access Name

In release 11g, Oracle introduced the Single Client Access Name (SCAN), as a preferred access method for clients connecting to the RAC. In this method, clients are not required to configure individual nodes in the RAC; rather, they use a single virtual IP known as the SCAN or the SCAN VIP.

The SCAN is a single network name defined for the cluster either in your organization's Domain Name Server (DNS) or in the Grid Naming Service (GNS) that rotates between several IP addresses,

reflecting multiple listeners in the cluster. The SCAN eliminates the need to change clients when nodes are added to or removed from the cluster.

The SCAN and its associated IP addresses provide a stable name for clients to use for connections, independent of the nodes that make up the cluster. 11g SCAN addresses, virtual IP addresses, and public IP addresses must all be on the same subnet. 12c SCAN allows configuring multiple SCAN listeners – one per subnet. OO only supports connecting to a single 12c SCAN listener.

The SCAN method is recommended when using OO in an Oracle 11g RAC environment.

Configure OO to work with Oracle RAC

Connecting to a SCAN Listener Virtual IP

1. In the OO installation wizard **Database Connection Configuration** page, enter the SCAN listener virtual IP address/network name in the **Hostname or IP address** field.
2. Select the **Service Name** radio button instead of the **SID** radio button and enter the Oracle RAC service name.
3. Enter the rest of the details and click the **Test Connection** button.

Providing the SCAN listener virtual IP address guarantees that the OO Central server(s) will be able to reconnect to the database cluster in the case of a failure.

This connection method is the preferred one as it enables the OO installation wizard to create the database as well as populate it.

For silent installation use (replace the highlighted text with the actual values):

```
db.url=jdbc:oracle:thin:@//[SCAN-Vip]:[PORT]/[ORCL.MY.DOMAIN]
```

Load-Balancing Using Explicit Connection String

Although using Oracle's SCAN listener virtual IP is the preferred method, you can also provide an explicit connection string by using the **Other database** installation option.

Here is an example of how a load balancing connection string would appear in Central's **database.properties** file.

1. Back up your current database.properties file located under **<OO installation>/central/conf** if you have an existing (usable) database connection.
2. Edit the **database.properties** file as follows, while replacing the highlighted items with the values that match your environment.Note When copying scripts, note that you may need to remove redundant line breaks from the copied version.jdbc.url=jdbc\:oracle\:thin\:@ (DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST= (ADDRESS=(PROTOCOL=TCP)(HOST=Server1-Vip)(PORT=1521)) (ADDRESS=(PROTOCOL=TCP)(HOST=Server2-Vip)(PORT=1521))) (CONNECT_DATA=(SERVICE_NAME=ORCL.MY.DOMAIN))) The **jdbc.url** above is broken into several lines for readability, while it should appear as a single line in the actual configuration file. When Load Balancing is set to "on", failover between listeners is active by default.

Related topics

Installation Wizard other database option

Installation Wizard 'Other database' option

Installation Wizard other database option

This topic contains information about the **Other database** option in the OO installation wizard. This option enables using specific JDBC driver and connection options. Use this option if:

- You want to use a different version of the JDBC driver, other than the ones provided in the OO installation (see the note below).
- You want to provide a JDBC connection URL yourself, to include an option or options which are not currently provided by the standard database connection options.

NOTES:

- Connection is restricted to the types and versions of database described in the "*Overview*" section in "*Prepare the database environment*" section.
- Use a non-provided JDBC driver at your own risk. The following JDBC drivers are supported:
  - jtds-1.3.0.jar – for SQL Server 2008R2 and 2012
  - sqljdbc4-4.2.jar – for SQL Server 2014
  - ojdbc7-12.1.0.2.jar
  - postgresql-9.4.1207.jar
  - mysql-connector-java-5.1.35 (not provided in the OO installation)

When you install OO using the **Other database** option, the installation wizard does not create the database and/or related user or roles. You must create these beforehand, using the installation wizard. Detailed instructions on how to create the database/user/role can be found in this document, under the “Manually Creating an OO Database” section in each database topic.

Microsoft SQL server named instance example

SQL Server 2008R2 and 2012

The following example show a connection to a Microsoft SQL Server Named Instance using jtds JDBC Connector. The database, login role, and user are pre-created by the DBA, and the login role is the owner of the database (has full DML and DDL privileges).

Use the following details in the installation wizard, and revise the highlighted values to match your environment.

Use JDBC URL Option #1 when a unique TCP port is assigned to each instance.

Use JDBC URL Option #2 when **Database Browser :service** is active and is able to route connections based on the instance name. Note that TCP port number does not appear in this option.

Field	Value	Comments
JDBC Driver jar	C:\my\path\jtds-1.3.0.jar	
JDBC Driver class name	net.sourceforge.jtds.jdbc.Driver	
JDBC URL Option #1	jdbc:jtds:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT>/<DB_NAME>;instance=<INSTANCE_NAME>;sendStringParametersAsUnicode=true	
JDBC URL Option #2	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<INSTANCE_PORT>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	
Username	<LOGIN_ROLE>	
Password	<LOGIN_ROLE_PASSWORD>	

SQL Server 2014 or later

The following example show a connection to a Microsoft SQL Server Named Instance using Microsoft JDBC Connector. The database, login role, and user are pre-created by the DBA, and the login role is the owner of the database (has full DML and DDL privileges).

Use the following details in the installation wizard, and revise the highlighted values to match your environment.

Use JDBC URL Option #1 when a unique TCP port is assigned to each instance.

Use JDBC URL Option #2 when Database Browser service is active and is able to route connections based on the instance name. Note that TCP port number does not appear in this option.

Field	Value	Comments
JDBC Driver jar	C:\my\path\sqljdbc4-4.2.jar	
JDBC Driver class name	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL Option #1	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<INSTANCE_PORT>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	INSTANCE_PORT represents the separate instance
JDBC URL Option #2	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>;instanceName=<INSTANCE_NAME>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	If Database Browser service is active
Username	<LOGIN_ROLE>	
Password	<LOGIN_ROLE_PASSWORD>	

Microsoft SQL server Windows authentication example

SQL Server 2008R2 and 2012

The following example shows a connection to a Microsoft SQL Server using Windows Authentication and jtds JDBC connector. The database is pre-created by the DBA, and the Windows login account is the owner of the database (has full DML and DDL privileges).

Use the following details in the installation wizard, and revise the highlighted values to match your environment:

Field	Value	Comments
JDBC Driver jar	C:\my\path\jtds-1.3.0.jar	Use jtds JDBC driver only
JDBC Driver class name	net.sourceforge.jtds.jdbc.Driver	
JDBC URL	jdbc:jtds:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT>/<DB_NAME>;domain=<DOMAIN_NAME>	
Username	;<WINDOWS_USERNAME>nicode=true	
Password	< WINDOWS_USERNAME_PASSWORD>	

Note Ignore the error regarding incompatible collation, but make sure to set your collation to one of the supported collations described in the “*Language support*” topic in "*Prepare the database environment*" section.

It is recommended but not required, that OO Central service(s) is run as the same user that is used for database authentication. Note that using a domain account for authentication may enforce security policies that may change from time to time by the domain administrator.

Whenever the domain account password changes, the new password must be encrypted and the new encrypted password must be placed in the **database.properties** file located under **<OO installation>/central/conf**. If an OO cluster is used, this operation must be repeated for all OO Central servers.

SQL Server 2014

The following example show connection to a Microsoft SQL Server 2014 using Windows Authentication and Microsoft JDBC Connector.

Prerequisites:

1. The database is pre-created by the DBA
2. The relevant Windows account is the owner of the database (has full DML and DDL privileges)
3. OO’s installation wizard must be unzipped and extracted to a folder form

4. **sqljdbc_auth.dll** must be downloaded and copied to OO's installation wizard under <installer_folder>\java\bin. This dll file is not provided along with the OO installation and is available for download from Microsoft web site. Download Microsoft JDBC 4.2 and extract 64bit version of **sqljdbc_auth.dll**.

Installation steps:

1. Run the OO installer by executing **<installer_folder>\installer.bat** as the same Windows account that owns the OOdatabase.
2. In the **Connectivity** page, select the **Do not start central server after installation...** checkbox.
3. In the **Database Connection** page select the **Other database** option and fill in the details according to the table below. Edit and replace highlighted parts with actual values.
4. After the installation:
   1. Verify **sqljdbc_auth.dll** is located in **<central installation>\java\bin**.
   2. Configure the service to run under the same Windows account that owns the OO database.
   3. Start the OO service and verify successful connection to the database.

Repeat the steps above for all central server installations

Use the following details in the installation wizard, and revise the highlighted values to match your environment:

Field	Value	
JDBC Driver jar	C:\my\path\sqljdbc4-4.2.jar	**Comments**
JDBC Driver class name	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT	
	>;databaseName=<DB_NAME>;integratedSecurity=true;sendStringParametersAsUnicode=true	
Username	Fully qualified Windows account: **DOMAIN\USERNAME**	
Password	***DOMAIN\USER PASSWORD***	

Note Ignore the error about incompatible collation, but make sure to set your collation to one of the supported collations described in the "*Language support*" topic in "*Prepare the database environment*" section.

Using a domain account for authentication may enforce security policies that may change from time to time by the domain administrator.

Oracle RAC example

The following example shows a connection to an Oracle RAC cluster using explicit JDBC URL. The database (user) must be pre-created by the DBA, and must be granted DML and DDL privileges.
Use the following details in the installation wizard, and revise the highlighted values to match your environment:

Field	Value	Comments
JDBC Driver jar	C:\my\path\ojdbc7-12.1.0.2.jar	
JDBC Driver class name	oracle.jdbc.OracleDriver	
JDBC URL	jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=Server1_VIP)(PORT=1521))(ADDRESS=(PROTOCOL=TCP)(HOST=Server2_VIP)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=RAC1.MY.DOMAIN))	
Username	<PRE-CREATED-DATABASE>	
Password	< PRE-CREATED-DATABASE-PASSWORD>	

Related topics

Prepare the database environment

Set up database environment for OO Workflow Designer

This section shows the database administrator how to configure four different types of database, to be used with OO Workflow Designer.

- Prepare the database environment
- Deploy and maintain a Microsoft SQL server database
- Deploy and maintain an Oracle database
- Deploy and maintain a MySQL database
- Deploy and maintain a Postgres database
- Installation wizard 'Other database' option

Prepare the database environment

This chapter is intended for trained database administrators. If you are not familiar with the type of database you wish to use, or you feel that you lack the relevant knowledge required in order to create and configure an OO Workflow Designer database, see the database vendor's documentation and make sure you fully understand each action you perform as described in this chapter. This chapter also contains information about the types of databases used with OO Workflow Designer and describes the required database settings. If the settings are not specified you can leave the default values or continue with those configured by your organization's DBA.

Overview

The term "database" may be interpreted in several ways, depending on the database vendor/technology used. In Oracle, the term "database" relates to a collection of files containing data and metadata. A single Oracle database may contain one or more schemas (and users). A Microsoft SQL Server "database" is closer in definition to Oracle's "schema" than to Oracle's "database". In order to avoid confusion, this topic will use the following terms:

- Instance/server – the software and memory structures providing RDBMS services
- Database – the entity containing tables, views, indexes, and so on OO Workflow Designer requires a single database to be created. This database may co-exist with other databases contained in a database server. You can set up an OO Workflow Designer database on one of the following database server types:
- Microsoft SQL Server Standard/Enterprise (2008 R2/2012/2014/2016)
- Oracle 11gR2 Standard/Enterprise Server, including RAC environment
- Oracle 12cR1 Standard/Enterprise Server - regular instance (non-CDB) including RAC environment and Oracle 12cR2 Standard/Enterprise Server - regular instance (CDB).
- Postgres (9.3/9.4/9.5/9.6)
- MySQL Community/Standard/Enterprise Server (5.5/5.6/5.7)

Language Support

OO Workflow Designer can be installed and used in any supported language (English, French, German, Japanese, Spanish, and Simplified Chinese). Databases and database servers should be properly configured in order to support the desired language.

OO Workflow Designer database connectivity relies on Java JDBC. If your environment requires specific adaptations or security measures, see the JDBC documentation (or database vendor

documentation) to find out exactly how the JDBC connection URL should be formatted.

Related topics

- Deploy and maintain a Microsoft SQL server database
- Deploy and maintain an Oracle database
- Deploy and maintain a MySQL database
- Deploy and maintain a Postgres database

Use database clusters

Database clusters help to make the OO Workflow Designer system more robust, by protecting the database from several types of failures. OO Workflow Designer does not provide database cluster-related features of its own, such as database connection failover. It relies on the functionality of the JDBC package being used, and the database cluster environment capabilities, such as SCAN/AG listeners. You can install OO Workflow Designer in conjunction with any type of database cluster environment that satisfies the following conditions:

- Connection pooling is supported
- A single, valid database URL can be provided
- It provides reliable transaction handling during failover (a single, complete transaction must either completely fail or succeed)

The OO Workflow Designer installer does not provide database cluster-specific installation options. During installation, you need to provide the installer a simplistic / "regular" form of database connection. Database URL adaptations usually have to be made after the system is already installed in the **database.properties** configuration file.

Related topics

- Additional guidelines for Microsoft SQL server
- Additional guidelines for Oracle

Database security

The OO Workflow Designer database is at the heart of the OO Workflow Designer system, containing the OO Workflow Designer system configuration and possibly corporate-sensitive data. It is highly recommended that you enforce strict security settings on your database by following these guidelines. Harden your database server by following the recommendations from the database vendor and the operating system vendor. Server hardening may include, among other items:

- Restrict SSH access on Linux servers to a well-controlled set of OS users
- Enforce a strict OS user password policy – password length, complexity, lockout-policy, and so on
- Set up an audit system to detect and report break-in attempts

Harden your database user/login accounts:

- Enforce the password policy – password length, complexity, lockout-policy, and so on
- Restrict access to administration-level accounts
- Set up an audit system to detect and report break-in attempts

Database encryption at file-system level is supported as long as it does not compromise OO Workflow Designer’s performance, and is completely transparent to OO Workflow Designer. For example, Oracle TDE may be used as long as it does not cause any performance degradation or prevent LOB free-space reclamation.

Database sizing

OO Workflow Designer keeps all content and configuration data in the database. This topic will help you prepare for installing OO Workflow Designer. By estimating your system scale (small/standard/enterprise), you will be able to derive the amount of disk space required, derive the amount of memory (RAM) used by the database, and determine additional database installation parameters. Estimate the system scale:

System Criteria\Scale	Small	Standard	Enterprise
Number of content packs*	1 - 9	10 - 49	50 or more
Number of users**	1 - 9	10 - 49	50 or more

`* Number of content packs refers to the number of distinct items in the OOST_CONTENT_ITEMS table (and satellite tables).`

`** Number of users means the number of workspaces, projects, and so on. This is assuming that every user has 10 workspaces.`

The following table provides disk space and memory requirements for different OO Workflow Designer deployment scales:

System Scale\Parameter	Database Disk Size Requirements	Memory
Small	10 GB	4 GB
Standard	50 GB	8 GB
Enterprise Between	50 and 100 GB	12 GB

Notes

- Disk space and memory values are estimates. Actual disk space and memory consumption vary, depending on the database vendor and database server configuration.
- Memory (RAM) reflects recommended database memory, not the overall amount of memory available on the database machine (virtual or physical server).
- Disk space reflects the amount of disk space required for day-to-day operation of the OO Workflow Designer system and reasonable historical data retention—not including long-term database backups.
- The amount of additional disk space required for keeping OO Workflow Designer database backups depends on the backup policy (frequency and retention period).

Hardware requirements

The following table describes the hardware (CPU and memory) requirements recommended for the each of the database servers.

Note

The memory values reflect database memory consumption as part of the entire machine's memory.

Database\Scale		Small/Standard				Enterprise			
		CPUs		RAM		CPUs		RAM	
		Small	Standard	Min	Rec	Min	Rec	Min	Rec
SQL Server	2	4	4 GB	8 GB	4	8 GB	4 GB	8 GB	
Oracle	2	4	4 GB	8 GB	4	8 GB	4 GB	8 GB	
MySQL	2	4	4 GB	8 GB	4	8 GB	4 GB	8 GB	
Postgres	2	4	4 GB	8 GB	4	8 GB	4 GB	8 GB	

Min = Minimal value, Rec = Recommended value In addition to the generalized hardware requirements above, refer to the appropriate hardware requirements and software requirements sections per database.

Deploy and maintain a Microsoft SQL server database

In order to deploy OO Workflow Designer using Microsoft SQL Server, you must have an existing SQL Server database service. If you need to create a new database service, see the relevant documentation provided by Microsoft, because this information is not included within this guide. However, this guide contains recommendations for the SQL Server configuration. It is also recommended to use the SQL Server Agent service in order to comfortably schedule index maintenance jobs.

Workflow for Microsoft SQL Server Deployment

To deploy OO Workflow Designer using Microsoft SQL Server, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure Microsoft SQL Server**.
4. **Create an OO Workflow Designer database on Microsoft SQL Server**.
5. (Optional) **Set up Windows authentication**. For details, see "Using Windows Authentication to Access Microsoft SQL Server Databases" in "*Additional guidelines for Microsoft SQL server*" topic. This step is only relevant if you are using Windows authentication instead of SQL Server authentication.

Related topics

- Prepare the database environment
- System requirements for Microsoft SQL server
- Configure SQL server
- Manually Create an OO Workflow Designer database on Microsoft SQL s8.NGerver
- Additional guidelines for Microsoft SQL server

System Requirements for Microsoft SQL Server

This section describes the system requirements for working with Microsoft SQL Server in conjunction with OO Workflow Designer.

Hardware requirements

For OO Designer database sizing guidelines and hardware requirements, see "OO Workflow Designer *Database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" section. For Microsoft SQL Server hardware requirements, see the relevant installation guide for your Microsoft SQL server release and operating system.

Software requirements

The following table lists the Microsoft SQL Server releases supported by OO Workflow Designer:

Microsoft SQL Server Database Releases			
Version	**Type**	**32/64Bit**	**Service Pack**
2016	Standard	64 Bit	1
	Enterprise	64 Bit	1
2014	Standard	64 Bit	1
	Enterprise	64 Bit	1
2012	Standard	64 Bit	2
	Enterprise	64 Bit	2
2008 R2	Standard	x64	3
		x86	3
	Enterprise	x64	3
		x86	3

See the Microsoft documentation for supported platforms.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by Micro Focus quality assurance personnel.

Database Release			Operating System
Version	**32/64Bit**	**Service Pack**	
Microsoft SQL Server 2016 Enterprise Edition	64 Bit	1	Windows 2016 Standard Edition 64 Bit
Microsoft SQL Server 2014 Enterprise Edition	64 Bit	1	Windows 2012 Standard Edition 64 Bit

Microsoft SQL Server 2012 Enterprise Edition	64 Bit	2	Windows 2012 Standard Edition 64 Bit
Microsoft SQL Server 2008 R2 Enterprise Edition	64 Bit	3	Windows 2012 Standard Edition 64 Bit

Language support

In Microsoft SQL Server, unlike other databases, OO Workflow Designer database does not use Unicode collation. Use one of the following collations depending on your OO Workflow Designer installation language:

Language	Database Collation
English	SQL_Latin1_General_CP1_CS_AS
Japanese	Japanese_Unicode_CS_AS
Simplified Chinese	Chinese_Simplified_Stroke_Order_100_CS_AS

NOTE:
When an MS SQL database is created with a case-sensitive collation, object names such as tables, keys, and so on become case-sensitive as well.

For example, for the MY_TABLE_STEP_LOG_BINDINGS table, a command such as `SELECT * FROM my_table_step_log_bindings` is seen as using an invalid object name.

Related topics

- Prepare the database environment

Configure SQL server

This topic contains information on Microsoft SQL Server and database configuration settings. You can install an OO Workflow Designer database in any SQL Server environment including clustered environments. Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Microsoft SQL Server 2008R2, 2012, 2014, and 2016	
Server Options/Features	
Configuration Item	**Supported Configuration Options**
Server Configuration Options	Defaults, unless instructed otherwise
Instances	Default, Single
Authentication Mode	Mixed, Windows 1
Full-Text Search	(Not required for OO Workflow Designer)

[1] At the moment, the OO Workflow Designer installer supports only SQL authentication. Windows authentication can be configured afterward.

Microsoft SQL Server 2008R2, 2012, 2014, and 2016			
Instance/Server Options			
	Mandatory	**Recommended**	**Supported**
Server Collation		**SQL_Latin1_General_CP1_CS_AS**	Any Collation
Network Libraries	Server: **TCP/IP** Client: **TCP/IP**		
Concurrent Connections	**>= 800**	**0** *(unlimited)*	
Max Server Memory	**> 4GB**	**2,147,483,647** *(default, unlimited) Allocate 4-12 GB depending on system scale*	

Microsoft SQL Server 2008R2, 2012, 2014, and 2016			
Database Options			
	Mandatory	**Recommended**	**Supported**
Collation	**Any collation listed in the *"Language support"* in *System requirements for Microsoft SQL server* topic.**		
Recovery Model		**Full**	Simple, Full
Allow Snapshot Isolation	True		

Is Read Committed Snapshot On	True		
Auto Shrink	False		
Auto Create Statistics	True		

NOTE:
SQL Server does not use READ-COMMITTED transaction isolation by default. It is mandatory to set the **Allow Snapshot Isolation** and **Is Read Committed Snapshot On** flags. OO Workflow Designer fails to operate properly using any other type of transaction isolation.

Related topics

- Additional guidelines for Microsoft SQL server
- System requirements for Microsoft SQL server

Manually create an OO database on Microsoft SQL server

This topic describes the procedure for manually creating an OO Workflow Designer database, login, and user on Microsoft SQL Server. During OO Workflow Designer setup, a new database can be created automatically by the OO Workflow Designer installer or an existing database can be used. If during installation, you are authorized to connect to the database server as **sysadmin** (that is, connect as “sa”), use the installer's “**Create the database/schema**” option, and you can skip this section.

NOTE:
Only the database, login and user are created at this point; objects such as tables and indexes are not created yet. These objects are created once OO Workflow Designer starts for the first time.

This topic is relevant for you if, for example, due to security restrictions, you do not wish to use login/user credentials with elevated privileges during the OO Workflow Designer installation. In such a case, you (or your organization’s DBA) should create the database, login, and user first, and then let the OO Workflow Designer installer connect to the pre-existing database using “lower” privileges. To create a database, you must connect to the SQL Server instance using a login that has **CREATE DATABASE** permission.

NOTE:

- Members of the sysadmin server roles automatically have **CREATE DATABASE** permission, and are also mapped to dbo in all databases.
- Perform the following procedures only if you are an experienced Microsoft SQL Server database administrator.
- If you prefer to use the database creation wizard/GUI, make sure you select all options that correspond with the T-SQL code presented below. For example, make sure you set **Allow Snapshot Isolation** to TRUE under the **Options** page/**Other Options** pane/**Miscellaneous** tab in the New Database dialog box.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to Microsoft SQL Server as “sa” or any other login with a **sysadmin** role or the **CREATE DATABASE** permission.
2. Run <installation folder>/designer/bin/sql/MS-SQL/mssql_create_ws_db.sql and verify that the database, login, and user were created successfully. Adapt the script parameters to match your environment.
3. (Optional) In order to verify that database objects can be created by the new login and user, connect to the database server using the newly created login role credentials and run **<installation folder>/designer/bin/sql/MS-SQL/mssql_create_test_table.sql**.

Verify that the script ran correctly and that no errors are shown.

Manually create database objects

Once the database, login, and user are in place, the database objects (tables, indexes, and so on) are created when the OO Designer service starts and connects to the database for the first time. You could create the database objects manually if the user created for OO Workflow Designer does not have permissions to create or modify database objects (limited to DML operation only). It is highly recommended to grant the OO Workflow Designer database user DDL-related privileges, in order to allow OO Workflow Designer to perform upgrade operations without manual intervention, as these operations sometimes require modifications to the database structure. To manually create the database objects:

1. Extract the **mssql.sql** files from the OO Workflow Designer installation zip file under **docs\sql**.
2. Log in to Microsoft SQL Server as the relevant user, who is permitted to create and modify database objects in the OO Workflow Designer database.
3. Run the **mssql.sql** file and verify that no errors occurred.

Microsoft SQL server database maintenance

This topic describes the various maintenance tasks that are recommended for OO Workflow Designer databases created on Microsoft SQL server, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Backing up the OO Designer database

Microsoft SQL Server databases are either configured for the **Full** recovery model, or the **Simple** recovery model. You can back up an OO Workflow Designer database using either one of these models. As OO Workflow Designer keeps all configuration and data in a single database, always backup the complete database.

Consider the following guidelines when you create your backup plan for OO Workflow Designer:

Backup method:

The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements.

If your organization is more tolerant to data loss, you can use the simple recovery model and perform a daily or weekly full backup.

Backup frequency:

Daily backup is recommended, especially if you are using/modifyingOO Workflow Designer on a daily basis.

You should back up once a month at the very least.

Timing:

Schedule backup for the time when OO Workflow Designer is least active.

Retention:

Retention depends on your business guidelines and regulations.

Create a maintenance plan

Maintaining an OO Workflow Designer database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO Workflow Designer database in good shape.

Utilities for OO Workflow Designer Database Housekeeping

NOTE:

SQL scripts are also available in the OO Workflow Designer installation under the **designer/bin/sql** folder.

If you are copying SQL scripts from this document, note that you may need to remove redundant line breaks from the copied version.

It is recommended to perform index maintenance on a regular basis.

Note that index rebuilding online (without OO Workflow Designer system downtime) requires an enterprise grade database. Make sure you are running an enterprise version of Microsoft SQL Server before attempting online index rebuilding.

Also, note that maintenance activity usually consumes additional resources from the database. This is why it is important to schedule maintenance activity at the times when OO Workflow Designer is least active.

Upgrade/Rollback Instructions

Prior to any upgrade or rollback of OO Workflow Designer, take the following steps:

1. Purge non-essential data from the database.Upgrade and rollback operations convert flow execution data. Less data means a shorter upgrade/rollback. If all flow execution data is essential, perform a complete database backup first and then purge.
2. Back up your database, preferably after the database was purged.
3. Schedule the operation to the database server's quiet period.
4. Verify that no shared resources are exhausted – temporary space, transaction log, and similar resources.

Always on support

Microsoft SQL Server 2008R2 and 2012 "**Always On**" is supported by OO Workflow Designer as a legacy client[1] to implement a high availability and/or disaster recovery solution.

[1] OO Workflow Designer does not support the "MultiSubnetFailover" feature for SQL Server 2008R2 and 2012, and requires that **RegisterAllProvidersIP** is set to 0 in the cluster/listener configuration.

Connection to SQL Server 2014 and 2016 "Always On" is supported including the "`MultiSubnetFailover`" feature, but have not been certified for OO Workflow Designer.

High availability setup

As OO Workflow Designer requires a single point of contact, it is essential that an Availability Group Listener (AG listener) exist in the high availability setup. OO Workflow Designer's database connection is defined by a single URL - containing a single hostname.

Normally, OO Workflow Designer interfaces with the AG listener in order to connect to the availability group primary replica (a READ-WRITE capable instance).

In the case of a database failover scenario, while database connectivity is lost, OO Workflow Designer will repeatedly attempt to connect to the database until a new READ-WRITE capable instance accepts the connection. The redirection of the configured database hostname to a different IP address should be done by the AG Listener and cluster environment.

NOTE:

OO Workflow Designer has no use for a read-only database (thus, has no use for read-only routing).

Disaster Recovery Solution

In a disaster recovery setup, OO Workflow Designer should be halted, reconfigured (i.e, edit the

database.properties file), and then started again once the new database instance is available.

Related topics

- Additional guidelines for Microsoft SQL server

Additional guidelines for Microsoft SQL server

This appendix contains additional guidelines relevant for OO Workflow Designer deployment on Microsoft SQL Server.

Use Windows authentication to access Microsoft SQL server databases

Unless configured otherwise, OO Workflow Designer uses Microsoft SQL Server authentication to access Microsoft SQL Server databases. Note that the OO Workflow Designer installer currently does not support using Windows authentication during OO Designer installation. However, Windows authentication can be used once OO Workflow Designer is installed.

This appendix describes how to enable OO Workflow Designer to use Windows authentication to access Microsoft SQL Server databases.

Configure OO Workflow Designer to work with Windows authentication

You can enable OO Workflow Designer to use Windows authentication instead of Microsoft SQL Server authentication to access OO Workflow Designer databases.

To enable OO Workflow Designer to use Windows authentication to access a Microsoft SQL database:

1. Encrypt the Windows user password using the **encrypt-password.bat** utility located under **<installation>/designer/bin** by running:encrypt-password.bat --encrypt --password <password>Save the generated string in order to use it in the next step.
2. Back up your current **database.properties** file located under **<installation>/designer/conf** if you have an existing (usable) database connection.
3. Edit the **database.properties** file located under **<installation>/designer/conf**, and change only the relevant parameter syntax to match the following example.

When copying scripts, note that you may need to remove redundant line breaks from the copied version.

```
jdbc.url=jdbc\:jtds\:sqlserver\://[DB_HOSTNAME]\:[PORT]/[DB_NAME]
;sendStringParametersAsUnicode\=true
;domain\=[DOMAIN NAME]

db.username=[USERNAME]

db.password=[the string generated by encrypt-password.bat]

jdbc.url=jdbc\:sqlserver\://[DB_HOSTNAME]\:[PORT];databaseName\=[DB_NA
ME];sendStringParametersAsUnicode\=true;integratedSecurity\=true;
```

There might be a need to copy sqljdbc_auth.dll to the system path of the OO Workflow Designer server machine. For more details, see the Microsoft JDBC documentation.

Replace all the highlighted items with the correct values that match your environment.

Note that the **jdbc.url** parameter is broken in this document into several lines for readability. When copying this example onto your **database.properties** file, remove all line breaks so the **jdbc.url** parameter is composed of a single line containing no white spaces.

Configure OO Workflow Designer to work with Always On

OO Workflow Designer connects to SQL Server databases using the Microsoft JDBC 4.2 connector.

Here is an example on how to format JDBC URL in order to connect to an Always On cluster:

```
db.url=jdbc:sqlserver://[AG-NET-NAME];instanceName=[NAMED-INST-NAME];
databaseName=[DB-
NAME];multiSubnetFailover=true;applicationIntent=ReadWrite;
sendStringParametersAsUnicode=true
```

The **multiSubnetFailover** and **applicationIntent** are added in order to allow more complex behaviors.

The Multi-Subnet Failover scenarios in conjunction with SQL Server 2014 Always On cluster were not certified with OO Workflow Designer.

For additional information and connection options, refer to "JDBC Driver Support for High Availability, Disaster Recovery" in the Microsoft documentation and *Setting the Connection Properties* in the Microsoft JDBC documentation.

Deploy and maintain an Oracle database

In order to deploy OO Workflow Designer using Oracle, you must have an existing Oracle database service. If you need to create a new database instance/service, see the relevant documentation provided by Oracle, because this information is not included within this guide. However, this guide contains recommendations for the Oracle instance configuration.

Workflow for Oracle deployment

To deploy OO Workflow Designer using Oracle, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure an Oracle Database**.
4. **Create a Database**.
5. (Optional) **Connect OO Workflow Designer to an Oracle RAC environment**. This step is only relevant if you are using OO Workflow Designer in an Oracle RAC environment.

Related topics

- Prepare the database environment
- System requirements for Oracle
- Configure an Oracle database
- Manually Create an OO Workflow Designer Database on an Oracle instance
- Additional guidelines for Oracle

System requirements for Oracle

This topic describes the system requirements for working with Oracle in conjunction with OO Workflow Designer.

Hardware requirements

For OO Workflow Designer database sizing guidelines and hardware requirements, see "OO Workflow Designer *database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" section.

NOTE: For Oracle hardware requirements, see the relevant installation guide for your Oracle release and operating system.

Software requirements

The following table lists the Oracle releases supported by OO Workflow Designer:

Oracle Releases			
Version	**Type**	**32/64Bit**	**Patch Set**
11g R2	Standard	64 Bit	11.2.0.1 – 11.2.0.4 [1]
	Enterprise	64 Bit	11.2.0.1 – 11.2.0.4 [1]
12c R1 Regular instance, non-CDB	Standard	64 Bit	12.1.0.1 - 12.1.0.2
	Enterprise	64 Bit	12.1.0.1 - 12.1.0.2

The listed patch sets are the minimal patch sets supported. Newer patch sets are also supported unless stated otherwise in the OO Workflow Designer *Release Notes*.

NOTE: See the Oracle documentation for supported platforms.

NOTE: Oracle 12c R1 is only supported in its regular, backward-compatible instance form. OO Designer does not support connecting to an Oracle 12c container database (CDB). The Oracle 12c RAC environment is supported in backward-compatible form. New features such as multiple cluster subnets for SCAN listener are not supported.

Oracle connector

The jdbc connector `.jar` fie is not provided with the OO Workflow Designer installation. You must download it and place it under each `Designer <installation folder>/designer/lib` folder. Verify that the connector version you are using is fully compatible with your database server version. The recommended driver is the Oracle JDBC driver version 7-12.1.0.2

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by quality assurance personnel.

Database Release			Operating System
Version	32/64Bit	Patch Set	
Oracle 11g R2 Enterprise Edition	64 Bit	11.2.0.4.6	Red Hat Enterprise Linux 6.5 64 Bit
Oracle 11g R2 Enterprise Edition	64 Bit	11.2.0.4.0	Windows 2012 Standard Edition 64 Bit
Oracle 12c R1 Enterprise Edition	64 Bit	12.1.0.2	Red Hat Enterprise Linux 6.5 64 Bit

Language support

The Oracle instance character set should be set to **AL32UTF8**. This enables using any Unicode character (and practically all common characters in all languages).

Related topics

- Prepare the database environment

Configure an Oracle database

This topic contains information on the Oracle instance and database configuration settings. You can install an OO Workflow Designer database in an Oracle clustered environment (Oracle RAC or other). Legend:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Oracle Database 11gR2 / 12cR1			
Instance/Server Options			
Instance configuration options	**Defaults, unless instructed otherwise**		
	Mandatory	**Recommended**	**Supported**
PROCESSES	**>= 500**		
SESSIONS	**>= 555**		
TIMED_ STATISTICS		**TRUE**	TRUE, FALSE
OPEN_CURSORS	**>= 900**		
Shared/Dedicated Server		**Dedicated**	Dedicated, Shared
UNDO_ MANAGEMENT		**AUTO**	Automatic, Manual
Undo size	**>= 4GB**	**6GB – 10GB**	
Memory Management		**ASMM**	AMM, ASMM
MEMORY_TARGET		**0 (disabled)**	>= 5G *(for AMM)*
SGA_TARGET		**8G – 12G**	>= 4G *(for ASMM)*
SGA_MAX_SIZE		**8G – 12G**	>= 4G *(for ASMM)*
PGA_AGGREGATE_ TARGET		**1G – 2G**	>= 500M *(for ASMM)*

- **Note**: All values reflect resources required by OO Workflow Designer. If OO Workflow Designer shares the Oracle instance with other users, these values should be added to whatever is currently consumed by others.

Oracle Database 11gR2 / 12cR1			
Instance/Server Options			
	Mandatory	**Recommended**	**Supported**
File system			ASM, Any

Storage options		**Locally managed tablespace**	
		Automatic segment space management (ASSM)	
		Automatic local extent management	
ARCHIVELOG mode		**ARCHIVELOG**	ARCHIVELOG, NOARCHIVELOG
Redo Log total size	**>= 600MB**	**1GB**	

- **Note**: All values reflect resources required by OO Workflow Designer. If OO Workflow Designer shares the Oracle instance with other users, these values should be added to whatever is currently consumed by others.

Manually create an Operations Orchestration database on an Oracle instance

This topic describes the procedure for manually creating an OO Workflow Designer database in an Oracle instance. During OO Workflow Designer setup, a new database can be created automatically by the OO Workflow Designer installer or a pre-existing database can be used. If during installation, you are authorized to connect to the database server as dba (connect as “SYSTEM”), use the “**create the database/schema**” option, and you can skip this section.

NOTE: In some cases, the term "database" is used but in the case of Oracle, it should be interpreted as "user".

NOTE: Only the database is created at this point; objects such as tables and indexes are not created yet. These objects are created once OO Workflow Designer starts for the first time.

This topic is relevant for you if, for example, due to security restrictions, you do not wish to use user credentials with elevated privileges during theOO Workflow Designer installation. In such a case, you (or your organization’s DBA) should create the user (database) first, and then let the OO Workflow Designer installer connect to the pre-existing database using basic privileges. To create a database, you must connect to the Oracle instance using a login that has **CREATE USER** system privilege—for example, system user.

NOTE:

- Any user with the DBA role has sufficient privileges to create the new user.
- Perform the following procedures only if you are an experienced Oracle database administrator.
- If you prefer to use the database creation wizard/GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to the oracle as “system” or any other user with a DBA role.
2. Edit the **<installation folder>/designer/bin/sql/Oracle/oracle_create_ws_db.sql** SQL script, run it, and verify that the database was created successfully.
3. (Optional) In order to verify that database objects can be created by the new user, connect to the Oracle instance as OO Workflow Designer user, edit and run: **<installation folder>/designer/bin/sql/Oracle/oracle_create_test_table.sql**

Verify that the script ran successfully, and no errors or warnings were shown.

Manually create database objects

Once the database is in place, the database objects (tables, indexes, and so on) are created when the OO Workflow Designer service starts and connects to the database for the first time. You can create the database objects manually if the user created for OO Workflow Designer does not have

permissions to create or modify database objects (limited to DML operation only). It is highly recommended to grant the OO Workflow Designer database user DDL-related privileges, in order to allow OO Workflow Designer to perform upgrade operations without manual intervention, as these operations sometimes require modifications to database structure. To manually create the database objects:

1. Extract the **oracle.sql** file from the OO Workflow Designer installation zip file under **docs\sql**.
2. Edit the **oracle.sql** file and prefix each object with the OO Workflow Designer user, to make sure the objects are created under the OO Workflow Designer user.
3. Connect to the OO Workflow Designer database as the relevant user, who is permitted to create and modify database objects in the OO Workflow Designer database.
4. Run the **oracle.sql** file and verify that no errors occurred, and that all objects are created and owned by the OO Workflow Designer user.

Connect to Oracle using SID or service name

Connecting to an Oracle database server requires you to specify either the SID (system ID) or Service Name. OO Workflow Designer supports specifying the service name during installation (using either the Installation Wizard or a silent installation). The following examples show how to format JDBC URL for each option, as it should appear in the **database.properties** file under the **designer\conf** folder. Connecting to an Oracle instance using the SID:

```
jdbc.url=jdbc\:oracle\:thin\:@DB_HOSTNAME_OR_IP\:PORT\:SID
```

Connecting to an Oracle instance using the Service Name:

```
jdbc.url=jdbc\:oracle\:thin\:@//DB_HOSTNAME_OR_IP\:PORT/SERVICE_NAME
```

Oracle database maintenance

This topic describes the various maintenance tasks that are recommended for OO Workflow Designer databases created on Oracle, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO Workflow Designer database

Oracle databases can be backed up using several tools, such as **expdp** and **RMAN**. An OO Workflow Designer database can be backed up using any type of method/tool as long as the complete database is backed up. Consider the following guidelines when you create your backup plan for OO Workflow Designer: **Backup method:** The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements. **Backup frequency:** Daily backup is recommended, especially if you are using/modifying OO Workflow Designer on a daily basis. You should back up once a month at the very least. **Timing:** Schedule backup for the time when OO Workflow Designer is least active. **Retention:** Retention depends on your business guidelines and regulations.

Create a maintenance plan

Maintaining an OO Workflow Designer database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO Workflow Designer database in good shape. **Utilities for OO Workflow Designer Database Housekeeping**

NOTE:

SQL scripts are also available in the OO Workflow Designer installation under the **designer/bin/sql** folder.

If you are copying SQL scripts from this document, note that you may need to remove redundant line breaks from the copied version.

It is recommended to perform index maintenance on a regular basis. **Note**: The index rebuilding online (withoutOO Workflow Designer system downtime) requires an enterprise grade database. Make sure you are running an enterprise version of Oracle before attempting online index rebuilding. You must also consider that maintenance activity usually consumes additional resources from the database. Therefore it is important to schedule maintenance activity at the times when OO Workflow Designer is least active.

NOTE: ONLINE index rebuilding should only be performed when the enterprise edition is used. Otherwise, the index rebuilding operation may lock tables and indexes and may interfere with the operation of OO Workflow Designer.

Additional guidelines for Oracle

This appendix contains the configuration that needs to be done for OO Workflow Designer to work with Oracle 11gR2 and 12cR1 Real Application Cluster (RAC) environments. This information is for advanced users only.

Oracle RAC

A cluster is a collection of interconnected servers that appear as one server to the end user and to applications. Oracle Real Application Cluster (RAC) is Oracle's solution for high availability, scalability, and fault tolerance. It is based on clustered servers that share the same storage. Oracle RAC is a single Oracle database installed on a cluster of hardware servers. Each server runs an instance of the database and all the instances share the same database files. For more details about Oracle RAC, see the Oracle Clusterware Guide and the Oracle Real Application Clusters Administration and Deployment Guide in the Oracle documentation set of your release. In this appendix, the following Oracle RAC example is used:

- Oracle RAC Cluster name: OORAC
- Service Name: ORCL.MY.DOMAIN
- Machine names: Server1, Server2
- On each machine, there is an Oracle instance of OORAC:
  - SID on Server1: OORAC1
  - SID on Server2: OORAC2
- On each machine, there is a virtual IP (Server1-Vip and Server2-Vip):
  - Server1-Vip is assigned to Server1
  - Server2-Vip is assigned to Server2The virtual IP is in addition to the static IP assigned to the machine.
- SCAN listener uses a virtual IP, usually published using DNS/GNS:SCAN-Vip
- The local listeners on both servers listen on the default port 1521 and support the database service OORAC. SCAN listener(s) reside on one of the cluster nodes, and may failover along with their virtual IP addresses in the case of a failure.

NOTE:
Although 12c SCAN allows configuring multiple SCAN listeners (per subnet), OO Workflow Designer only supports connecting to a single 12c SCAN listener.

Single client access name

In release 11g, Oracle introduced the Single Client Access Name (SCAN), as a preferred access method for clients connecting to the RAC. In this method, clients are not required to configure individual nodes in the RAC; rather, they use a single virtual IP known as the SCAN or the SCAN VIP. The SCAN is a single network name defined for the cluster either in your organization's Domain Name Server (DNS) or in the Grid Naming Service (GNS) that rotates between several IP addresses,

reflecting multiple listeners in the cluster. The SCAN eliminates the need to change clients when nodes are added to or removed from the cluster. The SCAN and its associated IP addresses provide a stable name for clients to use for connections, independent of the nodes that make up the cluster. 11g SCAN addresses, virtual IP addresses, and public IP addresses must all be on the same subnet. 12c SCAN allows configuring multiple SCAN listeners – one per subnet. OO Workflow Designer only supports connecting to a single 12c SCAN listener. The SCAN method is recommended when using OO Workflow Designer in an Oracle 11g RAC environment.

Configure OO Workflow Designer to work with Oracle RAC

Connecting to a SCAN Listener Virtual IP

1. In the OO Designer installation wizard **Database Connection** page, enter the SCAN listener virtual IP address/network name in the **Hostname or IP address** field.
2. Select the **Service Name** radio button instead of the **SID** radio button and enter the Oracle RAC service name.
3. Enter the rest of the details and click the **Test Connection** button.

Providing the SCAN listener virtual IP address guarantees that the OO Workflow Designer server will be able to reconnect to the database in the case of a failure. This connection method is the preferred one as it enables the OO Workflow Designer installation wizard to create the database as well as populate it. For silent installation use (replace the highlighted text with the actual values):

```
db.url=jdbc:oracle:thin:@//[SCAN-Vip]:[PORT]/[ORCL.MY.DOMAIN]
```

Load-Balancing Using Explicit Connection String Although using Oracle's SCAN listener virtual IP is the preferred method, you can also provide an explicit connection string by using the **Other database** installation option. For complete details, see "*Oracle RAC example*" in "*Installation wizard* '*Other database*' *option*" topic. Here is an example of how a load balancing connection string would appear in the OO Workflow Designer **database.properties** file.

1. Back up your current **database.properties** file located under **< installation>/designer/conf** if you have an existing (usable) database connection.
2. Edit the **database.properties** file as follows, while replacing the highlighted items with the values that match your environment.

NOTE:
When copying scripts, note that you may need to remove redundant line breaks from the copied version.

```
jdbc.url=jdbc\:oracle\:thin\:@
(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=TCP)(HOST=Server1-Vip)(PORT=1521))
(ADDRESS=(PROTOCOL=TCP)(HOST=Server2-Vip)(PORT=1521)))
```

```
(CONNECT_DATA=(SERVICE_NAME=ORCL.MY.DOMAIN)))
```

Note that the **jdbc.url** above is broken into several lines for readability, while it should appear as a single line in the actual configuration file. When Load Balancing is set to “on”, failover between listeners is active by default.

Related topics

- Installation wizard 'Other database' option

Deploy and maintain a MySQL database

In order to deploy OO Workflow Designer using MySQL, you must have an existing MySQL database. If you need to create a new database service, see the relevant documentation provided by MySQL, because this information is not included within this guide. However, this guide contains recommendations for the MySQL configuration.

Workflow for MySQL deployment

To deploy OO Workflow Designer using MySQL, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure MySQL** .
4. **Create an OO Workflow Designer database on MySQL** .

Related topics

- Prepare the database environment
- System requirements for MySQL
- Configure MySQL
- Manually create an OO Designer database on MySQL

System requirements for MySQL

This topic describes the system requirements for working with MySQL in conjunction with OO Workflow Designer.

Hardware requirements

For OO Workflow Designer database sizing guidelines and hardware requirements, see "OO Workflow Designer *database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" section.

NOTE: For MySQL hardware requirements, see the relevant installation guide for your MySQL release and operating system.

Software requirements

The following table lists the MySQL releases supported by OO Workflow Designer:

MySQL Database Releases		
Version	**Type**	**32/64Bit**
5.5	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit
5.6	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit
5.7	Community	x86 32-bit
		x86 64-bit
	Standard	x86 32-bit
		x86 64-bit
	Enterprise	x86 32-bit
		x86 64-bit

NOTE:
For MySQL 5.6.20 and 5.6.21, the requirements for the **innodb_log_file_size** have increased significantly.

For MySQL 5.6.1 - 19, the recommendation is 256M, but for MySQL 5.6.20 - 21, the recommendation is 2GB.
NOTE: See the MySQL documentation for supported platforms.

MySQL connector

The MySQL connector jar is not provided with the OO Workflow Designer installation. Download and place it under each OO Workflow Designer **<installation folder>/designer/lib** folder.
Verify that the connector version you are using is fully compatible with your database server version. Currently, the only supported version is 5.1.35.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by Micro Focus quality assurance personnel.

Database Release			Operating System
Version	32/64Bit	Patch	
MySQL Server 5.6.19 Community Edition	64 Bit		Windows 2012 Standard Edition 64 Bit
MySQL Server 5.6.12 Community Edition	64 Bit		Red Hat Enterprise Linux 6.3 64 Bit

Language support

The MySQL Server character set should be set to utf8. This lets you use any Unicode character (and practically all common characters in all languages).
NOTE: The OO Designer database uses the utf8_bin collation.

Related topics

- Prepare the database environment

Configure MySQL

This topic contains information on MySQL and OO Workflow Designer database configuration settings.
Legend details:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

MySQL 5.5, 5.6 and 5.7			
Instance/Server Options			
Server Configuration Options	Defaults, unless instructed otherwise		
[mysqld]	Mandatory	Recommended	Supported
character-set-server	**utf8**		
collation-server	**utf8_bin**		
transaction-isolation	**READ-COMMITTED**		
max-allowed-packet	**250M**		
max-connections	**>= 1000**		
default-storage-engine	**INNODB**		
innodb_log_file_size (for MySQL 5.6.1 - 19)	**256M**		
innodb_log_file_size (for MySQL 5.6.20 - 21)	**2GB**		
innodb_log_file_size (for MySQL 5.6.22 - 5.7)	**256M**		
explicit-defaults-for-timestamp	**0**		
sql-mode	**Default or none**		
innodb_file_per_table		1	
innodb_thread_concurrency		0	
table_open_cache		1000	
sort_buffer_size		2M	
read_buffer_size		2M	
tmp_table_size		400M	
max_heap_table_size		400M	
innodb_buffer_pool_size		4096M	
innodb_additional_mem_pool_size		20M	
binlog_format		row	
innodb_flush_log_at_trx_commit		2	
innodb_flush_method		O_DIRECT **(Linux Only)**	
innodb_doublewrite		0	

MySQL 5.5, 5.6 and 5.7			
Other Options			
Server Configuration Options	Defaults, unless instructed otherwise		
	Mandatory	Recommended	Supported
[client]			
default-character-set	**utf8**		
[mysql]			
default-character-set	**utf8**		
[mysqldump]			
max_allowed_packet	**250M**		

NOTE:

- MySQL options may be written using underscore or hyphen in different contexts. Use the correct form according to your server version and usage context.
- Using none-default sql-mode options may result in unexpected behavior.

Manually create an Operations Orchestration database on MySQL

This topic describes the procedure for manually creating an OO Workflow Designer database on MySQL.

During OO Workflow Designer setup, a new database can be created automatically by the OO Workflow Designer installer or a pre-existing database can be used.

If during installation, you are authorized to connect to the database server using the DBA role (connect as “root”), use the “**create the database/schema**” option, and you can skip this section.

NOTE:

Only the database and user are created at this point; objects such as tables and indexes are not created yet. These objects are created once OO Workflow Designer starts for the first time.

This topic is relevant for you if, for example, due to security restrictions, you do not wish to use login credentials with elevated privileges during the OO Workflow Designer installation. In such a case, you (or your organization’s DBA) should create the database first, and then let the OO Workflow Designer installer connect to the pre-existing database using basic privileges.

To create a database, you must connect to the SQL Server instance using a user that has **CREATE** permission (at the very least).

NOTE:

- **root** has all privileges. Any member of the DBA role will also be able to create the user and database.
- Perform the following procedures only if you are an experienced MySQL database administrator.
- If you prefer to use the MySQL Workbench GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to the MySQL as “root” or any other member of the DBA role.
2. Run **<installation folder>/designer/bin/sql/MySQL/mysql_create_ws_db.sql** and verify that the database and user were created successfully. Adapt the script parameters to match your environment.
3. (Optional) In order to verify that database objects can be created by the new user, connect to the database server using the newly created user and run **<installation folder>/designer/bin/sql/MySQL/mssql_create_test_table.sql**.

Verify that the script ran correctly and that no errors are shown.

NOTE:

SQL scripts are also available in the OO Workflow Designer installation under the **designer**/**bin**/**sql** folder.

If you are copying SQL scripts from this document, note that you may need to remove redundant line breaks from the copied version.

```
SET @OODB='OODB';
SET @OOUSER='OOUSER';
SET @OOPASS='OOPa55WorD';

SET @SQL1 = CONCAT('CREATE DATABASE IF NOT EXISTS `',@OODB,`` COLLATE
utf8_bin');
SET @SQL2 = CONCAT('CREATE USER ''',@OOUSER,'''@''%'' IDENTIFIED BY
''',@OOPASS,'''');
SET @SQL3 = CONCAT('GRANT ALL PRIVILEGES ON `',@OODB,'`.* TO
''',@OOUSER,'''');
PREPARE stmt1 FROM @SQL1;
PREPARE stmt2 FROM @SQL2;
PREPARE stmt3 FROM @SQL3;
EXECUTE stmt1;
EXECUTE stmt2;
EXECUTE stmt3;
DEALLOCATE PREPARE stmt1;
DEALLOCATE PREPARE stmt2;
DEALLOCATE PREPARE stmt3;
FLUSH PRIVILEGES;
```

Manually Create database objects

Once the database and user are in place, the database objects (tables, indexes, and so on) are created when the OO Workflow Designer service starts and connects to the database for the first time.

You can create the database objects manually if the user created for OO Workflow Designer does not have permissions to create or modify database objects (limited to DML operation only).

It is highly recommended to grant the OO Workflow Designer database user DDL-related privileges, in order to allow OO Workflow Designer to perform upgrade operations without manual intervention, as these operations sometimes require modifications to database structure.

To manually create the database objects:

1. Extract the **mysql.sql** file from the OO Workflow Designer installation zip file under **docs\sql**.
2. Connect to the OO Workflow Designer database as the relevant user, permitted to create and modify database objects in the OO Workflow Designer database.
3. Run the **mysql.sql** file and verify that no errors occurred.

MySQL database maintenance

This topic describes the various maintenance tasks that are recommended for OO Workflow Designer databases created on MySQL, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO Workflow Designer database

You can back up MySQL database using several tools, such as **mysqldump** or **mysqlbackup**. You can back up the OO Workflow Designer database using any type of method/tool as long as the complete database is backed up.

NOTE: Consider the following guidelines when you create your backup plan for OO Workflow Designer:

Backup method:

The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements.

Backup frequency:

Daily backup is recommended, especially if you are using/modifying OO Workflow Designer on a daily basis.

You should back up once a month at the very least.

Timing:

Schedule backup for the time when OO Workflow Designer is least active.

Retention:

Retention depends on your business guidelines and regulations.

Create a maintenance plan

Maintaining an OO Workflow Designer database includes rebuilding the index and reclaiming free space. Use the scripts and tools described in this section, in order to keep the OO Workflow Designer database in good shape.

Recommended Utility for Database Maintenance

In order to keep the OO Workflow Designer database in good shape, it is recommended to schedule **mysqlcheck** utility to run during a system maintenance window.

Important! Note that this operation locks tables! Only perform this action during a maintenance window when the OO Workflow Designer system is not operating!

Here is an example of how to run this utility:

```
mysqlcheck -uwsuser –p????? -os --auto-repair WS
```

Replace "wsuser" and "WS" with the actual OO Workflow Designer user name and database name,

respectively.
It is recommended not to provide the password explicitly. See the MySQL documentation for recommendations on how to secure database passwords.

Deploy and maintain a Postgres database

In order to deploy OO Workflow Designer using Postgres, you must have an existing Postgres database service. If you need to create a new database service, see the relevant documentation provided by Postgres, because this information is not included within this guide. However, this guide contains recommendations for the Postgres configuration.

Workflow for Postgres deployment

To deploy OO Workflow Designer using Postgres, perform the following steps:

1. **Review sizing guidelines**.
2. **Review Hardware and Software Requirements**.
3. **Configure Postgres**.
4. **Create OO Workflow Designerdatabase on Postgres**.

Related topics

- Prepare the database environment
- System requirements for Postgres
- Configure Postgres
- Manually create an OO Workflow Designer database on Postgres

System Requirements for Postgres

This topic describes the system requirements for working with Postgres in conjunction with OO Workflow Designer.

Hardware requirements

For OO Workflow Designer database sizing guidelines and hardware requirements, see "OO Workflow Designer *database sizing*" and "*Hardware requirements*" in "*Prepare the database environment*" section.

Note: For Postgres hardware requirements, see the relevant installation guide for your Postgres release and operating system.

Software requirements

The following table lists the Postgres releases supported by OO Workflow Designer:

Postgres Database Releases	
Version	**Type**
9.3	x86 32-bit
	x86 64-bit
9.4	x86 32-bit
	x86 64-bit
9.5	x86 32-bit
	x86 64-bit
9.6	x86 32-bit
	x86 64-bit

Only supported versions should be used.

Note: See the Postgres documentation for supported platforms.

Examples of tested deployments

The following table lists the deployment environments that have been rigorously tested by Micro Focus quality assurance personnel.

Database Release		Operating System
Version	**32/64Bit**	
Postgres 9.3.2	64 Bit	Red Hat Enterprise Linux 6.3 64 Bit
Postgres 9.4.6	64 Bit	Red Hat Enterprise Linux 7 64 Bit
Postgres 9.5.1	64 Bit	Red Hat Enterprise Linux 7 64 Bit
Postgres 9.6.1	64 Bit	Red Hat Enterprise Linux 7 64 Bit

Postgres 9.6.3	64 Bit	Windows 2012 R2

Language support

Postgres determines character set and collation at the database level. The OO Workflow Designer database uses Unicode (utf8) encoding and collation. This lets you use any Unicode character (and practically all common characters in all languages).

Related topics

- Prepare the database environment

Configuring Postgres

This topic contains information on OO Workflow Designer Postgres configuration settings. Legend details:

- **Mandatory** configuration options/values appear in **bold/orange** font.
- **Recommended** configuration options/values appear in **bold/purple** font.
- Supported configuration options/values appear in normal font, and may show as a comma-separated list.
- *Comments* appear in *italic* font.

Postgres 9.3 - 9.6			
Instance/Server Options			
Instance Configuration Options	Defaults, unless instructed otherwise		
	Mandatory	Recommended	Supported
max_connections	**>= 1000**		
default_transaction_isolation	**'read committed'**		
autovacuum	**on**		
track_counts	**on**		
shared_buffers	**>=512MB** [1]		
effective_cache_size	**>=2048MB**[1]		
work_mem	**>=1MB**[1]		
maintenance_work_mem	**>=32MB** [1]		
lc_messages		**'en_US.UTF-8'**	Any
lc_monetary		**'en_US.UTF-8'**	Any

[1] - Minimal values. See the Postgres documentation on how to tune these values in accordance with your environment.

Manually create an Operations Orchestration database on Postgres

This topic describes the procedure for manually creating an OO Workflow Designer database on Postgres.

During OO Workflow Designer setup, a new database can be created automatically by the OO Workflow Designerinstaller or a pre-existing database can be used.

If during installation, you are authorized to connect to the database server as a privileged user (connect as “postgres”), use the “**create the database/schema**” option, and you can skip this section.

NOTE:

Only the database and role are created at this point; objects such as tables and indexes are not created yet. These objects are created once OO Workflow Designer starts for the first time.

This topic is relevant for you if, for example, due to security restrictions, you do not wish to use login/user credentials with elevated privileges during the OO Designer installation. In such a case, you (or your organization’s DBA) should create the database, login, and user first, and then let the OO Workflow Designer installer connect to the pre-existing database using basic privileges.

To create a database, you must connect to the Postgres instance using a login that has **CREATEUSER** and **CREATEDB** privileges at the very least.

NOTE:

- The **postgres** built-in user has all the required privileges.
- Perform the following procedures only if you are an experienced Postgres database administrator.
- If you prefer to use the PgAdmin GUI, make sure you select all options that correspond with the SQL code presented below.
- Not all database creation options are specified—only those that differ from the default value. When in doubt, use default values.

To create a database:

1. Log in to Postgres as “postgres” or any other login role with **CREATEUSER** and **CREATEDB** privileges.
2. Run <**installation folder>/designer/bin/sql/PostgreSQL/postgres_create_ws_db_linux.sql** or **postgres_create_ws_db_windows.sql** depending on the database server’s operating system, and verify that the database and user were created successfully. Adapt the script parameters to match your environment.
3. (Optional) In order to verify that database objects can be created by the new user, connect to the database server using the newly created user and run <installation folder>/designer/bin/sql/PostgreSQL/postgres_create_test_table.sql. Verify that the script ran correctly and that no errors are shown. Note SQL scripts are also available in the OO Workflow Designer installation under the designer/bin/sql folder. If you are copying SQL scripts from this

document, note that y ou may need to remove redundant line breaks from the copied version.

Manually create database objects

Once the database and role are in place, the database objects (tables, indexes, and so on) are created when the OO Workflow Designer service starts and connects to the database for the first time.

To manually create the database objects (instead of the OO Workflow Designer service):

1. Extract the **postgres.sql** file from the OO Workflow Designer installation zip file under **docs\sql**.
2. Connect to the OO Workflow Designer database as the OO Workflow Designer database user.
3. Run the **postgres.sql** file and verify that no errors occurred.

Postgres database maintenance

This topic describes the various maintenance tasks that are recommended for OO Workflow Designer databases created on Postgres, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Back up the OO Workflow Designer database

You can back up a Postgres database using several tools, such as the **pg_dump** or **pg_backup** script. You can back up the OO Workflow Designer database using any type of method/tool as long as the complete database is backed up.

Consider the following guidelines when you create your backup plan for OO Workflow Designer:

Backup method:

The backup method depends mainly on business considerations—how much information ”may” be lost? What is the maximum time for system recovery? If you need to be able to perform point-in-time recovery, and only “allow” a few hours-worth of data loss, use the full recovery model and perform full or differential backups daily, and transaction log backup every N hours depending on your business requirements.

If your organization is more tolerant to data loss, you can use the simple recovery model and perform a daily or weekly full backup.

Backup frequency:

Daily backup is recommended, especially if you are using/modifying OO Workflow Designer on a daily basis.

You should back up once a month at the very least.

Timing:

Schedule backup for the time when OO Workflow Designer is least active.

Retention:

Retention depends on your business guidelines and regulations.

Create a maintenance plan

OO Workflow Designer Postgres database maintenance mainly includes table REINDEX, as **autovacuum** needs to be activated. Use the example below, in order to keep the OO Workflow Designer database in good shape.

Recommended Utility for Database Maintenance

In order to keep the OO Workflow Designer database in good shape, it is recommended to run the REINDEX action during a system maintenance window.

Important! Note that this operation locks tables! Only perform this action during a maintenance window when the OO Workflow Designer system is not operating!

Here is an example of how to REINDEX a complete database using the **reindexdb** utility:

```
reindexdb -d WS –U wsuser –W ?????
```

Replace “WS” and “wsuser” with actual OO Workflow Designer database and user names.
It is recommended not to provide the password explicitly. See the Postgres documentation for recommendations on how to secure database passwords.

Installation Wizard 'Other database' option

This appendix contains information about the **Other database** option in the OO Workflow Designer installation wizard.

This option enables using specific JDBC driver and connection options. Use this option if:

- You want to use a different version of the JDBC driver, other than the ones provided in the OO Workflow Designer installation (see the note below).
- You want to provide a JDBC connection URL yourself, to include an option or options which are not currently provided by the standard database connection options.

NOTES:

- Connection is restricted to the following types and versions of database:
  - Microsoft SQL Server Standard/Enterprise (2008 R2/2012/2014/2016)
  - Oracle 11gR2 Standard/Enterprise Server, including RAC environment
  - Oracle 12cR1 Standard/Enterprise Server - regular instance (non-CDB), including RAC environment
  - Postgres (9.1/9.2/9.3/9.4/9.5/9.6)
  - MySQL Community/Standard/Enterprise Server (5.5/5.6/5.7)
- Use a non-provided JDBC driver at your own risk. The following JDBC drivers are supported:
  - sqljdbc4-4.2.jar – for SQL Server 2014
  - ojdbc7-12.1.0.2.jar
  - postgresql-9.4.1207.jar
  - mysql-connector-java-5.1.35 (not provided in the OO Workflow Designer installation)

When you install OO Workflow Designer using the **Other database** option, the installation wizard does not create the database and/or related user or roles. You must create these beforehand. Detailed instructions on how to create the database/user/role can be found under the “Manually Creating an OO Workflow Designer Database” section in each database chapter.

Microsoft SQL server named instance example

The following example show a connection to a Microsoft SQL Server Named Instance using Microsoft JDBC Connector. The database, login role, and user are pre-created by the DBA, and the login role is the owner of the database (has full DML and DDL privileges).

Use the following details in the installation wizard, and revise the highlighted values to match your environment.

Use JDBC URL Option #1 when a unique TCP port is assigned to each instance.

Use JDBC URL Option #2 when Database Browser :service is active and is able to route connections based on the instance name. Note that TCP port number does not appear in this option.

Field	Value	Comments

JDBC Driver jar	C:\my\path\sqljdbc4-4.2.jar	
JDBC Driver class name	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL Option #1	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<INSTANCE_PORT>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	INSTANCE_PORT represents the separate instance
JDBC URL Option #2	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>;instanceName=<INSTANCE_NAME>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	If Database Browser service is active
Username	<LOGIN_ROLE>	
Password	<LOGIN_ROLE_PASSWORD>	

Microsoft SQL Server Windows Authentication Example

NOTE:
Ignore the error regarding incompatible collation, but make sure to set your collation to one of the supported collations, depending on your OO Designer installation language:

The following example show connection to a Microsoft SQL Server 2014 using Windows Authentication and Microsoft JDBC Connector.

Prerequisites:

1. The database is pre-created by the DBA.
2. The relevant Windows account is the owner of the database (has full DML and DDL privileges).
3. The OO Workflow Designer installation wizard must be unzipped and extracted to a folder form.
4. **sqljdbc_auth.dll** must be downloaded and copied to the OO Workflow Designer installation wizard under <installer_folder>\java\bin. This dll file is not provided along with the OO Workflow Designer installation and is available for download from Microsoft web site. Download Microsoft JDBC 4.2 and extract 64bit version of **sqljdbc_auth.dll**.

Installation steps:

1. Run the OO Workflow Designer installer by executing <**installer_folder>\installer.bat** as the same Windows account that owns the OO Workflow Designer database.
2. In the **Connectivity** page, select the **Do not start OO Workflow Designer server after installation...** checkbox.
3. In the **Database Connection** page select the **Other database** option and fill in the details according to the table below. Edit and replace highlighted parts with actual values.
4. After the installation:
   1. Verify **sqljdbc_auth.dll** is located in **<installation>\java\bin**.
   2. Configure the service to run under the same Windows account that owns the OO Workflow Designer database.
   3. Start the OO Workflow Designer service and verify successful connection to the database.

Repeat the steps above for all OO Workflow Designer server installations.

Use the following details in the installation wizard, and revise the highlighted values to match your environment:

Field	Value	Comments
JDBC Driver jar	C:\my\path\sqljdbc4-4.2.jar	
JDBC Driver class name	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT>;databaseName=<DB_NAME>;integratedSecurity=true;sendStringParametersAsUnicode=true	
Username	Fully qualified Windows account:**DOMAIN\USERNAME**	
Password	---- keep empty ----	

NOTE:

Ignore the error about incompatible collation, but make sure to set your collation to one of the supported collations, depending on your OO Workflow Designer installation language:

Language	Database Collation
English	SQL_Latin1_General_CP1_CS_AS
Japanese	Japanese_Unicode_CS_AS
Simplified Chinese	Chinese_Simplified_Stroke_Order_100_CS_AS

Note that using a domain account for authentication may enforce security policies that may change from time to time by the domain administrator.

Oracle RAC example

The following example shows a connection to an Oracle RAC cluster using explicit JDBC URL. The database (user) must be pre-created by the DBA, and must be granted DML and DDL privileges.

Use the following details in the installation wizard, and revise the highlighted values to match your environment:

Field	Value	Comments
JDBC Driver jar	C:\my\path\ojdbc7-12.1.0.2.jar	
JDBC Driver class name	oracle.jdbc.OracleDriver	
JDBC URL	jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=Server1_VIP)(PORT=1521))(ADDRESS=(PROTOCOL=TCP)(HOST=Server2_VIP)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=RAC1.MY.DOMAIN)))	
Username	<PRE-CREATED-DATABASE>	
Password	< PRE-CREATED-DATABASE-PASSWORD>	

Set up security and hardening

This section explains how to deploy and manage OO instances in a secure manner, and how to configure security hardening for OO.

Introduction
Security overview
Hardening Operations Orchestration

Introduction

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

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

Technical System Landscape

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 OO 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, 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 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 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.
- 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 OO 10.10 and 10.20 the following security updates were made:

- It is now possible to grant permissions for system accounts in 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.
- 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.
- When you upgrade an 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.
- 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.

- OO now 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).

Security overview

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

Security concepts
Secure implementation and deployment
Secure installation guidelines
Network and communication security
Administration interface security
User management and authentication
Authorization
Configure OO Central in IDM mode
Configure LDAP in IDM mode
Backup
Encryption

Security concepts

OO Glossary

Term	Description
Role Permission	A permission is a predefined authorization to perform a task. 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** - 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 systems that support LWSSO. For example, if users have logged onto another product web client that has LWSSO enabled, they can enter the OO Central application directly, bypassing the 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.

Related topics

Key concepts

Secure implementation and deployment

OO security hardening

The "Hardening" section provides recommendations for safeguarding your 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 OO system, it is necessary to secure both OO and the computing environment (for example, the infrastructure and the operating system) upon which the application runs.

The section also provides recommendations to help secure 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

It is recommended that OO is protected by physical security controls defined by your organization. The 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 section "*Support Matrix*" *Support Matrix* of *OO Install Guide*.

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

Installation permissions

The following permissions are required to install and run OO:

Installing OO	Windows/Linux: Any standard user who is able to run a Java process, and who has permission to create folders and services
Running OO	• Windows: The Windows service runs as the system user or a specific user (the user must have access to the 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.

Related topics

Support matrix

Network and communication security

The *OO Architecture Guide* describes the basic OO topology, high availability, and load balancer security.
The *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
OO supports the TLS protocol.
For more information, see "*Replace the Central TLS server certificate*" section.
The Central ports are defined by the administrator during the installation.
Channel Security
OO supports the following secure channels:

Channel (Directed)	Supported Secure Protocol
OOSH, browser, Studio Remote Debugger, or RAS à Central	For a secure channel, use the TLS communication for encryption and Client Certificate for authentication.
Central à 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 the “Setting Up Topology – Workers and RASes” in the *OO Central User Guide* ."*Set up Topology – Workers and RASes*" topic.

Related topics

Replace the Central TLS server certificate
Set up Topology – Workers and RASes

Administration interface security

Access the administration interface

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

- Credentials
- Client certificate
- SAML

Secure the administration interface recommendations

1. Authentication must be enabled in Central.
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.
3. It is recommended to work with LDAP users, rather than internal users, because this is more secure. LDAP users should be defined with a strong password policy.
4. It is recommended to set up authentication to access Central via Client Certificates. This is more secure than user passwords.

Related topics

Work with server and client certificates
Enable authentication

User management and authentication

Authentication model

To enable easy bootstrapping of the authentication mechanism in OO, the product starts with authentication disabled.
You must enable authentication immediately after installation.
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 (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.

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.

Database authentication

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.

Related topics

Enable authentication

Authorization

Authorization model

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

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.

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.

Related topics

Set up Security – Roles
Assigning permissions to a system account
Set permissions for content

Configure OO Central in IDM mode

When Central is connected to IDM, authentication is enabled by default at startup. To use OO Central in the IDM mode, you must switch from the LW-SSO security profile to the IDM security profile.

Switch to the IDM security profile

1. Open the **<installation dir>\central\conf\central-wrapper.conf** file in a text editor and locate the `wrapper.java.additional` parameter called `-Doverride.startup.mode` under the `# Java Additional Parameters` section.
2. Update the value of the `-Doverride.startup.mode` parameter to `IDM`. For example: wrapper.java.additional.33=Doverride.startup.mode=IDM To switch back to the LW-SSO security profile, update the value of the `-Doverride.startup.mode` parameter to `LWSSO`.
3. Save and close the **central-wrapper.conf** file.
4. Restart the Central service.

Update the IDM properties file

You can configure IDM properties in the **<installation dir>\central\conf\idm.properties** file. After configuring a property, save and close the file. Then restart the Central service.

The following table describes the properties in the **idm.properties** file.

Property	Description
idm.configuration.url	Represents the URL of the IDM service. For example, <idm_protocol>://<idm_hostname>:<idm_port>/<idm_service_path>
idm.configuration.username	User name of the REST IDM integration
idm.configuration.password	Password of the REST IDM integration. This value can be encrypted using the **<installation dir>\central\bin\encrypt-password.bat** tool.
idm.configuration.internal.username	User name of the internal account (used for PAS authentication and API calls towards the IDM service). This user has the IDM_ADMIN permission to be able to perform IDM administration operations for consumer organizations.
idm.configuration.internal.password	Password of the internal account (used for PAS authentication and API calls towards the IDM service). This value can be encrypted using the **<installation dir>\central\bin\encrypt-password.bat** tool.
idm.configuration.scheduler.username	User name of the scheduling account used to execute Central flow schedules.

idm.configuration.scheduler.password	Password of the scheduling account used to execute Central flow schedules. This value can be encrypted using the **<installation dir>\central\bin\encrypt-password.bat** tool.
idm.configuration.signing.key	Signing key of the IDM configuration
idm.configuration.oo.central.tenant	Tenant value of Central configured in the IDM service. The default value is `OO_Central`.
idm.configuration.oo.central.return.url	Return URL of Central after being redirected from the IDM login page. For example, <central_protocol>://<central_hostname>:<central_port>/oo

Configure the return URL of Central in IDM mode

When Central is running in IDM mode, users trying to visit Central are first redirected to the IDM login page to provide their credentials. This return URL is used to redirect the user back to Central's dashboard after a successful login. The value of the return URL can be configured using the `idm.configuration.oo.central.return.url` property in the idm.properties file.

Ensure that the protocol and ports used in the return URL are consistent throughout the entire login flow. For example, you cannot start the login flow through HTTP and provide an HTTPS return URL.

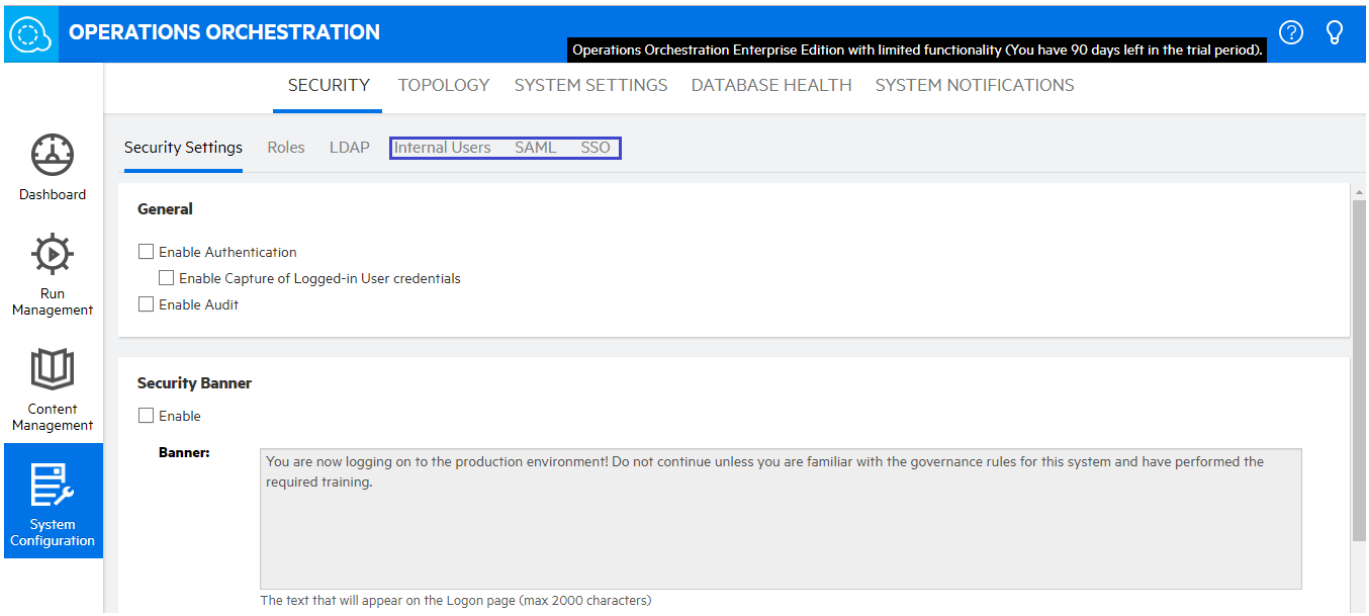
Example of the idm.properties file

```
# IDM Properties
idm.configuration.url=https://idm-hostname:8443/idm-service/
idm.configuration.username=idmTransportUser
idm.configuration.password={ENCRYPTED}uNEBrRr29t/78P700j3BiA==
idm.configuration.internal.username=admin
idm.configuration.internal.password={ENCRYPTED}30l15qHg/7770l15qHgP700d2MiA==
idm.configuration.signing.key={ENCRYPTED}uWRUrRQ4BzUNr2EG/4PDfB700j9LiA==
idm.configuration.oo.central.tenant=OO_Central
idm.configuration.oo.central.return.url=https://central-hostname:8433/oo
```

Hidden tabs in the IDM mode

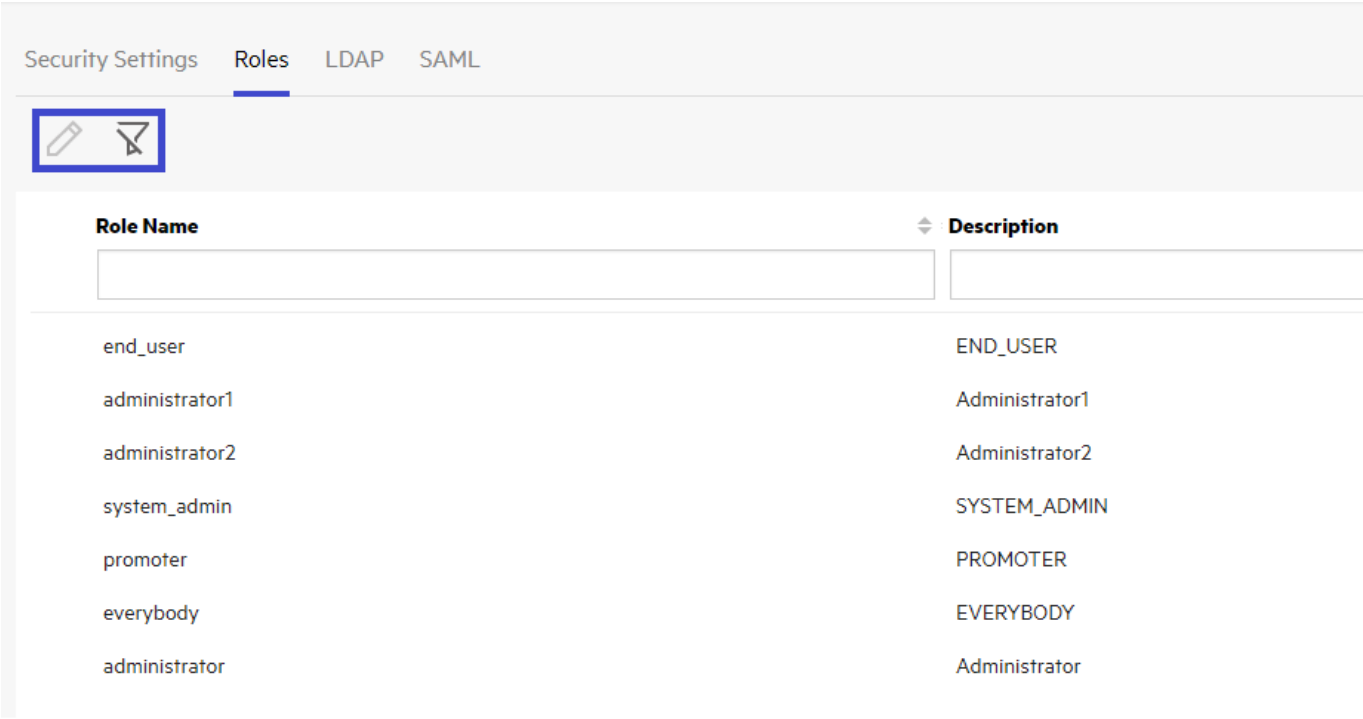
OO authentication allows the OO administrator to define internal users; however, in the IDM mode, the OO administrator cannot define internal users. Therefore, some tabs are hidden when OO is used in the IDM mode.

The following figure shows the tabs that are hidden in the IDM mode.



Roles in IDM

In the IDM mode, the OO administrator cannot create additional roles. The administrator can map only existing roles to groups. Therefore, the **Roles** tab in the UI displays only the **Edit** and **Clear Filters** icons.


Security Settings
Roles
LDAP
SAML
Role Name
Description
end_user
END_USER
administrator1
Administrator1
administrator2
Administrator2
system_admin
SYSTEM_ADMIN
promoter
PROMOTER
everybody
EVERYBODY
administrator
Administrator

Configure LDAP in IDM mode

When Central is configure to use IDM for authentication, you can add an LDAP configuration for IDM directly from Central.
The IDM server will use the LDAP configuration to authenticate and authorize user access to Central. Central supports a single LDAP configuration for an IDM server.

Add an LDAP configuration for the IDM server

1. Click the **System Configuration** button to display the **System Configuration** workspace.
2. Select **Security > LDAP**.
3. Click **Add**.
   The LDAP tab displays the fields relevant to the LDAP configuration.
4. In the **Base DN** field, enter the root DNs for fetching users.
5. In the **User ID** field, enter the distinguished name of a user with search privileges.
   It is recommended to provide a privileged user, because some OO functionality uses searching, for example, the scheduler. If there is no privileged user, this functionality may not work properly, depending on how the LDAP is configured.
   For example, `uid=john,ou=people,dc=devlab,dc=ad`
   Ensure that you use the exact DN for the user with these privileges.
6. In the **Password** field, enter the password of the user with search privileges.
7. In the **Connections** section, select **Secured Channel** to support secured channel transport.
8. Click **Add** in the **Connections** section to add a the connectivity details of the LDAP host.
9. In the **Host** field, enter the IP address or host name of the LDAP server.
10. In the **Port** field, enter the port number of the LDAP server. This value must be between 0 and 64435.
11. In the **User Name Attribute** field, enter the LDAP attribute that contains the user name.
12. In the **Common Name** field, enter the attribute that defines how the user name will be displayed on the screen. In most LDAP implementations, **cn** is the default.
13. In the **User Search Base** field, enter the root DN for fetching users.
14. In the **Users Search Filter** field, enter the search filter parameters in order to filter users. This filter will be applied on the user DN that was entered in the **User Search Base** field.
15. In the **User Email** field, enter the LDAP attribute that contains the user email.
16. Select the **Users SUB Search Scope** check box if you want the users filter to run recursively.
17. In the **Group Membership** field, enter the LDAP attribute that defines to which groups the user belongs.
18. In the **Manager Identifier** field, enter the LDAP attribute that identifies the group.
19. In the **Manager Identifier Value** field, enter the search value for the **Manager Identifier** field.
20. Click **Save** to save the configuration.

If the LDAP configuration is not saved, an asterisk appears next to the domain name in the LDAP navigation pane to the left. If you leave the page before saving, a message appears, warning that

there are unsaved changes.

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 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 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.

Related topics

Back up and recover Operations Orchestration
Set up disaster recovery

Encryption

Encryption model

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 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

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.

Encryption administration

Recommended Encryption Best Practices

In order to reach higher levels of security and cryptography, it is recommended to configure OO to be compliant with Federal Information Processing Standards (FIPS) 140-2. 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 OO for FIPS 140-2 compliance, OO uses the following security algorithm:

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

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).

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

Certificate Location

Installations of 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 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 OO Self-signed Certificate

It is recommended to replace the OO self-signed certificate after a new installation of 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.

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.

- OO out-of-the-box content packs contain a digital signature from Verisign.
- 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.

Sensitive information in a content pack page break before

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 OO security best practice is to configure the system account passwords in Central.

Related topics

Change or disable the HTTP/HTTPS ports
Configure compliance with FIPS 140-2
Replace the Central TLS server certificate
Deploy and manage content packs
Set up configuration items for a content pack

Hardening Operations Orchestration

This section describes how to configure security hardening for OO.

Hardening recommendation
Working with server and client certificates
Encrypting the communication using a server certificate
Client certificate manual authentication
Signing OO content packs
Verifying the integrity of the Studio tools JARs
Configure Operations Orchestration for FIPS 140-2 level 1 compliance
Configure the TLS protocol
Preventing flows from accessing the Central/RAS local file system
Adding Java security manager
Enable redirection to HTTPS
Disable CSRF protection

Note For information about other administrative tasks, see "*Administer*" section.

Related topics

Administer

Hardening recommendation

Security hardening recommendations

1. Install the latest version of OO.
2. (Optional) Configure OO for FIPS 140-2 Compliance. If you choose to do this, it must be configured before you start the Central server.
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.
4. Harden the OO Central server by removing the HTTP port and replacing the passwords of the KeyStore and TrustStore with strong passwords.
5. Harden OO Studio by replacing the KeyStore and TrustStore passwords with strong passwords, and encrypt or obfuscate the passwords in the configuration files.
6. It is not mandatory to harden Studio. While Central and RAS check the validity of the certificate by default, when Studio connects to a remote Central over HTTPS, it does not check the validity of the certificate. This makes it possible to use the Studio Debugger, even if you don’t have a CA-signed or self-signed certificate. If you want to harden your Studio environment, follow the steps in Import a CA root certificate to the Studio TrustStore.
7. Remove the RC4 cipher from the SSL-supported ciphers.
8. (Optional) Configure the TLS protocol version.
9. Enable authentication in Central. Internal users are not secured, so use a secured LDAP with a strong password policy.
10. Harden/secure the operating system and database.
11. 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.
12. In the Windows and SQL server environment, configure OO to work with Windows authentication.
13. Make sure that auditing is enabled in Central.

Related topics

Install
Configure OO for FIPS 140-2 level 1 compliance
Work with server and client certificates
Change or disable the HTTP/HTTPS ports
Change and encrypt/obfuscate the KeyStore/TrustStore password
Remove vulnerable ciphers from the SSL supported Ciphers
Configure the TLS protocol
Enable authentication

Set up security – LDAP authentication
Additional guidelines for Microsoft SQL server
Enable auditing

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.
OO uses the Keytool utility to manage cryptographic keys and trusted certificates. This utility is included in the 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#$filename|.
Note Keytool is an open source utility.
Installations of 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 OO certificate (private key).

Note If you use client certificates with LDAP, the LDAP must be configured as default in Central.
Recommendations:

- It is recommended to replace the OO self-signed certificate after a new installation of 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.

Related topics

Set up security – LDAP authentication

Encrypting the communication using a server certificate

This section describes the encriptions scenarios and has the following topics.

Replacing the Central TLS server certificate
Importing a certificate to the Central TrustStore
Importing a certificate to a RAS TrustStore
Importing a certificate to the OOSH TrustStore
Importing a certificate to the Studio TrustStore
Checking the revoke status of a certificate
Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password
Removing vulnerable Ciphers from the SSL supported Ciphers
Changing the HTTP/HTTPS ports or disabling the HTTP port
Troubleshooting the HTTPS connector

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\Micro Focus\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. It is also recommended to change the default “changeit” password in the automatically generated KeyStore in the Central server.
8. Start Central.

Related topics

Change and encrypt/obfuscate the KeyStore/TrustStore password

Import 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, 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.

Import 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, 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.

Import a CA root certificate to the OOSH TrustStore

If you are using a custom root certificate for Central, you will need to import the trusted root certificate authority (CA) to the OOSH **client.truststore**. If you are using a well known root CA (like Verisign) you do not have to perform the following procedure, because the certificate will already be in the **client.truststore** file.
By default, 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. Edit the **oosh.bat** from **<installation dir>/central/bin** or the main OOSH directory for a standalone OOSH.
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 the `-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 Note This property is set to **true** by default.
5. Import the trusted root certificate authority (CA) to the Central **client.truststore** file if it doesn't already exist in the CA list (by default, all the well known CAs are there): keytool -importcert -alias <any_alias> -keystore <path to the client.truststore> -file <certificate_name.cer> -storepass <changeit>
6. Run OOSH.
7. Start Central.

Import 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, 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.

Adding the Client Truststore Password

When you add a new property of **client.truststore.password** to the **studio.properties** file, you must do the following:

1. Using keytool, change the Studio client truststore password from the client truststore file at **<user>/.oo**: keytool -storepasswd -keystore <installation_folder>/central/var/security/client.truststore
2. Modify the **studio.properties** file with the new entry for client truststore password. client.truststore.password=={OBFUSCATED}obfuscated_passwordxxxxx Note After an upgrade to 10.5x, the new truststore location is the **<user>/.oo** folder.

Related topics

Debug a remote Central with Studio

Check 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.

Central has configuration in the Central wrapper xml files that contains the field checking the revocation status enable set to true.

The Central will then go to the CRL file and validate the entity’s certificate against this CRL file. In case of revocation there will be no connection between the Central and the Entity

In Central side there will be errors in the log file that the certificate was revoked and also the reason for the revocation.

The operation that Central is trying to perform fails.

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,TtrustStore, and server certificate passwords in the Central configuration

Make sure that Central is running. Note Before doing this step, make sure that there are encrypted passwords. 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.

b. Use the following c. 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

server.xml file located in **<installation** d. **dir>/central/tomcat/conf/server.xml**.

a. Locate the HTTPS connector. For example: keyPass="changeit" keystoreFile="C:/Program Files/Micro Focus/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/Micro Focus/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.

central-wrapper.conf file located in **<installation dir> central\conf\central** and e. 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>

f.

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.

Related topics

Encrypt and obfuscate passwords

Encrypt and obfuscate 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.

Encrypt 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>.

Obfuscate 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>.

Create a prompt for the password

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

```
C:\Program Files\Micro Focus\Operations Orchestration\central\bin>encrypt-password.bat
Password (typing will be hidden):
Confirm password (typing will be hidden):
{ENCRYPTED}gAkPCLQsYDhoR1Y2q9bjCQ==

C:\Program Files\Micro Focus\Operations Orchestration\central\bin>
```

This will create a prompt for the hidden password inputs.

Remove 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 OO:

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 OO 10.x, repeat these steps.

Change or disable the HTTP/HTTPS ports

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

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

By default, the connectors look as follows.

HTTP connector:

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

HTTPS connector:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false"
compression="on" keyAlias="tomcat" keyPass="changeit"
keystoreFile="C:/Program Files/Micro Focus/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/Micro Focus/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.

Change 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.

Disable 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.

Troubleshoot 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:\Micro Focus\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:/Micro Focus/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 manual 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.

Configure 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 "*Work with server and client certificates*" topic.

By default, the user certificate is matched against internal users of Operations Orchestration (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/Micro Focus/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/Micro Focus/Operations Orchestration/central/var/security/client.truststore" truststorePass="changeit" truststoreType="JKS"/>`
   - 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
   MTIwMTAwWjAiMCACASScXDTk5MDEwMTEyMDAwMFowDDAKBgNVHRUEAwoBAaAjMCEw
   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
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.

Note that even if 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.

Update 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.

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.

You can find the RAS ID under the **Topology** tab in Central in "*Set up topology – workers and RASes*" topic.

In 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.

Configure a client certificate in Studio remote debugger

Prerequisite: You must import the CA root certificate of Central into the 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:

- In 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.
- 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.
- Note that even if 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 OO internal domain.

Configure a client certificate in OOSH

Prerequisite: You must import the CA root certificate of Central into 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 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.

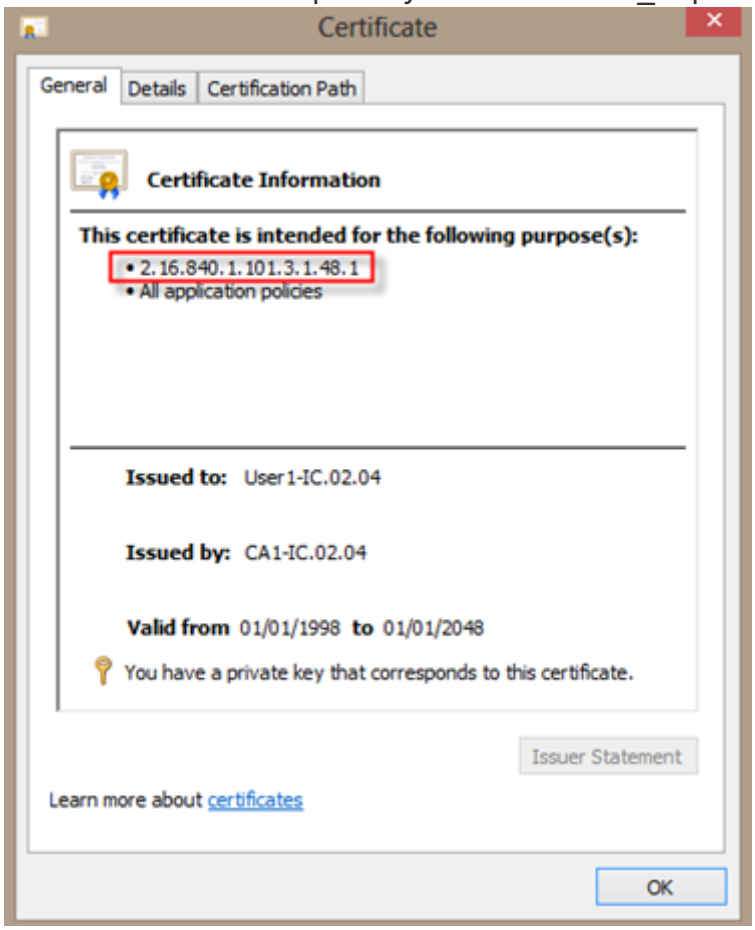
For the client certificate, you need to define a user, either an internal user or LDAP user. The name of the user should be defined in the certificate attributes. The default is value of the CN attribute.

Note that even if 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 OO internal domain.

Processing certificate policies

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

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



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 OO (LDAP or internal user).
- To change the regular expression, change the configuration item:

`x509.subject.principal.regex`.

Enable 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 _FTc_Basic_Variables._FT_Product_Acronymkeeps 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.

Related topics

Work with server and client certificates
Encrypt and obfuscate passwords
Import a CA root certificate to a RAS TrustStore
Set up topology – workers and RASes
Import a CA root certificate to the Studio TrustStore
Import a CA root certificate to the OOSH TrustStore

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**.

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#$filename|
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

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

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.

Verify 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.

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#$filename|
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
4. 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.

Configure Operations Orchestration for FIPS 140-2 level 1 compliance

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

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

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 OO to be compliant with FIPS 140-2, you can only revert back to the standard configuration if you re-install OO.

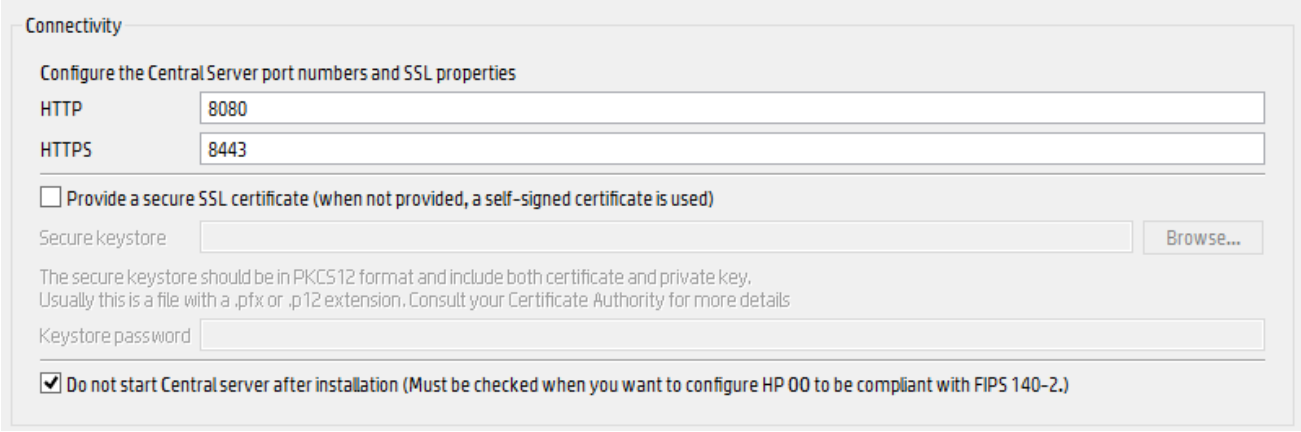
Prerequisites

Note for upgraders:
If you are upgrading from an installation of OO 10.10 (and later) that was already configured with FIPS, see *Steps for upgrade using FIPS* sub-topic in "*Upgrade to Operations Orchestration 10.8x from an earlier version of Operations Orchestration 10.x*" topic

Before configuring 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 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 OO that is in use (whether version 9.x or 10.x).
2. Verify that when 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#$filename|, and replace the **OpenJDK (Zulu) JRE** with the **Server Oracle JRE**.
   a. Delete everything inside the **<installation dir>\java** folder.
   b. Extract the downloaded archive.
   c. Copy the **JRE** folder content to **<installation dir>\java**.
5. Download and install the Java Cryptographic Extension (JCE) Unlimited Strength Jurisdiction Policy Files from the following site: http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154#$filename| Note See the **ReadMe.txt** file from the downloaded content for information on how to deploy the files and upgrade the JRE used by OO.
6. Install the RSA BSAFE Crypto software files. On the system on which OO is installed, copy the following to **<oo_jre>\lib\ext** (where **<oo_jre>** is the directory in which the JRE used by 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**

Note If you are upgrading from an installation of OO 10.10 (and later) that was already configured with FIPS, you must repeat steps 4 and 5 from the "*Prerequisites*" section above, and then repeat the steps in the "*Configure the properties in the Java Security file*" topic in "*Configure compliance with FIPS 140-2*" section.

Prerequisite steps for upgraders

1. Download the Server Oracle JRE 8 and replace the OpenJDK (Zulu) JRE with the Server Oracle JRE

from the following site:
http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154#$filename|
   1. Delete everything inside the **<upgrade dir>\JAVA folder**.
   2. Extract the downloaded archive.
   3. Copy the **JRE** folder content to **<upgrade dir>\JAVA**.
2. Download and install the Java Cryptographic Extension (JCE) Unlimited Strength Jurisdiction Policy Files from the following site:
   http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166#$filename|
   See the **ReadMe.txt** file from the downloaded content for information on how to deploy the files and upgrade the JRE used by OO.
3. Install the RSA BSAFE Crypto software files. On the system on which 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 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

Related topics

Configure compliance with FIPS 140-2

Configure compliance with FIPS 140-2

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

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 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 OO 10.10 and later that was already configured with FIPS, you must follow the "*Prerequisite steps for upgraders*" topic in "*Configure for FIPS 140-2 level 1 compliance*" section, 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 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\Micro Focus\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 OO encryption store file, so that it is FIPS-compliant.

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.

Note It is recommended to delete all the default CA root certificates that are not in use from the 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 "*Change the database password*" topic.

Step 5: Start OO

Related topics

Configure OO for FIPS 140-2 level 1 compliance
Replace the FIPS encryption
Change and encrypt/obfuscate the KeyStore/TrustStore password
Change the database password

Replace the FIPS encryption

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 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.

Change 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.

Change 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*" topic in "*Configure for FIPS 140-2 level 1 compliance*" section.

2. Complete all the steps in the "*Configure the properties in the Java Security file*" topic in "*Configure compliance with FIPS 140-2*" section.
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.
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.

Related topics

Configure OO for FIPS 140-2 level 1 compliance
Configure compliance with FIPS 140-2

Configure the TLS protocol

You can configure OO to define the supported TLS protocol version. By default,OOallows 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.

Prevent 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 Operations Orchestration 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 as described in "*Preventing flows from accessing the Central/RAS local file system*" topic.
The following sections describe how to set up the Java policy file for standalone RAS and Central embedded worker.

Configure 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.

Configure 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";
   }
   ```

Related topics

Prevent flows from accessing the Central/RAS local file system

Enable redirection to HTTPS

If you selected **Allow HTTP access** in the Installation wizard, and you want to change this to redirect the HTTP traffic to HTTPS, you can configure this manually in the **web.xml** file.

1. Stop the Central server.
2. Open the **<installation_folder>/central/tomcat/conf/web.xml** file.
3. Locate the `<security-constraint>` property.
4. Change the value of `<transport-guarantee>` to `CONFIDENTIAL`. For example: <security-constraint> <web-resource-collection> <web-resource-name>HTTPSOnly</web-resource-name> <url-pattern>/*</url-pattern> </web-resource-collection> <user-data-constraint> <transport-guarantee>CONFIDENTIAL</transport-guarantee> </user-data-constraint> </security-constraint>
5. Start the Central server.

Disable CSRF protection

A CSRF (Cross-Site Request Forgery) attack is when a malicious web site, email, blog, instant message, or program causes a user's web browser to perform an unwanted action on a trusted site for which the user is currently authenticated. The impact of a successful cross-site request forgery attack is limited to the capabilities exposed by the vulnerable application.

In a fresh installation of Micro Focus OO, or when you upgrade from an earlier version, CSRF protection is enabled by default.

You can enable or disable CSRF protection by setting the `csrf.protection.enabled` system configuration item to **true** or **false**. This can be done via REST API.

When using REST APIs, in some situations you will need to add the CSRF token header (just for POST/DELETE/PUT).

Note Make sure that the load balancer does not remove the XSRF header and cookie from the request/response.

You can disable CSRF protection by setting the `csrf.protection.enabled` system configuration item to the value **false**. This can be done via REST API.

To re-enable, set the value back to **true**.

To set this system configuration item, you must have **Manage System Settings** permissions.

You can enable or disable this protection in the **central-wrapper.conf** file.

1. Stop the Central server.
2. Open the **<installation_folder>/central/conf/central-wrapper.conf** file .
3. Locate the `Dcsrf.enabled` property and change it to `true`, to enable CSRF protection.
4. Start the Central server.

You can also enable or disable CSRF protection using the OO Shell (OOSH) tool. Use the following command:

```
oosh> set-sys-config –key csrf.protection.enabled –value true ……
```

Make sure to specify the options (`--key, --value, --url, --user, --password`) for this command.

For more information about CSRF protection, see https://www.owasp.org/index.php/Cross-Site_Request_Forgery_%28CSRF%29_Prevention_Cheat_Sheet#Disclosure_of_Token_in_URL.

Configure the database for localization

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

- English: `SQL_Latin1_General_CP1_CS_AS`
- Japanese: `Japanese_Unicode_CS_AS`
- Simplified Chinese: `Chinese_Simplified_Stroke_Order_100_CS_AS`
- German: `SQL_Latin1_General_CP1_CS_AS`
- French: `French_100_CS_AS`
- Spanish: `SQL_Latin1_General_CP1_CS_AS`

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

Setting the system locale in Central-wrapper.conf

If your Operations Orchestration system is localized, you will need to set the properties to reflect the system locale, in the **central-wrapper.conf** file.

1. Open the **central-wrapper.conf** file in a text editor (located under <**installation_folder**>**/central/conf**).
2. Edit the following properties:set.LANG= set.LC_ALL= set.LANGUAGE= wrapper.java.additional.<x>=-Duser.language= wrapper.java.additional.<x>=-Duser.country= For example, for Japanese: `set.LANG=ja_JP` `set.LC_ALL=ja_JP`
3. Save the **central-wrapper.conf** file.

Set up security and hardening for OO Workflow Designer

This chapter explains how to deploy and manage OO Workflow Designer instances in a secure manner, and how to configure security hardening for OO Workflow Designer.

- Security overview
- Hardening OO Workflow Designer

Security overview

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

Security concepts

- **User**

A user is an object associated with a person (or application identity) representing the person and defining their authorization.
OO Workflow Designer users are authenticated via LW-SSO (Lightweight Single Sign On). This is a mechanism in which a single action of user authentication and authorization can permit a user to access all Micro Focus systems that support LWSSO. For example, if users have logged onto another Micro Focus product web client that has LWSSO enabled, they can enter the OO Workflow Designer application directly, bypassing the OO Workflow Designer logon screen.

- **System Security**

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

- **Authentication**

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

- **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.

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** – OO Workflow Designer uses LWSSO for authentication.
- **TLS Encryption** – By default, OO Workflow Designer supports TLS protocol 1.2.
- **TLS Server Certificate** – By default, the user is asked to provide a CA certificate during the installation of the OO Workflow Designer server.
- **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.

OO Workflow Designer Security Hardening

The Hardening chapter provides recommendations for safeguarding your OO Workflow Designer 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 OO Workflow Designer system, it is necessary to secure both OO Workflow Designer and the computing environment (for example, the infrastructure and the operating system) upon which the application runs.

The Hardening section provides recommendations to help secure OO Workflow Designer at the application level and does not cover how to secure the infrastructure within the customer environment. The user is solely responsible for understanding the infrastructure or environment and applying the respective hardening policies.

Physical security

It is recommended that the OO Workflow Designer is protected by physical security controls defined by your organization. The OO Workflow Designer 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.

Related topics

- Change and encrypt/obfuscate the KeyStore/TrustStore password

Secure installation guidelines

Supported operating systems

For the types and versions of supported operating systems, see the "*System requirements*" section of OO Workflow Designer Install section.

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
- Recommended: Put up a firewall to protect against DDoS.

Tomcat hardening

When you install OO Workflow Designer, 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 OO Workflow Designer:

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

Note: See also the recommendations in the CIS Apache Tomcat documentation.

Security protection in OO Workflow Designer

The following security features have been designed to protect the security of your OO Workflow Designer installation.

CSRF protection

OO Workflow Designer is protected against CSRF (Cross-Site Request Forgery).

NOTE:
A CSRF attack is when a malicious web site, email, blog, instant message, or program causes a user's web browser to perform an unwanted action on a trusted site for which the user is currently authenticated. The impact of a successful cross-site request forgery attack is limited to the capabilities exposed by the vulnerable application.

For more information about CSRF, see https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF) .

XSS protection

OO Workflow Designer is protected against XSS (Cross-Site Scripting).

NOTE:
A XSS attack is when an attacker injects malicious code, generally in the form of a browser side script, to a different end user. An XSS vulnerability may be used by attackers to bypass access controls such as the same-origin policy.

For more information about XSS, see https://www.owasp.org/index.php/Cross-site_Scripting_(XSS).

Clickjacking protection

OO Workflow Designer is protected against clickjacking.

NOTE:
Clickjacking is when an attacker manipulates a website user's activity by concealing hyperlinks

beneath legitimate clickable content. Thus, the attacker is "hijacking" clicks meant for one page and routing them to another page.

For more information about clickjacking, see https://www.owasp.org/index.php/Clickjacking.

Network and communication security

The *OO Workflow Designer Use* section describes the basic OO Workflow Designer topology, high availability, and load balancer security.

Communication channel security

Supported Protocols and Configuration

OO Workflow Designer supports the TLS protocol.

The *OO Workflow Designer* ports are defined by the administrator during the installation.

Channel Security

OO Workflow Designer supports the following secure channel:

Channel (Directed)	Supported Secure Protocol
Browser à OO Workflow Designer	For a secure channel, use the TLS communication for encryption and Client Certificate for authentication.

Related topics

- Replace the OO Workflow Designer TLS server certificate

User management and authentication

Authentication model

There are two methods to authenticate access to OO Workflow Designer:

- Single sign on (LW-SSO) - supported at the backend

Micro Focus LW-SSO is used to authenticate user access to OO Workflow Designer at the backend and IDM service for the installer.

Database authentication

OO Workflow Designer 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 "*Additional guidelines for Microsoft SQL server* " topic.

Property	Description
idm.entry.point.return.url	Designer url, in the format: <http or https>://<HOSTNAME>:<PORT>/oo-designer
idm.provider.port	The port of the IDM service
idm.provider.hostname	The hostname of the IDM service
idm.provider.integration.acct.username	The IDM transport username which OO Designer will use in order to get the authentication token from the IDM service. Default value: idmTransportUser
idm.provider.integration.acct.password	The IDM transport username's password.
idm.tenant	The name of the tenant/organization that must exist in IDM. Default value: OO_Designer

idm.signing.key	The IDM configuration signing key. The signing key should be at least 32 characters. It must contain lower case, upper case and numeri characters.
idm.provider.protocol	The protocol of the IDM service. Default value: https

Related topics

- Additional guidelines for Microsoft SQL server

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 OO Workflow Designer, make sure to back up the **designer\var\security** folder and the **designer\conf\database.properties** file.
Some data on the database schema is encrypted and the keys for decryption are stored locally on the OO Workflow Designer 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.

Encryption

Encryption model

OO Workflow Designer 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 OO Workflow Designer.
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 are left in clear text. For example, the database passwords are encrypted.

- **Data in-transit**

OO Workflow Designer uses the Transport Layer Security (TLS) protocol to encrypt the data between components.

- **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.

Encryption administration

- **Recommended Encryption Best Practices**

In order to reach higher levels of security and cryptography, it is recommended to configure OO Workflow Designer to be compliant with Federal Information Processing Standards (FIPS) 140-2. OO Workflow Designer 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 OO Workflow Designer for FIPS 140-2 compliance, it uses the following security algorithm:
  - Symmetric-key algorithm: AES256
  - Hashing algorithm: SHA256

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 OO Workflow Designer server, you need to install a digital certificate on the server side.

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

Certificate Location

Installations of OO Workflow Designer include two files for the management of certificates using Keytool:

- **<installation dir>/designer/var/security/client.truststore**: Contains the list of trusted certificates
- **<installation dir>/desginer/var/security/key.store**: Contains the OO Workflow Designer 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 OO Workflow Designer service.

Replacing the OO Workflow Designer Self-signed Certificate

It is recommended to replace the OO Workflow Designer self-signed certificate after a new installation of OO Workflow Designer 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.

Related topics

- Change and disable the HTTP/HTTPS ports
- Configure compliance with FIPS 140-2
- Replace the OO Workflow Designer TLS server certificate

Log files

Logs let you trace errors, warnings, information, and debugging messages.
The logs are saved in the file server, in the following location:

- **<installation>/designer/var/logs**

Note: No sensitive data is kept in the log files in OO Workflow Designer.

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.

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

When you install OO Workflow Designer, you need to configure at least one available port for the OO Workflow Designer 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 OO Workflow Designer and the other components, see "*Network and communication security*" topic.

How do I configure self-signed SSL certificates for OO Workflow Designer?

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

How detailed are 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.

How is sensitive information encrypted?

Which authentication mechanism does OO Workflow Designer support?

OO Workflow Designer supports Client Certificate and LWSSO.

NOTE: Client certificates are not supported.

Is OO Workflow Designer FIPS 140-2 compliant?

Yes. For more information, see *Configure compliance with FIPS 140-2* topic.

Can I limit the OO Workflow Designer user IP address?

No, this is not supported at the moment.

Related topics

- Replace the OO Workflow Designer TLS server certificate
- Network and communication security
- Log files
- Encryption
- User management and authentication
- Configure compliance with FIPS 140-2

Hardening Operations Orchestration

This section describes how to configure security hardening for OO Workflow Designer.

Security hardening recommendations

1. Install OO Workflow Designer.
2. (Optional) Configure OO Workflow Designer for FIPS 140-2 Compliance.
3. Configure the OO Workflow Designer server certificate for TLS encryption and client certificate for strong authentication (mutual).

NOTE:
This can be done during installation.

1. Harden the OO Workflow Designer server by removing the HTTP port and replacing the passwords of the KeyStore and TrustStore with strong passwords.
2. Remove the RC4 cipher from the SSL-supported ciphers.
3. Harden/secure the operating system and database.
4. In the Windows and SQL server environment, configure OO Workflow Designer to work with Windows authentication.

Default security settings

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

- **Authentication** – OO Workflow Designer uses LWSSO for authentication.
- **TLS Encryption** – By default, OO Workflow Designer supports TLS protocol 1.2.
- **TLS Server Certificate** – By default, the user is asked to provide a CA certificate during the installation of the OO Workflow Designer server.
- **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.

Related topics

- Configure Operations Orchestration for FIPS 140-2 level 1 compliance
- Change and disable the HTTP/HTTPS ports
- Change and encrypt/obfuscate the KeyStore/TrustStore password
- Remove the RC4 cipher from the SSL-supported ciphers

- Additional guidelines for Microsoft SQL server
- Change and encrypt/obfuscate the KeyStore/TrustStore password

Working with server and client certificates

The 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.
OO Workflow Designer uses the Keytool utility to manage cryptographic keys and trusted certificates. This utility is included in the OO Workflow Designer 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 OO Workflow Designer include two files for the management of certificates:

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

Recommendations:

- It is recommended to replace the OO Workflow Designer self-signed certificate after a new installation of OO Workflow Designer 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 OO Workflow Designer service.
- It is recommended to clear the console after using Keytool or to use the prompt for password inputs.

Encrypt the communication using a server certificate

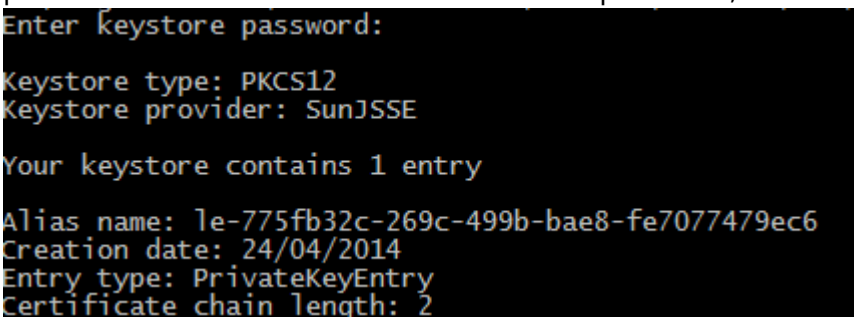
This section describes the encryption and obfuscation of the communication using a server certificate.

- Replace the OO Workflow Designer TLS server certificate
- Import a CA root certificate to the OO Workflow Designer TrustStore
- Change and encrypt/obfuscate the KeyStore/TrustStore password
- Encrypt passwords
- Remove the RC4 cipher from the SSL-supported ciphers

Replace the OO Workflow Designer 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 between brackets < > 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 OO Workflow Designer and back up the original **key.store** file, located in **<installation dir>/designer/var/security**.
2. Open a command line in **<installation dir>/designer/var/security**.
3. Delete the existing server certificate from the OO Workflow Designer **key.store** file, using the following command: keytool -delete -alias tomcat -keystore key.store -storepass changeit
4. If you already have a certificate with a **.pfx** or **.p12** extension, then 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.


6. Import the PKCS12 format server certificate to the OO Workflow Designer **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` NOTE: The alias should be tomcat unless it is also changed in the server.xml file.

7. If the imported server certificate has a different password from the original server certificate, it is important to change the keyPass password. It is also recommended to change the default “changeit” password in the automatically generated KeyStore in the OO Workflow Designerserver.
8. Start OO Workflow Designer. Related topics Replace the OO Workflow Designer TLS server certificate

Import a CA root certificate to the OO Workflow Designer TrustStore

If you are using a custom root certificate for OO Workflow Designer, 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, OO Workflow Designer 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 between brackets < >.

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

1. Stop OO Workflow Designer and back up the original **client.truststore** file, located in **<installation dir>/designer/var/security/client.truststore**.
2. Import the trusted root certificate authority (CA) to the OO Workflow Designer **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>
3. Start OO Workflow Designer.

Change and encrypt obfuscate the KeyStore TrustStore password

Change the KeyStore, TrustStore, and server certificate passwords in the OO Workflow Designer configuration

1. Make sure that OO Workflow Designer is running. NOTE: Before doing this step, make sure that there are encrypted passwords.
2. Stop the OO Workflow Designer 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>/designer/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>/designer/var/security/key.store Use the following keytool command to change the TrustStore password: keytool -storepasswd -keystore <installation_folder>/designer/var/security/client.truststore
4. Change the passwords also in the **server.xml** file located in **<installation dir>/designer/tomcat/conf/server.xml**.
   1. Locate the HTTPS connector. For example: keyPass="changeit" keystoreFile="C:/Program Files/Micro Focus/Operations Orchestration Designer/designer/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="TLSv1.2" truststoreFile="C:/Program Files/Micro Focus/Operations Orchestration Designer/designer/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.
   2. 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.

3. Save the file.

5. Edit the **designer-wrapper.conf** file located in **<installation dir> designer\conf** 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>
6. Start the OO Workflow Designer service. Related topics Encrypt passwords

Remove the RC4 cipher from the SSL-supported ciphers

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

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

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

1. Open the **<installation_dir>/java/bin/lib/security/java.security** file.
2. Disable the RC4 cipher by removing the comments and changing the parameters according to the example below: jdk.certpath.disabledAlgorithms=RC4, MD2, RSA keySize < 1024 jdk.tls.disabledAlgorithms=RC4, MD5, DSA, RSA keySize < 1024
3. Restart the OO Designer server.

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

Change or disable the HTTP/HTTPS ports

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

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

By default, the connectors look as follows.

HTTP connector:

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

HTTPS connector:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false"
compression="on" keyAlias="tomcat" keyPass="changeit"
keystoreFile="C:/Program Files/Micro Focus/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/Micro Focus/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.

Change 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.

Disable 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.

Configure the secured properties

When OO Workflow Designer was installed, decisions were made, which may affect security: the LW-SSO initialization string was set, a decision was made about whether to keep the HTTP port open, and a proxy may have been configured. If you need to change these decisions after installation, you can adjust the parameters in the **secured.properties** file.

For example, an administrator might decide to harden the configuration and close the HTTP port. In this case, you will need to set the secure flag for the CSRF cookie.

In the **secured.properties** file, you can configure the size limit of content packs that can be uploaded to OO Workflow Designer. The default size limit is 200 MB (the exact size limit is 209715200 bytes). However, you may want to adjust this to protect against DOS (Denial of Service). You can also configure the number of content packs that can be uploaded in parallel, and the number of content packs that can be created in parallel.

1. Open the **secured.properties** file, located in **<installation_dir>/designer/var/security/**.
2. If you need to close the HTTP port, make sure that the **is.secured.cookie** property is activated, as follows. is.secured.cookie=true Setting the **is.secured.cookie** property to **true** adds a secured attribute to the CSRF cookie. This the default when OO Workflow Designer is installed with no HTTP connection. However, if OO Workflow Designer was installed with an HTTP connection and you want to harden the installation later, you can change this property to **true**. NOTE: This should be set to true only when HTTP is disabled. Setting it to true when HTTP is enabled will prevent the users from working in HTTP. To close the HTTP port, you also need to complete the other steps described in "*Disable the HTTP port*" under "*Change and disable the HTTP/HTTPS ports*" topic.
3. To verify that the LW-SSO initialization string is encrypted, enter the encryption string in the **lwsso.initString** property. This string must be at least 12 characters in length and must be the same LW-SSO initialization string that is used by the embedding application.
4. To adjust the number of content packs that can be created in parallel across the system (for all users), update the value of the **max.parallel.cp.creation** property. (The default is 30).
5. To adjust the number of content packs that can be uploaded to OO Workflow Designer in parallel, update the value of the **max.parallel.cp.upload** property. (The default is 50).
6. To adjust the size limit of content packs that can be uploaded to OO Workflow Designer, update the value of the **upload.max.fileSize.limit** property.
7. Restart OO Workflow Designer. Related topics
   - Change and disable the HTTP/HTTPS ports

Set SSL configuration properties

Setting SSL configuration properties affects secured communication with external apps\services such as git server and OO Central.

Supporting Self-signed Certificates

Self-signed certificates are user generated certificates that have not been signed by a well-known CA and are, therefore, not guaranteed to be authentic. While self-signed certificates can be useful for some testing scenarios, they are not suitable for production use. For more information, see https://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html.

To allow the OO Workflow Designer server accept work with self-signed certificates, the **ssl.support-self-signed** property must be enabled (in the designer-wrapper file). This property is enabled by default.

Host Name Verification

Host name verification ensures that both the host name in the URL and the host name sent back by the server in the digital certificate match. By default, host name verification is enabled, and we recommend enabling this in a production environment to prevent man-in-the-middle attacks.

To disable this verification, you can set the **ssl.verifyHostname** property to **false**.

1. Open the **designer.wrapper.conf** file, located under **<installation dir>/conf/** .
2. Add the **ssl.support-self-signed** and the **ssl.verifyHostName** properties to the **designer.wrapper.conf** file.
3. If you set the **ssl.support-self-signed** value to **true** the OO Workflow Designer server will allow work with self-signed certificates.
4. If you set the **ssl.verifyHostName** value to **false**, this disables the verification of host name matching between the URL and the certificate.
5. Restart OO Workflow Designer.

Troubleshoot 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 **<installation>designer\tomcat\conf\server.xml**.

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

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" allowTrace="false"
clientAuth="false" compression="on" keyAlias="tomcat"
keyPass="changeit" keystoreFile="C:/Program Files/Micro
Focus/Operations Orchestration
Designer/designer/var/security/key.store" keystorePass="changeit"
keystoreType="JKS" maxSwallowSize="-1" maxThreads="200" port="8445"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https"
secure="true" sendReasonPhrase="true" server="OODESIGNER"
sslEnabledProtocols="TLSv1.2" sslProtocol="TLSv1.2"
truststoreFile="C:/Program Files/Micro Focus/Operations Orchestration
Designer/designer/var/security/client.truststore"
truststorePass="changeit" truststoreType="JKS"/>
```

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

Troubleshoot multiple deployment requests

Sending multiple deployment requests can block and even slow down the whole system. In the **secured.properties** file, located in **<installation_dir>/designer/var/security/**, the value of the **max.parallel.cp.creation** key limits the number of simultaneous CP deployments to Central (the key refers only to parallel CP creation; for CP deployments to Central there is a different parameter: **max.parallel.cp.upload** with the default value =50). The default is 30. If users get errors that the limit was reached, the administrator should consider increasing this value.

This behavior might also be used by an attacker together with automated tools in order to create a DoS attack on the system and prevent legitimate users from using it. In such a case, the administrator should identify the malicious IP address/domain and block it.

Configure OO Workflow Designer for FIPS 140-2 level 1 compliance

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

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

OO Workflow Designer 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 OO Workflow Designer to be compliant with FIPS 140-2, you can only revert back to the standard configuration by re-installing OO Workflow Designer.

Prerequisites

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

NOTE:
In order to be FIPS140-2 compliant, you need to turn off LW-SSO.

1. Verify that you are configuring a new installation of OO Workflow Designer, and that it is not in use. You cannot configure an installation of OO Workflow Designer that is in use.
2. Verify that when OO Workflow Designer was installed, it was configured not to start the OO Workflow Designer server after installation:
   - In a silent installation, the `should.start.designer` parameter was set to **no**.
   - In a wizard installation, in the **Connectivity** step, the **Do not start OO Workflow Designer server after installation** check box was selected.
3. Back up the following directories:
   - **<installation dir>\designer\tomcat\webapps\designer.war**
   - **<installation dir>\designer\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**.
   1. Delete everything inside the **<installation dir>\java** folder.

2. Extract the downloaded archive.
3. Copy the **JRE** folder content to **<installation dir>\java**.

5. Copy the **policy.url.3=file:<installation_path>designer/var/security/java.policy** line from the old JDK **java.security** file located in the lib folder into the new JDK **java.security** file.
6. 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 OO Workflow Designer.
7. Install the RSA BSAFE Crypto software files. On the system on which OO Workflow Designer is installed, copy the following to **<java>\lib\ext** (where **<java>** is the directory in which the JRE used by OO Workflow Designer is installed. By default, this is **<installation dir\java**).
   - **<installation dir>\designer\lib\cryptojce-6.2.1.jar**
   - **<installation dir>\designer\lib\cryptojcommon-6.2.1.jar**
   - **<installation dir>\designer\lib\jcmFIPS-6.2.1.jar**
8. Next, follow the steps in the "*Configure the properties in the java security file*" in "*Configure compliance with FIPS 140-2*" topic.

Related topics

- Configure compliance with FIPS 140-2

Configure compliance with FIPS 140-2

Configure compliance with FIPS 140-2

The following list shows the procedures that you need to perform in order to configure OO Workflow Designer 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-Compliant OO Workflow Designer Encryption.
4. Re-encrypt the database password with the new encryption.
5. Start OO Workflow Designer.

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

1. For every provider listed, in the format **security.provider.<nn>=<provider_name>**, increment the preference order number <nn> by two. For example, change a provider entry from: security.provider.1=sun.security.provider.Sun to security.provider.3=sun.security.provider.Sun
2. Add a new default provider (RSA JCE). Add the following provider at the top of the provider list: security.provider.1=com.rsa.jsafe.provider.JsafeJCE
3. Add the RSA BSAFE SSL-J Java Secure Sockets Extension (JSSE) Provider: security.provider.2=com.rsa.jsse.JsseProvider
4. Copy and paste the following line into the **java.security** file to ensure **RSA BSAFE** is used in FIPS 140-2 compliant mode: com.rsa.cryptoj.fips140initialmode=FIPS140_SSL_MODE You can paste this line anywhere in the **java.security** file.
5. Because the default DRBG algorithm ECDRBG128 is not safe (according to NIST), set the security property **com.rsa.crypto.default** to **HMACDRBG**, by copying the following line into the **java.security** file: com.rsa.crypto.default.random=HMACDRBG You can paste this line anywhere in the **java.security** file.
6. Save and exit the **java.security** file.

Configure the encryption properties file and enable FIPS mode

The OO Workflow Designer 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>\designer\var\security**.
2. Open the **encryption.properties** file in a text editor. For example, edit the following file: <installation>\designer\var\security\encryption.properties.
3. Locate `keySize=128` and replace it with `keySize=256`.
4. Locate `secureHashAlgorithm=SHA1` and replace it with `secureHashAlgorithm=SHA256`.
5. Locate `FIPS140ModeEnabled=false` and replace it with `FIPS140ModeEnabled=true`. NOTE: If `FIPS140ModeEnabled=false` does not exist, add `FIPS140ModeEnabled=true` as a new line to the end of the file.
6. Save and close the file.

Create FIPS-Compliant OO Workflow Designer encryption

To create or replace the OO Workflow Designer encryption store file, so that it is FIPS-compliant, see "*Replace the FIPS encryption*" topic.

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.

NOTE: It is recommended to delete all the default CA root certificates that are not in use from the OO Workflow Designer truststore. (The **client.truststore** is located at **<installation>/designer/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.

Re-encrypt the database password with the new encryption

Re-encrypt the database password, as described in the "*Change the database password*" topic.

Start OO Workflow Designer

Related topics

Replace the FIPS encryption
Change and encrypt obfuscate the KeyStore TrustStore password
Encrypt passwords
Changing the database password

Replace the FIPS encryption

Operations Orchestration Workflow Designer complies 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 OO Workflow Designer, you have the option to change the FIPS encryption key.

Change the FIPS encryption key on OO Workflow Designer

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