Hewlett Packard Enterprise

Operations Orchestration Designer

Software Version: 1.2

Windows and Linux

Administer

Document Release Date: September 2017
Software Release Date: September 2017

Support

Visit the HPE Software Support site at: https://softwaresupport.hpe.com/.

Most of the support areas require that you register as an HP Passport user and to sign in. Many also require a support contract. To register for an HP Passport ID, click **Register** on the HPE Support site or click **Create an Account** on the HP Passport login page.

To find more information about access levels, go to: https://softwaresupport.hpe.com/web/softwaresupport/access-levels.

HPE Software Solutions Now accesses the HPSW Solution and Integration Portal website. This site enables you to explore HPE Product Solutions to meet your business needs, includes a full list of Integrations between HPE Products, as well as a listing of ITIL Processes. The URL for this website is https://softwaresupport.hpe.com/km/KM01702731.

Contents

Administrating OO Designer

This document contains information for the OO Designer administrator.

Initial Administration Tasks

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

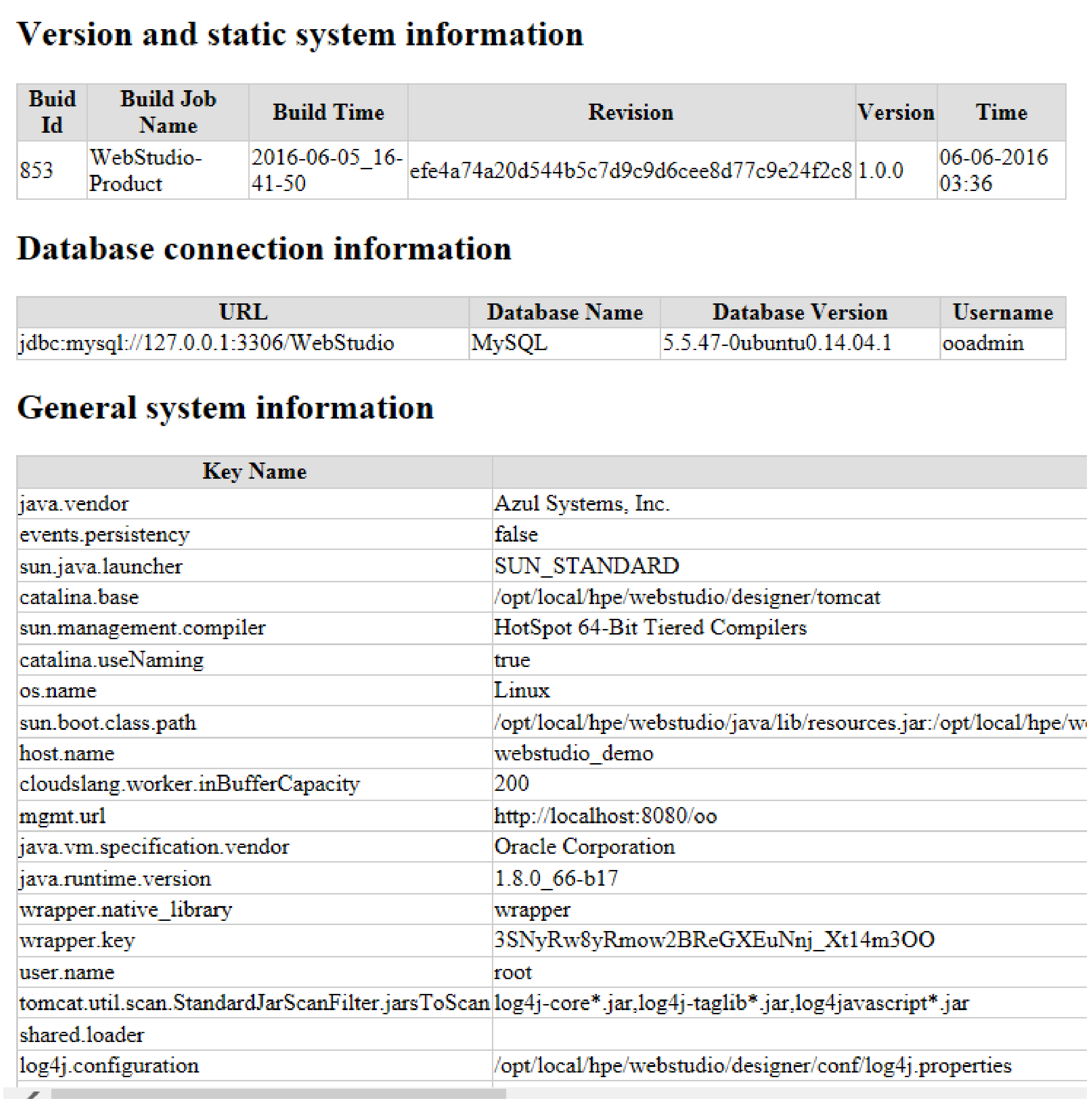
Displaying System Information

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
853	WebStudio-Product	2016-06-05_16-41-50	efe4a74a20d544b5c7d9c9d6cee8d77c9e24f2c8	1.0.0	06-06-2016 03:36

Database connection information

URL	Database Name	Database Version	Username
jdbc:mysql://127.0.0.1:3306/WebStudio	MySQL	5.5.47-0ubuntu0.14.04.1	ooadmin

General system information

Key Name	
java.vendor	Azul Systems, Inc.
events.persistency	false
sun.java.launcher	SUN_STANDARD
catalina.base	/opt/local/hpe/webstudio/designer/tomcat
sun.management.compiler	HotSpot 64-Bit Tiered Compilers
catalina.useNaming	true
os.name	Linux
sun.boot.class.path	/opt/local/hpe/webstudio/java/lib/resources.jar:/opt/local/hpe/w
host.name	webstudio_demo
cloudslang.worker.inBufferCapacity	200
mgmt.url	http://localhost:8080/oo
java.vm.specification.vendor	Oracle Corporation
java.runtime.version	1.8.0_66-b17
wrapper.native_library	wrapper
wrapper.key	3SNyRw8yRmow2BReGXEuNnj_Xt14m3OO
user.name	root
tomcat.util.scan.StandardJarScanFilter.jarsToScan	log4j-core*.jar,log4j-taglib*.jar,log4javascript*.jar
shared.loader	
log4j.configuration	/opt/local/hpe/webstudio/designer/conf/log4j.properties

Configuring a Proxy for OO Designer

If you need to configure a proxy for OO 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.

Changing the Database Password

1. If OO Designer is up and running, stop the OO 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>
   ```

   See "Encrypting Passwords" on the next page.
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.

Changing the Database IP

This section is relevant when you need to configure OO 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;sendStringPara
   metersAsUnicode\=true
   ```

3. Change the IP address/FQDN of the database server.
4. Save the file.
5. Start OO Designer.

Changing the Hostname/IP

If the hostname/IP has changed in OO Designer, you will need to manually update this hostname/IP address.

1. If you are using an OO Designer certificate, generate a new certificate with the new FQDN.

   It is not recommended to use the IP address.

   You will also need to replace the certificate in the OO Designer keystore. See "Replacing the OO Designer TLS Server Certificate" on page 70. See "Replacing the TLS Server Certificate" in the OO Designer *Security and Hardening Guide*.

2. Restart OO Designer.

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.

Encrypting 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>.
   ```

Creating a Prompt for the Password

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

```
C:\Program Files\Hewlett-Packard Enterprise\HPE Operations Orchestration
Designer\designer\bin>encrypt-password.bat

Password <typing will be hidden>:

Confirm password <typing will be hidden>:

<ENCRYPTED>sdf874sgs787fds09ewff==

C:\Program Files\Hewlett-Packard Enterprise\HPE Operations Orchestration
Designer\designer\bin>
```

This will create a prompt for the hidden password inputs.

Setting the System Locale in designer-wrapper.conf

If your OO 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.

Creating a Git Repository

If you do not have a Git server account, you can create a local Git repository. OO 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 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**

Adjusting the Logging Levels

Because logging activity can slow the performance of OO Designer and create very large log files, it is important for OO 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 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.

Changing 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 Designer service.
2. Navigate to the **designer\bin** directory. The default is **C:\Program Files\HPE\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=HPEOODesigner
# Display name of the service
wrapper.displayname=HPE Operations Orchestration 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 Designer service.

Backing Up OO 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 Designer with an existing schema. The installation will fail on the **Start OO Designer** step.
2. Stop the OO Designer service and verify that OO 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 Designer service.

Setting up Disaster Recovery

Planning for Disaster Recovery

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

- Database configuration
- Schema
- OO 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.

Performing Disaster Recovery

OO 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 Designer server(s) by reinstalling all nodes. Use the database configuration, schema, and OO Designer 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 "Set Up Database Environment" on page 21 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 Designer in the primary datacenter.
4. Switch over to the secondary datacenter.

System Tuning

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

For detailed information on database housekeeping (index rebuilding, table statistics updated, and so on), see Preparing the Database Environment.

Set Up Database Environment

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

Introduction to Preparing the Database Environment

This chapter contains information about the types of databases used with OO Designer.

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 document 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 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 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
- Postgres (9.1/9.2/9.3/9.4/9.5/9.6)
- MySQL Community/Standard/Enterprise Server (5.5/5.6/5.7)

See the appropriate deployment chapter below for details:

- "Deploying and Maintaining a Microsoft SQL Server Database" on page 25
- "Deploying and Maintaining an Oracle Database" on page 34
- "Deploying and Maintaining a MySQL Database" on page 42
- "Deploying and Maintaining a Postgres Database" on page 50

Appendices contain additional information that is pertinent to all database types.

Language Support

OO 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.

Important Notes

- This document 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 Designer database, see the database vendor’s documentation and make sure you fully understand each action you perform as described in this guide.
- OO 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.
- This document describes the required database settings for OO Designer. 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 help to make the OO Designer system more robust, by protecting the database from several types of failures.

OO 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 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 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.

See "Additional Guidelines for Microsoft SQL Server" on page 87 for an Always On configuration.

See "Additional Guidelines for Oracle" on page 90 for an Oracle RAC example of a database URL configuration.

Database Security

The OO Designer database is at the heart of the OO Designer system, containing the OO 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 Designer’s performance, and is completely transparent to OO 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 Designer keeps all content and configuration data in the database.

This section will help you prepare for installing OO 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 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 Designer system and reasonable historical data retention—not including long-term database backups.

- The amount of additional disk space required for keeping OO 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.

Deploying and Maintaining a Microsoft SQL Server Database

In order to deploy OO 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 Designer using Microsoft SQL Server, perform the following steps:

1. **Review sizing guidelines**. For details, see "OO Designer Database Sizing" in "Introduction to Preparing the Database Environment" on page 21.
2. **Review Hardware and Software Requirements**. For details, see "System Requirements for Microsoft SQL Server" below.
3. **Configure Microsoft SQL Server**. For details, see "Configuring SQL Server" on page 28.
4. **Create an OO Designer database on Microsoft SQL Server**. For details, see "Manually Creating an OO Designer Database on Microsoft SQL Server" on page 30.
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" on page 87. This step is only relevant if you are using Windows authentication instead of SQL Server authentication.

System Requirements for Microsoft SQL Server

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

Hardware Requirements

For OO Designer database sizing guidelines and hardware requirements, see "OO Designer Database Sizing" and "Hardware Requirements” in "Introduction to Preparing the Database Environment" on page 21.

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 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

The listed service packs are the minimal service packs supported. Newer service packs are also supported unless stated otherwise in the *OO Designer 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 HPE quality assurance personnel.

Database Release			
Version	**32/64Bit**	**Service Pack**	**Operating System**
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 Designer database does not use Unicode collation.

Use one of the following collations depending on your OO 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.

Configuring SQL Server

This section contains information on Microsoft SQL Server and database configuration settings.

You can install an OO 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 Designer)

[1] At the moment, the OO Designer installer supports only SQL authentication. Windows authentication can be configured afterward. See "Additional Guidelines for Microsoft SQL Server" on page 87.

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" section in "System Requirements for Microsoft SQL Server" on page 26.**		
Recovery Model		**Full**	Simple, Full
Allow Snapshot	**True**		

Microsoft SQL Server 2008R2, 2012, 2014, and 2016			
Database Options			
	Mandatory	**Recommended**	**Supported**
Isolation			
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 Designer fails to operate properly using any other type of transaction isolation.

Manually Creating an OO Designer Database on Microsoft SQL Server

During OO Designer setup, a new database can be created automatically by the OO 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.

This section describes the procedure for manually creating an OO Designer 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 once OO Designer 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 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 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.

- 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 Creating 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 Designer does not have permissions to create or modify database objects (limited to DML operation only).

It is highly recommended to grant the OO Designer database user DDL-related privileges, in order to allow OO 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 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 Designer 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 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 Designerdatabase using either one of these models. As OO 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 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 Designer on a daily basis.

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

Timing:

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

Retention:

Retention depends on your business guidelines and regulations.

Creating a Maintenance Plan

Maintaining an OO 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 Designer database in good shape.

Utilities for OO DesignerDatabase Housekeeping

> **Note:** SQL scripts are also available in the OO 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 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 Designer is least active.

Always On Support

Connection to SQL Server "Always On" is supported including the “MultiSubnetFailover” feature, but have not been certified for OO Designer.

See "Additional Guidelines for Microsoft SQL Server" on page 87 for additional information.

High Availability Setup

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

Normally, OO 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 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 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 Designer should be halted, reconfigured (edit the **database.properties** file), and then started again once the new database instance is available.

See the *OO Designer Installation Guide* for details on reconfiguring the database connection details.

Deploying and Maintaining an Oracle Database

In order to deploy OO 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 Designer using Oracle, perform the following steps:

1. **Review sizing guidelines**. For details, see "OO Designer Database Sizing" in "Introduction to Preparing the Database Environment" on page 21.
2. **Review Hardware and Software Requirements**. For details, see "System Requirements for Oracle" below.
3. **Configure an Oracle Database**. For details, see "Configuring an Oracle Database" on page 36.
4. **Create a Database**. For details, see "Manually Creating an OO Designer Database on an Oracle Instance" on page 38.
5. (Optional) **Connect OO Designer to an Oracle RAC environment**. For details, see "Oracle Real Application Cluster (RAC)" in "Additional Guidelines for Oracle" on page 90. This step is only relevant if you are using OO Designer in an Oracle RAC environment.

System Requirements for Oracle

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

Hardware Requirements

For OO Designer database sizing guidelines and hardware requirements, see "OO Designer Database Sizing" and "Hardware Requirements” in "Introduction to Preparing the Database Environment" on page 21.

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 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 Designer *Release Notes*.

See the Oracle documentation for supported platforms.

Note that 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 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 HPE 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).

Configuring an Oracle Database

This section contains information on the Oracle instance and database configuration settings.

You can install an OO 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 that all values reflect resources required by OO Designer. If OO 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

Oracle Database 11gR2 / 12cR1			
Instance/Server Options			
	Mandatory	**Recommended**	**Supported**
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 that all values reflect resources required by OO Designer. If OO Designer shares the Oracle instance with other users, these values should be added to whatever is currently consumed by others.

Manually Creating an OO Designer Database on an Oracle Instance

During OO Designer setup, a new database can be created automatically by the OO 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".

This section describes the procedure for manually creating an OO Designer 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 Designer 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 theOO Designer installation. In such a case, you (or your organization’s DBA) should create the user (database) first, and then let the OO 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.

- 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 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 Creating Database Objects

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

It is highly recommended to grant the OO Designer database user DDL-related privileges, in order to allow OO 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 Designer installation zip file under **docs\sql**.
2. Edit the **oracle.sql** file and prefix each object with the OO Designer user, to make sure the objects are created under the OO Designer user.
3. Connect to the OO Designer database as the relevant user, who is permitted to create and modify database objects in the OO 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 Designer user.

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

Backing up the OO Designer Database

Oracle databases can be backed up using several tools, such as **expdp** and **RMAN**. An OO 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 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 Designer on a daily basis.

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

Timing:

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

Retention:

Retention depends on your business guidelines and regulations.

Creating a Maintenance Plan

Maintaining an OO 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 Designer database in good shape.

Utilities for OO Designer Database Housekeeping

> **Note:** SQL scripts are also available in the OO 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 (withoutOO Designer system downtime) requires an enterprise grade database. Make sure you are running an enterprise version of Oracle 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 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 Designer.

Deploying and Maintaining a MySQL Database

In order to deploy OO 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 Designer using MySQL, perform the following steps:

1. **Review sizing guidelines**. For details, see "OO Designer Database Sizing" in "Introduction to Preparing the Database Environment" on page 21.
2. **Review Hardware and Software Requirements**. For details, see "System Requirements for MySQL" below.
3. **Configure MySQL** . For details, see "Configuring MySQL" on page 44.
4. **Create an OO Designer database on MySQL** . For details, see "Manually Creating an OO Designer Database on MySQL" on page 46.

System Requirements for MySQL

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

Hardware Requirements

For OO Designer database sizing guidelines and hardware requirements, see "OO Designer Database Sizing" and "Hardware Requirements" in "Introduction to Preparing the Database Environment" on page 21.

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 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.

See the MySQL documentation for supported platforms.

MySQL Connector

The MySQL connector jar is not provided with the OO Designer installation. Download and place it under each OO 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 HPE quality assurance personnel.

Database Release			
Version	**32/64Bit**	**Patch**	**Operating System**
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 Designer database uses the utf8_bin collation.

Configuring MySQL

This section contains information on MySQL and OO Designer 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	**Default or none**		
innodb_file_per_table		**1**	
innodb_thread_concurrency		**0**	
table_open_cache		**1000**	
sort_buffer_size		**2M**	
read_buffer_size		**2M**	

MySQL 5.5, 5.6 and 5.7			
Instance/Server Options			
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.
- Using none-default sql-mode options may result in unexpected behavior.

Manually Creating an OO Designer Database on MySQL

During OO Designer setup, a new database can be created automatically by the OO 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.

This section describes the procedure for manually creating an OO Designer 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 Designer 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 the OO Designer installation. In such a case, you (or your organization’s DBA) should create the database first, and then let the OO 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).

> - **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 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 Creating Database Objects

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

It is highly recommended to grant the OO Designer database user DDL-related privileges, in order to allow OO 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 Designer installation zip file under **docs\sql**.
2. Connect to the OO Designer database as the relevant user, permitted to create and modify database objects in the OO Designer 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 Designer databases created on MySQL, such as backing up the database, checking database integrity, handling index fragmentation, and monitoring the database.

Backing up the OO Designer Database

You can back up MySQL database using several tools, such as **mysqldump** or **mysqlbackup**. You can back up the OO 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 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 Designer on a daily basis.

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

Timing:

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

Retention:

Retention depends on your business guidelines and regulations.

Creating a Maintenance Plan

Maintaining an OO 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 Designer database in good shape.

Recommended Utility for Database Maintenance

In order to keep the OO 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 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 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.

Deploying and Maintaining a Postgres Database

In order to deploy OO 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 Designer using Postgres, perform the following steps:

1. **Review sizing guidelines**. For details, see "OO Designer Database Sizing" in "Introduction to Preparing the Database Environment" on page 21.
2. **Review Hardware and Software Requirements**. For details, see "System Requirements for Postgres" on the next page.
3. **Configure Postgres**. For details, see "Configuring Postgres" on page 52.
4. **Create OO Designerdatabase on Postgres**. For details, see "Manually Creating an OO Designer Database on Postgres" on page 53.

System Requirements for Postgres

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

Hardware Requirements

For OO Designer database sizing guidelines and hardware requirements, see "OO Designer Database Sizing" and "Hardware Requirements" in "Introduction to Preparing the Database Environment" on page 21.

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 Designer:

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

Postgres Database Releases	
Version	**Type**
9.6	x86 32-bit
	x86 64-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 HPE quality assurance personnel.

Database Release		
Version	**32/64Bit**	**Operating System**
Postgres 9.2.3	64 Bit	Windows 2012 Standard Edition 64 Bit
Postgres 9.1.9	64 Bit	Red Hat Enterprise Linux 6.3 64 Bit
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

Language Support

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

Configuring Postgres

This section contains information on OO Designer 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 9.1 - 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 Creating an OO Designer Database on Postgres

During OO Designer setup, a new database can be created automatically by the OO 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.

This section describes the procedure for manually creating an OO Designer 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 Designer 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 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 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.

> - 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 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.

Manually Creating Database Objects

Once the database and role 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.

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

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

Postgres Database Maintenance

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

Backing up the OO 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 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 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 Designer on a daily basis.

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

Timing:

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

Retention:

Retention depends on your business guidelines and regulations.

Creating a Maintenance Plan

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

Recommended Utility for Database Maintenance

In order to keep the OO 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 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 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.

Security and Hardening

This chapter explains how to deploy and manage OO Designer instances in a secure manner, and how to configure security hardening for OO Designer.

Security Overview

This section provides an overview of the security models and recommendations for a secure implementation of OO Designer. This includes subjects such as authentication, authorization, encryption, and more. Where relevant, there are references to other OO Designer documents, 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 Designer users are authenticated via HPE LWSSO, (HPE 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 HPE systems that support LWSSO. For example, if users have logged onto another HPE product web client that has LWSSO enabled, they can enter the OO Designer application directly, bypassing the OO 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 Designer uses LWSSO for authentication.
- **TLS Encryption** – By default, OO 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 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. For more information, see "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73

OO Designer Security Hardening

The Hardening chapter provides recommendations for safeguarding your OO 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 Designer system, it is necessary to secure both OO Designer and the computing environment (for example, the infrastructure and the operating system) upon which the application runs.

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

Physical Security

HPE Software recommends that OO Designer is protected by physical security controls defined by your organization. The OO 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.

Secure Installation Guidelines

Supported Operating Systems

For the types and versions of supported operating systems, see the *System Requirements* section of *OO Designer* Installation 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
- Recommended: Put up a firewall to protect against DDoS.

Tomcat Hardening

When you install OO 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 Designer:

Installing OO 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 Designer	• Windows: The Windows service runs as the system user or a specific user (the user must have access to the OO Designer installation directory) • Linux: Any standard user who is able to run a Java process

See also the recommendations in the CIS Apache Tomcat documentation.

Security Protection in OO Designer

The following security features have been designed to protect the security of your OO Designer installation.

CSRF Protection

OO 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 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 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 Designer User Guide* describes the basic OO Designer topology, high availability, and load balancer security.

Communication Channel Security

Supported Protocols and Configuration

OO Designer supports the TLS protocol.

For more information, see "Replacing the OO Designer TLS Server Certificate" on page 70.

The *OO Designer* ports are defined by the administrator during the installation.

Channel Security

OO Designer supports the following secure channel:

Channel (Directed)	Supported Secure Protocol
Browser → OO Designer	For a secure channel, use the TLS communication for encryption and Client Certificate for authentication.

User Management and Authentication

Authentication Model

There are two methods to authenticate access to OO Designer:

- Single sign on (LWSSO) - supported at the backend
- IDM Service - supported with the new installer

HPE LWSSO is used to authenticate user access to OO Designer at the backend and IDM service for the installer.

Database Authentication

OO 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" on page 87 ".
- See "Changing the Database Password" in the *OO Designer Administration Guide*.

Configure OO Designer in IDM Mode

When OO Designer is connected to IDM, authentication is enabled by default at startup. To use OO Designer in the IDM mode, you must switch from the LWSSO security profile to the IDM security profile.

Switch to the IDM security profile

1. Open the **<installation dir>\designer\conf\designer-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 LWSSO security profile, update the value of the `-Doverride.startup.mode` parameter to `LWSSO`.
3. Save and close the `designer-wrapper.conf` file.
4. Restart the Designer service.

Update the IDM Priperties

You can configure IDM properties in the **<installation dir>\designer\conf\idm.properties** file. After configuring a property, save and close the file. Then restart the Designer service.

The following table describes the properties in the idm.properties file.

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

Example of the idm.properties file

```
idm.entry.point.return.url = https://UTIUD5.emea.cpqcorp.net:8450/oo-designer
idm.provider.port = 8450
idm.provider.hostname = UTIUD5.emea.cpqcorp.net
idm.provider.integration.acct.username = idmTransportUser
idm.provider.integration.acct.password = {ENCRYPTED}
RLG4MCHLBdgHNJonXqo/7dd6/YHX7TAk3tcK4X6ksKc=
idm.tenant = OO_Designer
idm.signing.key = {ENCRYPTED}
5rLWIqOuy2Imke6Ot3ybezxhJq1fH8Jjf00wAqWzFPZAgCi4o2UW+SDRMDyI3fq2
idm.provider.protocol = https
```

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 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 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 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 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 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. For more information, see "Changing the HTTP/HTTPS Ports or Disabling the HTTP Port" on page 76.

Encryption Administration

- **Recommended Encryption Best Practices**

  In order to reach higher levels of security and cryptography, it is recommended to configure OO Designer to be compliant with Federal Information Processing Standards (FIPS) 140-2. OO 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 Designer for FIPS 140-2 compliance, it uses the following security algorithm:

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

- See "Configuring Compliance with FIPS 140-2" on page 83.

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 Designer server, you need to install a digital certificate on the server side.

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

Certificate Location

Installations of OO 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 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 Designer service.

Replacing the OO Designer Self-signed Certificate

It is recommended to replace the OO Designer self-signed certificate after a new installation of OO 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.

For more information, see "Replacing the OO Designer TLS Server Certificate" on page 70.

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**

No sensitive data is kept in the log files in OO 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. For instructions, see "Replacing the OO Designer TLS Server Certificate" on page 70.

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

When you install OO Designer, you need to configure at least one available port for the OO 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 Designer and the other components, see "Network and Communication Security" on page 61

How do I configure self-signed SSL certificates for OO Designer?

During the installation of OO 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. HPE recommends working with a certificate from a custom-root CA or from a well-known CA.

For more information about configuring certificates for OO Designer, see "Encrypting the Communication Using a Server Certificate" on page 70.

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?

See "Encryption" on page 64.

Which authentication mechanism does OO Designer support?

OO Designer supports Client Certificate and LWSSO. See "User Management and Authentication" on page 62.

> **Note:** Client certificates are not supported.

Is OO Designer FIPS 140-2 compliant?

Yes. For more information, see "Configuring Compliance with FIPS 140-2" on page 83.

Can I limit the OO Designer user IP address?

No, this is not supported at the moment.

Hardening OO Designer

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

Security Hardening Recommendations

1. Install OO Designer.

   For more information, see the *OO Designer Installation Guide*.

2. (Optional) Configure OO Designer for FIPS 140-2 Compliance. See "Configuring OO Designer for FIPS 140-2 Level 1 Compliance" on page 81.
3. Configure the OO Designer server certificate for TLS encryption and client certificate for strong authentication (mutual).

   Note: This can be done during installation.

4. Harden the OO Designer server by removing the HTTP port and replacing the passwords of the KeyStore and TrustStore with strong passwords. See "Changing the HTTP/HTTPS Ports or Disabling the HTTP Port" on page 76 and "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73.
5. Remove the RC4 cipher from the SSL-supported ciphers. See "Removing the RC4 Cipher from the SSL-supported Ciphers" on page 75.
6. Harden/secure the operating system and database.
7. In the Windows and SQL server environment, configure OO Designer to work with Windows authentication. See ""Additional Guidelines for Microsoft SQL Server" on page 87".

Default Security Settings

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

- **Authentication** – OO Designer uses LWSSO for authentication.
- **TLS Encryption** – By default, OO 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 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. For more information, see "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73

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

Recommendations:

- It is recommended to replace the OO Designer self-signed certificate after a new installation of OO 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 Designer service.
- It is recommended to clear the console after using Keytool or to use the prompt for password inputs.

Encrypting the Communication Using a Server Certificate

Replacing the OO 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 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 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.

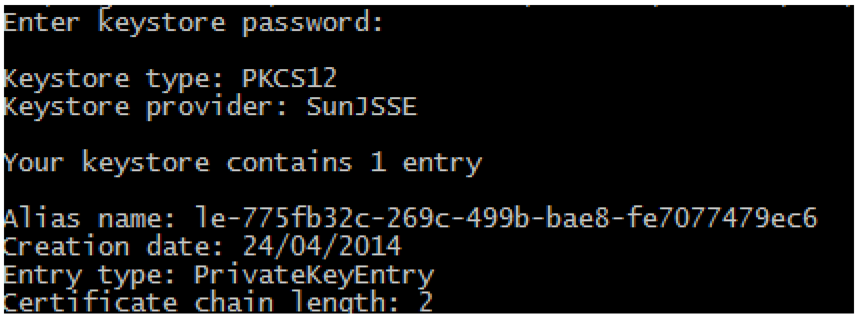
> **Note:** Make a note of the password that you provide. You will need this password for the private key when you input the KeyStore passphrase later in this procedure.
>
> Make sure to choose a strong password.

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.

   ```
   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 OO 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. Follow the instructions in "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73.

   It is also recommended to change the default “changeit” password in the automatically generated KeyStore in the OO Designerserver. See "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73.

8. Start OO Designer.

Importing a CA Root Certificate to the OO Designer TrustStore

If you are using a custom root certificate for OO 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 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 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 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 Designer.

Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password

Changing the KeyStore, TrustStore, and Server Certificate Passwords in the OO Designer Configuration

1. Make sure that OO Designer is running.

   > **Note:** Before doing this step, make sure that there are encrypted passwords. For information about how to encrypt a password, see "Encrypting Passwords" on the next page.

2. Stop the OO 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**.

   a. Locate the HTTPS connector. For example:

   ```
   keyPass="changeit" keystoreFile="C:/Program Files/Hewlett-Packard
   Enterprise/HPE 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/Hewlett-
   ```

```
Packard
Enterprise/HPE 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.

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

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

b. Save the file.

5. Edit the **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>
   ```

   For information about how to encrypt or obfuscate a password, see "Encrypting Passwords" below.

6. Start the OO Designer service.

Encrypting Passwords

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

Our recommendation is to use encryption.

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

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

Encrypting Passwords

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

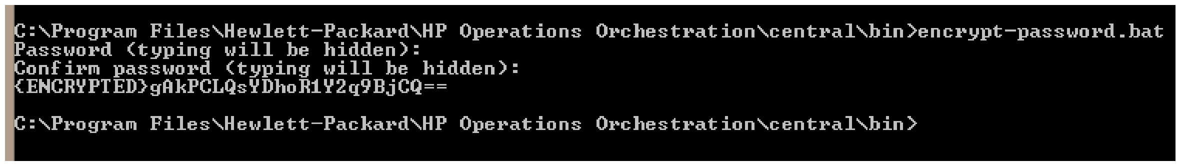
   Note: You can either use `-p` as the flag to encrypt the password or `--password`.

   The encrypted password should appear as follows:

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

Creating a Prompt for the Password

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



```
C:\Program Files\Hewlett-Packard\HP Operations Orchestration\central\bin>encrypt-password.bat
Password (typing will be hidden):
Confirm password (typing will be hidden):
{ENCRYPTED}gAkPCLQsYDhoR1Y2q9BjCQ==

C:\Program Files\Hewlett-Packard\HP Operations Orchestration\central\bin>
```

This will create a prompt for the hidden password inputs.

Removing the RC4 Cipher from the SSL-supported Ciphers

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

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

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

1. Open the **<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.

Changing the HTTP/HTTPS Ports or Disabling the HTTP Port

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

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

By default, the connectors look as follows.

HTTP connector:

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

HTTPS connector:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false"
compression="on" keyAlias="tomcat" keyPass="changeit" keystoreFile="C:/Program
Files/Hewlett-Packard Enterprise/HPE 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,TLSv1.1,TLSv1.2"
truststoreFile="C:/Program Files/Hewlett-Packard Enterprise/HPE Operations
Orchestration Designer/designer/var/security/client.truststore"
```

```
truststorePass="changeit" truststoreType="JKS"/>
```

By default, in the installer, HTTP is disabled.

Important! If you change or disable one of the OO Designer ports in the **server.xml** file, you will also need to update the **designer-wrapper.conf** file to point to the OO Designer URL with the updated port. Also, make sure to check the load balancer configurations.

Changing Port Values

To change the values of one of the ports:

1. Edit the **server.xml** file located in **<installation_dir>/designer/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 OO Designer.

Disabling the HTTP Port

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

1. Edit the **server.xml** file, located in **<installation_dir>/designer/tomcat/conf/**.
2. Locate the HTTP connector, and delete or comment out the line.
3. Edit the **secured.properties** file, located in **<installation_dir>/designer/var/security/**.
4. Make sure that the **is.secured.cookie** property is activated, as follows:

   ```
   is.secured.cookie=true
   ```

   This adds a secured attribute to the CSRF cookie.

5. Import the trusted root certificate authority (CA) to the OO Designer **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.

6. Save the file.
7. Restart OO Designer.

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

Configuring the Secured Properties

When OO Designer was installed, decisions were made, which may affect security: the LWSSO 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 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 Designer is installed with no HTTP connection. However, if OO 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 "Disabling the HTTP Port" under "Changing the HTTP/HTTPS Ports or Disabling the HTTP Port" on page 76.

3. To verify that the LWSSO 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 LWSSO 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 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 Designer, update the value of the **upload.max.fileSize.limit** property.
7. Restart OO Designer.

Setting 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 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 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 Designer.

Troubleshooting the HTTPS Connector

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

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

All the HTTPS connector parameters are in the Tomcat configuration file located at **<installation>designer\tomcat\conf\server.xml**.

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

```
<Connector SSLEnabled="true" clientAuth="false" keyAlias="tomcat"
keystoreFile="C:/HPE/oo/designer/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="TLSv1.2"/>
```

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

Troubleshooting 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.

Configuring OO Designer for FIPS 140-2 Level 1 Compliance

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

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

OO 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 Designer to be compliant with FIPS 140-2, you can only revert back to the standard configuration by re-installing OO Designer.

Prerequisites

Before configuring OO 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 LWSSO.

1. Verify that you are configuring a new installation of OO Designer, and that it is not in use.

   You cannot configure an installation of OO Designer that is in use.

2. Verify that when OO Designer was installed, it was configured not to start the OO 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 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**.
   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. 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 Designer.
7. Install the RSA BSAFE Crypto software files. On the system on which OO Designer is installed, copy the following to **<java>\lib\ext\** (where **<java>** is the directory in which the JRE used by OO 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**

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

Configuring Compliance with FIPS 140-2

The following list shows the procedures that you need to perform in order to configure OO 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 Designer Encryption.
4. Re-encrypt the database password with the new encryption.
5. Start OO Designer.

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.

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.

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

The OO 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.

Step 3: Create FIPS-Compliant OO Designer Encryption

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

> **Note:** AES has three approved key lengths: 128/192/256 by NIST SP800-131A publication.
>
> The following secure hash algorithms are supported in FIPS: SHA1, SHA256, SHA384, SHA512.

> **Note:** It is recommended to change the passwords of the key.store (and its private key entry) and

the truststore. See "Changing and Encrypting/Obfuscating the KeyStore/TrustStore Password" on page 73

Note: It is recommended to delete all the default CA root certificates that are not in use from the OO 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.

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

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

Step 5: Start OO Designer

Replacing the FIPS Encryption

OO 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 Designer, you have the option to change the FIPS encryption key.

Changing the FIPS Encryption Key on OO 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.

Additional Guidelines for Microsoft SQL Server

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

Using Windows Authentication to Access Microsoft SQL Server Databases

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

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

Configuring OO Designer to Work with Windows Authentication

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

To enable OO 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.

> **Note:** 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_
NAME];sendStringParametersAsUnicode\=true;integratedSecurity\=true;
```

There might be a need to copy sqljdbc_auth.dll to the system path of the OO 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.

Configuring OO Designer to Work with Always On

OO 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
```

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

Note that Multi-Subnet Failover scenarios in conjunction with SQL Server 2014 Always On cluster were not certified with OO 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.

Additional Guidelines for Oracle

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

Oracle Real Application Cluster (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 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 Designer only supports connecting to a single 12c SCAN listener.

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

Configuring OO 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 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 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" on the next page.

Here is an example of how a load balancing connection string would appear in the OO 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.

Installation Wizard 'Other database' Option

This appendix contains information about the **Other database** option in the OO 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 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 Designer installation)

When you install OO 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 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 Designer installation wizard must be unzipped and extracted to a folder form.
4. **sqljdbc_auth.dll** must be downloaded and copied to the OO Designer installation wizard under <installer_folder>\java\bin. This dll file is not provided along with the OO 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 Designer installer by executing <**installer_folder>\installer.bat** as the same Windows account that owns the OO Designer database.
2. In the **Connectivity** page, select the **Do not start OO 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:
   a. Verify **sqljdbc_auth.dll** is located in **<installation>\java\bin**.
   b. Configure the service to run under the same Windows account that owns the OO Designer database.
   c. Start the OO Designer service and verify successful connection to the database.

Repeat the steps above for all OO 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	

Field	Value	Comments
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 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)))	

Field	Value	Comments
Username	<PRE-CREATED-DATABASE>	
Password	< PRE-CREATED-DATABASE-PASSWORD>	

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Administer (Operations Orchestration Designer 1.2)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to oo_ie@hpe.com.

We appreciate your feedback!

Hewlett Packard
Enterprise

Java™
COMPATIBLE