Oracle® Cloud

Migrating Oracle Java Cloud Service Instances to Oracle Cloud Infrastructure

Release 19.1.4
F14247-01
March 2019

Oracle Cloud Migrating Oracle Java Cloud Service Instances to Oracle Cloud Infrastructure, Release 19.1.4

F14247-01

Primary Author: TJ Palazzolo

Contents

4 Complete the Post-Migration Tasks

1

Learn About Migrating to Oracle Cloud Infrastructure

Learn about the benefits to migrating your existing Oracle Java Cloud Service instances to Oracle Cloud Infrastructure, and get an overview of the migration process and tools.

Topics:

Why Migrate to Oracle Cloud Infrastructure

Oracle encourages you to migrate your existing cloud resources from Oracle Cloud Infrastructure Classic regions. You can gain several advantages by doing so.

In Oracle Cloud, you provision resources in specific regions, which are localized to geographic locations. A region supports either the Oracle Cloud Infrastructure Classic or Oracle Cloud Infrastructure platform.

Oracle Cloud Infrastructure is Oracle's more modern infrastructure platform that's based on the latest cloud technologies and standards. It typically provides better performance than Oracle Cloud Infrastructure Classic. Oracle Cloud Infrastructure also has more predictable pricing and lower costs in terms of Oracle Compute Units (OCPUs) per hour. Most importantly, Oracle continues to invest in Oracle Cloud Infrastructure, including the addition of new regions, services, and features. See Data Regions for Platform and Infrastructure Services.

You can benefit from these additional administrative features in Oracle Cloud Infrastructure when you migrate your cloud resources from Oracle Cloud Infrastructure Classic:

- Organize cloud resources into a hierarchy of logical compartments.
- Create fine-grained access policies for each compartment.

About the Migration Scope

Before you migrate your existing Oracle Java Cloud Service instances to Oracle Cloud Infrastructure, ensure that the service instance meets the prerequisites for the migration.

Oracle does *not* currently support the migration of Oracle Java Cloud Service instances that meet any of these conditions:

- Oracle Fusion Middleware products and applications are deployed to the service instance.
- Custom applications that use the Oracle Application Development Framework (ADF), Oracle Platform Security Services (OPSS), or Oracle Web Services Manager (OWSM) are deployed to the service instance.
- Java Message Service (JMS) resources are deployed to the service instance.
- The service instance includes multiple domain partitions.
- The service instance is configured to use Oracle Identity Cloud Service for authentication.
- The service instance includes an Oracle-managed or customer-managed load balancer.

This guide does not include detailed procedures on the configuration of basic Oracle Cloud Infrastructure security, network and storage resources that might be required to support your Oracle Java Cloud Service instance. Instead, this guide provides references to the Oracle Cloud Infrastructure documentation as appropriate.

Most service instances connect to one or more databases in order to access your application schemas. This guide does not include the detailed procedure for migrating these application databases from Oracle Cloud Infrastructure Classic to Oracle Cloud Infrastructure. See Migrate a single-instance database from Oracle Database Cloud Service to a Virtual Machine DB System.

This guide does not include detailed procedures on the migration of Oracle WebLogic Server users, groups, roles, or policies from the source instance to the target instance. This includes users and groups that are defined in the default WebLogic Server authentication provider (embedded LDAP), as well as users and groups that exist in an external identity provider like an LDAP server. Refer to the migration (export and import) capabilities of your identity provider.

About Oracle Cloud Infrastructure

Get familiar with basic Oracle Cloud Infrastructure security, network, and storage concepts, compared to their equivalent concepts in Oracle Cloud Infrastructure Classic.

Cloud resources in Oracle Cloud Infrastructure are created in logical compartments. You also create fine-grained policies to control access to the resources within a compartment.

You create instances within an Oracle Cloud Infrastructure region. You also specify an availability domain (AD), if supported in the selected region. Oracle Cloud Infrastructure Classic does not use availability domains.

A virtual cloud network (VCN) is comprised of one or more subnets, and an instance is assigned to a specific subnet. In Oracle Cloud Infrastructure Classic, you assign instances to IP networks or the shared network. Typically, you create one subnet for the shared network, and create a separate subnet for each IP network in Oracle Cloud Infrastructure Classic. Note that unlike Oracle Cloud Infrastructure Classic, Oracle Cloud Infrastructure does not allow you to reserve IP addresses for platform services.

A subnet's security lists permit and block traffic to and from specific IP addresses and ports. In Oracle Cloud Infrastructure Classic, an instance's access rules provide similar capabilities, although security lists are configured at the subnet level.

Instances can communicate with resources outside of Oracle Cloud by using Oracle Cloud Infrastructure FastConnect, which provides a fast, dedicated connection to your on-premises network. This service is equivalent to Oracle Cloud Infrastructure FastConnect Classic. Alternatively, use IPSec VPN in Oracle Cloud Infrastructure as a replacement for VPN as a Service (VPNaaS) or Corente in Oracle Cloud Infrastructure Classic.

A bucket in Oracle Cloud Infrastructure Object Storage can be used to store files and share them with multiple instances. A user's generated authentication token (auth token) is required to access the bucket. Oracle Cloud Infrastructure Object Storage Classic provides the same service in Oracle Cloud Infrastructure Classic, but does not use auth tokens.

To learn more, see Key Concepts and Terminology in the Oracle Cloud Infrastructure documentation.

In Oracle Cloud Infrastructure Classic, you can create rules that automatically scale an instance. You must scale instances in Oracle Cloud Infrastructure manually.

In Oracle Cloud Infrastructure Classic, you can create an instance that uses Oracle Identity Cloud Service for authentication. This configuration is not currently available in Oracle Cloud Infrastructure.

About Oracle Cloud Infrastructure Users and Groups

Use the Identity and Access Management (IAM) system in Oracle Cloud Infrastructure to manage users, groups, and policies.

For example, the following Oracle Cloud Infrastructure policy grants members of the group `MyGroup` all privileges to all resources in the compartment `MyCompartment`:

```
Allow group MyGroup to manage all-resources in compartment MyCompartment
```

By default, this system is also configured to use Oracle Identity Cloud Service as a federated identity provider. Therefore, when you define policies in Oracle Cloud Infrastructure, you can reuse existing users and groups in Oracle Identity Cloud Service. You can either add users to a new group in Oracle Cloud Infrastructure, or map an existing Oracle Identity Cloud Service group to an Oracle Cloud Infrastructure group.

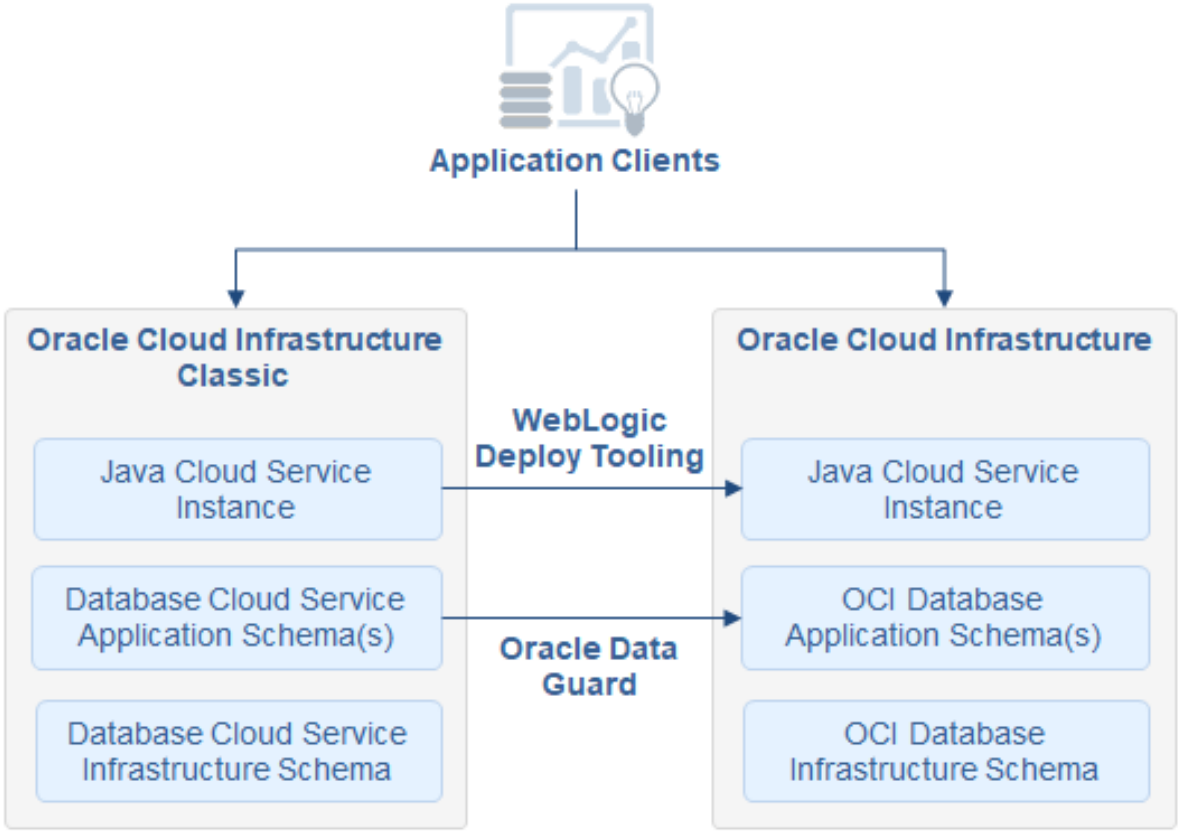
While policies control access to services in Oracle Cloud Infrastructure, administrator roles control access to platform services that are found only on the My Services Dashboard. Assign Oracle Identity Cloud Service users and groups to administrator roles in order to grant them access to services that are not found in Oracle Cloud Infrastructure.

- Common Policies in the Oracle Cloud Infrastructure documentation
- Federating with Oracle Identity Cloud Service in the Oracle Cloud Infrastructure documentation
- Create a Service Administrator in *Getting Started with Oracle Cloud*

About the Migration Task Flow

Get an overview of the process that you use to migrate your existing Oracle Java Cloud Service instances to Oracle Cloud Infrastructure.

The following diagram shows the migration topology for a typical Oracle Java Cloud Service instance.



At a high level, the migration process is comprised of these tasks:

1. Prepare for the migration and perform any prerequisite tasks in Oracle Cloud Infrastructure.
2. Create the target Oracle Java Cloud Service in an Oracle Cloud Infrastructure region.
3. Use Oracle Data Guard to migrate any application databases in Oracle Cloud Infrastructure Classic regions to Oracle Cloud Infrastructure Database.
4. Use the Oracle WebLogic Server Deploy Tooling to discover and export the domain configuration, applications and other supporting files from your source Oracle Java Cloud Service instance.
5. Use the Oracle WebLogic Server Deploy Tooling to update the domain configuration on your target Oracle Java Cloud Service instance and to deploy your applications.
6. Test your applications on the target instance, and perform any other post-migration tasks.

About the Migration Tooling

You can use the Oracle WebLogic Server Deploy Tooling software to automate many of the tasks involved in migrating an Oracle Java Cloud Service instance to Oracle Cloud Infrastructure.

Oracle WebLogic Server Deploy Tooling is an open-source project. It provides scripts that enable you to discover and export the configuration and application files from one Oracle WebLogic Server domain, and then import the configuration and applications into another existing domain.

Oracle WebLogic Server Deploy Tooling exports a domain as a YAML file, which is referred to as the metadata model. It supports a placeholder syntax to dynamically insert variables from an external file into the model file. For example, you can define your Lightweight Directory Access Protocol (LDAP) or database passwords as variables in an external file.

When updating a domain, the YAML metadata model needs to describe only the resources that you want to add or update. For example, if a data source that is defined in the model is already created in the target domain, the tool does not try to recreate the data source. Instead, the tool updates the data source if the model description is different than the existing data source configuration. Similarly, if an application is already deployed, the tool compares the binaries and determines whether the application needs to be redeployed.

To learn more, see the Oracle WebLogic Server Deploy Tooling project on GitHub.

2

Prepare to Migrate Oracle Java Cloud Service to Oracle Cloud Infrastructure

Before you migrate your service instances from Oracle Cloud Infrastructure Classic to Oracle Cloud Infrastructure, understand how the migration affects your existing instances, identify the necessary compute shapes, and create the network to support your migrated service instances.

Topics:

About Downtime Requirements

The migration process does not affect the availability of your existing Oracle Java Cloud Service instance in Oracle Cloud Infrastructure Classic. This instance continues to run and can serve client requests during this process.

You can run the `discoverDomain` script in the Oracle WebLogic Server Deploy Tooling on your Oracle WebLogic Server domain while it is running. The script does not modify your domain or significantly affect its performance.

After a service instance is migrated successfully, clients can be rerouted to the new instance in Oracle Cloud Infrastructure.

Select Oracle Cloud Infrastructure Shapes

Identify the compute shapes that provide similar IaaS resources in Oracle Cloud Infrastructure to the shapes that you're currently using for your service instances in Oracle Cloud Infrastructure Classic.

A compute shape defines the IaaS resources, such as OCPUs and memory, that are available to a specific node in a service instance. Oracle Cloud Infrastructure and Oracle Cloud Infrastructure Classic each has its own set of standard compute shapes. See:

- About Shapes in *Using Oracle Cloud Infrastructure Compute Classic*
- Compute Shapes in the Oracle Cloud Infrastructure documentation

To ensure that a migrated service instance has the same performance characteristics as the original instance, and can support an equivalent workload, choose Oracle Cloud Infrastructure shapes that most closely map to the Oracle Cloud Infrastructure Classic shapes that you specified when you created the instance.

You must also confirm that the chosen shapes are available in your Oracle Cloud tenancy. Oracle configures shape limits for an Oracle Cloud Infrastructure region, or

for a specific availability domain within a region. You can use the console to view the current shape limits for your tenancy, and to request a limit increase if necessary. See Service Limits in the Oracle Cloud Infrastructure documentation.

Unlike nodes in Oracle Cloud Infrastructure Classic, you cannot change the shape of an existing node (scale up or down) in Oracle Cloud Infrastructure. However, you can add storage to an existing node if necessary.

Design the Oracle Cloud Infrastructure Network

Before you migrate your service instances from Oracle Cloud Infrastructure Classic to Oracle Cloud Infrastructure, you must design and implement a virtual cloud network (VCN) to support your migrated service instances.

You can create new Oracle Cloud Infrastructure compartments, VCNs, and subnets for your service instances, or you can use existing ones. See these topics in the Oracle Cloud Infrastructure documentation:

- Managing Compartments
- VCNs and Subnets
- Security Lists

Consider the following guidelines when you create or select a network for your service instances:

- If instances communicate using the default shared network in Oracle Cloud Infrastructure Classic, then use a single subnet for these instances.
- If instances are on separate IP networks in Oracle Cloud Infrastructure Classic, then use separate subnets for these instances.
- A VCN should have an address range that includes all of the IP networks in Oracle Cloud Infrastructure Classic that need to communicate. Alternatively, configure peering between multiple VCNs.
- A subnet should have at least the same number of addresses as the corresponding IP network in Oracle Cloud Infrastructure Classic.
- If an instance was created in Oracle Cloud Infrastructure Classic without public IP addresses, then use a private subnet for this instance.
- If custom access rules were created for an instance in Oracle Cloud Infrastructure Classic to control communication to or from the instance, then create a security list in Oracle Cloud Infrastructure and assign the security list to the appropriate subnets. To use custom security lists, you must assign the instance to a custom subnet, and not the default subnet.

Before you create service instances in Oracle Cloud Infrastructure that use your new network resources, you must create policies that grant your service access to these resources. See Prerequisites for Oracle Platform Services in the Oracle Cloud Infrastructure documentation.

3

Migrate an Oracle Java Cloud Service Instance to Oracle Cloud Infrastructure

Create a new Oracle Java Cloud Service instance in Oracle Cloud Infrastructure, and then use the Oracle WebLogic Server Deploy Tooling to migrate your Oracle WebLogic Server domain resources and applications from your existing instance in Oracle Cloud Infrastructure Classic.

When you migrate an Oracle Java Cloud Service instance, the following terms are used:

- *Source*: The Oracle Java Cloud Service instance in Oracle Cloud Infrastructure Classic.
- *Target*: The Oracle Java Cloud Service instance in Oracle Cloud Infrastructure.

> **Note:**
>
> The migration of a single Oracle Java Cloud Service instance takes approximately one day (8 hours). This does not include the time required to migrate any application databases.

Topics:

Perform Oracle Cloud Infrastructure Prerequisites

Before you create an Oracle Java Cloud Service instance in an Oracle Cloud Infrastructure region, you must create the required infrastructure and database resources.

1. Create the following Oracle Cloud Infrastructure resources if they don't already exist:
   - A compartment
   - A virtual cloud network (VCN) and at least one subnet
   - A storage bucket for backups
   - A user authentication token (auth token)
   - Policies that allow Oracle Java Cloud Service to access the resources in your compartment

   See Prerequisites for Oracle Platform Services in the Oracle Cloud Infrastructure documentation.
2. Create a database in Oracle Cloud Infrastructure Database if one doesn't already exist.

   Oracle Java Cloud Service will provision the required infrastructure schema to this database. See Managing Bare Metal and Virtual Machine DB Systems in the Oracle Cloud Infrastructure documentation.

Create the Target Instance

Create a new Oracle Java Cloud Service instance in an Oracle Cloud Infrastructure region. This instance must have the same topology and configuration as the source instance in Oracle Cloud Infrastructure Classic.

If your source and target instances are located in the same Oracle Cloud account, they cannot have identical instance names.

The domain, server, and cluster names in a service instance are derived from the first eight characters of the instance name. For example, the following instance names are different, but result in identical domain, server, and cluster names in Oracle WebLogic Server:

- **`MyJCSIns`**`tance`
- **`MyJCSIns`**`tanceOCI`

From the Oracle Java Cloud Service console:

1. Click **Create Instance**, and then select **Java**.
2. For **Instance Name**, enter a value so that the first eight characters are the same as those in your source instance name.

   For example, if the source instance is named `MyJCSInstance`, name the target instance `MyJCSInstanceOCI`.
3. Select an Oracle Cloud Infrastructure **Region**, **Availability Domain**, and **Subnet**.

4. For **Service Level** and **Software Edition**, select the same values as the source instance.
5. For **Software Release**, select the same major version (`X.Y`) as the source instance.

   For example, `12.2.1.2` and `12.2.1.3` are the same major version of Oracle WebLogic Server.
6. Click **Next**.
7. Click **Advanced**.
8. For **WebLogic Clusters**, create the same number of clusters as the source instance. Also set the cluster names and server counts to the same values as the source instance.

   For example, if the source instance has a single cluster named `cluster1` with a server count of `3`, then the target instance must have the same configuration.
9. For the **Compute Shape** of your WebLogic Cluster, select an Oracle Cloud Infrastructure shape that most closely matches the number of Oracle Compute Units (OCPUs) and the amount of memory that are available in the Oracle Cloud Infrastructure Classic shape in your source instance.

   See Select Oracle Cloud Infrastructure Shapes.
10. For **SSH Public Key**, upload an existing key or generate a new one.
11. For **Local Administrative User Name** and **Password**, enter the same Oracle WebLogic Server administrator credentials as your source instance.
12. If your source instance includes an Oracle Coherence data grid cluster, then select the same **Cluster Size** and **Managed Servers Per Node** as the data grid cluster in the source instance. Also select a **Compute Shape** for the data grid cluster that most closely matches the Oracle Cloud Infrastructure Classic shape.
13. For **Database Type**, select **Oracle Cloud Infrastructure Database**.
14. Select the **Compartment Name** where your Oracle Cloud Infrastructure Database resides.
15. For **Database Instance Name**, select the Oracle Cloud Infrastructure Database that you created for the Oracle Java Cloud Service infrastructure schema.

    Also enter a value for **PDB Name** if applicable.
16. Enter the **Password** for your database system administrator.
17. For **Backup Destination**, select **Both Remote and Disk Storage**.
18. For **Object Storage Container**, enter the URL of an existing bucket in Oracle Cloud Infrastructure Object Storage.
19. For **User Name**, enter the name of a cloud user that has access to the storage bucket.
20. For **Password**, enter the authentication token (auth token) that was generated for the cloud user.
21. Complete the instance creation wizard.

For more information, see Create an Oracle Java Cloud Service Instance Attached to a Public Subnet on Oracle Cloud in *Administering Oracle Java Cloud Service*.

Migrate the Application Databases

If the applications in your Oracle Java Cloud Service instance use database instances that were created in an Oracle Cloud Infrastructure Classic region, migrate these application databases to Oracle Cloud Infrastructure Database.

See Migrate a single-instance database from Oracle Database Cloud Service to a Virtual Machine DB System

1. Create the Oracle Cloud Infrastructure Database instances in the same region and virtual cloud network (VCN) as your target Oracle Java Cloud Service instance.
2. If the databases and target Oracle Java Cloud Service instance are on different subnets, then configure security rules that allow the service instance's subnet to communicate with the database ports.

Get Information About the Application Databases

Gather information about the Oracle Cloud Infrastructure Database instances that your target Oracle Java Cloud Service instance will use to access your application schemas. You will use this information to perform the migration.

1. Access the Oracle Cloud Infrastructure console.
2. Click the menu icon, and under **Database**, select **Bare Metal, VM, and Exadata**
3. Select the **Region** and **Compartment** where your database resides.
4. Click the name of your database.
5. From the DB System Details page, record these values.
   - The public IP address of the first database node
   - The host name prefix for the database (for example, `myappdb`)
   - The domain name for the database (for example, `mydbsubnet.myvcn.oraclevcn.com`)
   - The database port number
   - The database name and unique name (for example, `ORCL` and `ORCL_iad1zj`)
6. If your database is running Oracle Database 12c or later, then identify the pluggable database (PDB) that contains your application schemas.
   a. Use a Secure Shell (SSH) client to connect to the database node as the `opc` user.

      ```
      ssh -i <privatekey> opc@<database_IP>
      ```

   b. Switch to the `oracle` user.

      ```
      sudo su - oracle
      ```

   c. Locate the `ORACLE_HOME` directory for the database on the file system.

Example:

```
/u01/app/oracle/product/12.1.0.2/dbhome_1
```

d. If you are accessing this database node for the first time, run the `oraenv` command to configure the environment.

```
source oraenv
```

When prompted, enter the database name (SID) and the `ORACLE_HOME` directory.

Example:

```
ORACLE_SID = ORCL
ORACLE_HOME = /u01/app/oracle/product/12.1.0.2/dbhome_1
```

e. Start `sqlplus` as the `sysdba` role.

```
sqlplus / as sysdba
```

f. Print the list of PDBs in this database.

```
SELECT PDB, NETWORK_NAME, CON_ID FROM CDB_SERVICES;
```

g. In the command output, identify the PDB name.

Example:

```
MYPDB  mypdb.mydbsubnet.myvcn.oraclevcn.com
```

7. Repeat Steps 1 to 6 for any other application databases to which the target instance will connect.

Get Information About the Service Instances

Gather information about your source and target Oracle Java Cloud Service instances. You will use this information when you perform the migration.

1. Access the Oracle Java Cloud Service console.
2. Click the name of your *source* instance.
3. From the Overview page, record these values.
   - The public IP address of the first node that is running the Administration Server
   - The host names of all Managed Server nodes (for example, `myinstance-wls-2`)
   - The names of the Administration Server and all Managed Servers (for example, `MyInstan_server_1`)

4. Access the Oracle WebLogic Server administration console on the source instance.

   `https://<source_admin_IP>:7002/console`

   If you did not enable console access when you created the source instance, see Enable Console Access for a Service Instance in *Administering Oracle Java Cloud Service*.
5. After you sign in to the console, record the domain name (for example, `MyInstan_domain`).
6. From the **Domain Structure** panel, expand **Environment**, and then click **Clusters**.
7. Record the names of the clusters (for example, `MyInstan_cluster`).
8. From the **Domain Structure** panel, expand **Environment**, and then click **Machines**.
9. Record the names of the machines (for example, `MyInstan_machine_1`).
10. Return to the Instances page of the Oracle Java Cloud Service console.
11. Click the name of your *target* instance.
12. From the Overview page, record these values.
    - The public IP address of the first node that is running the Administration Server
    - The host names of all Managed Server nodes
    - The names of the Administration Server and all Managed Servers, if different from the source instance
13. Access the Oracle WebLogic Server administration console on the target instance.

    `https://<target_admin_IP>:7002/console`

14. After you sign in to the console, record the domain, cluster and machine names, if different from the source instance.

Create a Backup of the Target Instance

Before you update your target Oracle Java Cloud Service instance in Oracle Cloud Infrastructure, create a backup of the service instance.

If you encounter problems during or after the migration process, you can restore this backup and try again.

1. Access your service console.
2. Click the name of the service instance for which you want to create a backup.
3. On the Overview page, click the **Administration** tile.
4. Click the **Backup** tab.
5. Click **Manage backups for this instance**  , and then select **Backup Now**.

6. If you select **Keep Forever**, then this backup can only be deleted manually. If not selected, this backup will be deleted at the end of the current backup retention period for this service instance.

   If you select the **Include Database** option and if the Oracle Real Application Clusters (RAC) option is enabled on your database, then this option does not apply to the database backup. The database instance's retention policy determines how long the database backup is kept.

7. For **Notes**, enter up to 255 characters of text to provide additional information about the backup (for example, when to restore from this backup, why the backup was created, or the state of the service instance at the time of the backup).

8. Click **Back Up**.

9. To check the status of the backup operation, periodically click **Refresh**  .

Stop All Oracle WebLogic Server Processes on the Target Instance

Before you perform the migration on the target Oracle Java Cloud Service instance, you must stop all Oracle WebLogic Server and Node Manager processes.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *target* instance as the `opc` user.

   ```
   ssh -i <privatekey> opc@<target_admin_IP>
   ```

2. Switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. List the IDs of all running Java processes on this node.

   ```
   jps -l
   ```

4. Kill all `weblogic.Server` and `weblogic.NodeManager` processes.

   ```
   kill -9 <processID>
   ```

5. Use SSH to connect from the Administration Server node to the first Managed Server node in this instance.

   Example:

   ```
   ssh myinstance-wls-2
   ```

6. List the IDs of all running Java processes on this node.

   ```
   jps -l
   ```

7. Kill all `weblogic.Server` and `weblogic.NodeManager` processes.

   ```
   kill -9 <processID>
   ```

8. Disconnect from this Managed Server node.

   ```
   exit
   ```

9. Repeat Steps 5 to 8 for all remaining Managed Server nodes in this instance.

Install the Oracle WebLogic Server Deploy Tooling

Download and install the Oracle WebLogic Server Deploy Tooling to your source and target Oracle Java Cloud Service instances.

Oracle WebLogic Server Deploy Tooling is an open-source project. It provides scripts that enable you to discover and export the configuration and application files from one Oracle WebLogic Server domain, and then import the configuration and applications into another domain.

1. Download the latest `weblogic-deploy.zip` file from the Oracle WebLogic Server Deploy Tooling project on GitHub.
2. Use a Secure Copy (SCP) client to upload the file to the Administration Server node in your *source* instance.

   ```
   scp -i <privatekey> weblogic-deploy.zip opc@<source_admin_IP>:/tmp
   ```

3. Use a Secure Shell (SSH) client to connect to the node.

   ```
   ssh -i <privatekey> opc@<source_admin_IP>
   ```

4. Change the owner of the `weblogic-deploy.zip` file to the `oracle` user.

   ```
   sudo chown oracle:oracle /tmp/weblogic-deploy.zip
   ```

5. Switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

6. Extract `weblogic-deploy.zip` to `/u01`.

   ```
   unzip -d /u01 /tmp/weblogic-deploy.zip
   ```

7. Disconnect from the node.
8. Repeat Steps 2 to 7 for your *target* instance.

Discover the Oracle WebLogic Server Domain on the Source Instance

Run the Oracle WebLogic Server Deploy Tooling on your source Oracle Java Cloud Service instance to capture its domain configuration, applications and other supporting files.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *source* instance as the `opc` user.

   ```
   ssh -i <privatekey> opc@<source_admin_IP>
   ```

2. Switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. Determine the locations of the `DOMAIN_HOME` and `MIDDLEWARE_HOME` directories on the file system.

   ```
   echo $DOMAIN_HOME
   echo $MIDDLEWARE_HOME
   ```

4. Navigate to the `/u01/weblogic-deploy` directory.

   ```
   cd /u01/weblogic-deploy
   ```

5. Run the `discoverDomain.sh` command and specify the following parameters:
   - The locations of your `DOMAIN_HOME` and `MIDDLEWARE_HOME` directories
   - The names of the two output files (model and archive)
   - The `JRF` domain type

   > **Caution:**
   >
   > You must specify the `JRF` domain type, so that the tool ignores standard resources and applications that are found in all service instances.

   Format:

   ```
   /u01/weblogic-deploy/bin/discoverDomain.sh -domain_home /u01/data/
   domains/<source_domain> -oracle_home /u01/app/oracle/middleware/ -
   model_file <source_domain>.yaml -archive_file <source_domain>.zip -
   domain_type JRF
   ```

   Example:

   ```
   /u01/weblogic-deploy/bin/discoverDomain.sh -domain_home /u01/data/
   domains/MyInstan_domain -oracle_home /u01/app/oracle/middleware/ -
   ```

```
model_file MyInstan_domain.yaml -archive_file MyInstan_domain.zip -
domain_type JRF
```

6. Verify that the `discoverDomain.sh` command completed successfully with no errors.

   ```
   ####<timestamp> <INFO> <ValidationResults> <log_results> <WLSDPLY-05204>
   <Validation found 0 error, 0 warning, and 0 informational messages.>
   discoverDomain.sh completed successfully (exit code = 0)
   ```

   Ignore any warnings about the Trust Service Identity Asserter.

7. Copy the output files to `/tmp`.

   ```
   cp <source_domain>.* /tmp
   ```

8. Change the owner of the output files to the `opc` user.

   ```
   exit
   sudo chown opc:opc /tmp/<source_domain>.*
   ```

9. Disconnect from the node.

Edit the Domain Model and Copy It to the Target Instance

Oracle WebLogic Server Deploy Tooling exports a domain as a YAML file, which is referred to as the metadata model. Modify the YAML file so that it matches the configuration of your target Oracle Java Cloud Service instance.

For security purposes, Oracle WebLogic Server Deploy Tooling excludes the values of all password configuration attributes in the model file.

The domain model syntax allows you to externalize variables in a separate properties file. Oracle recommends that you use a separate file to configure the passwords that are required in your domain configuration, including data source and keystore passwords. To refer to a variable in the model file, use the **@@PROP:<*property_name*>@@** format .

1. Using a Secure Copy (SCP) client, download the model file and archive file from the Administration Server node in your *source* instance to your local computer.

   ```
   scp -i <privatekey> opc@<source_admin_IP>:/tmp/<source_domain>.* .
   ```

2. Create a backup copy of the model file.

   ```
   cp <source_domain>.yaml <source_domain>.yaml.bak
   ```

3. Open the `<source_domain>.yaml` model file in a text editor.
4. If necessary, find the names of all servers, clusters and machines in the model file, and replace them with the corresponding server, cluster and machine names of your target instance.

> **Note:**
>
> If the first eight characters of the source instance name are the same as the first eight characters in the target instance name, then this step is not required.

Example:

```
...
Cluster:
    MyTarget_cluster:
        ...
Server:
    MyTarget_adminserver:
        ...
        Machine: MyTarget_machine_1
    MyTarget_server_1:
        ...
        Machine: MyTarget_machine_1
        Cluster: MyTarget_cluster
        ServerTemplate: MyTarget_cluster_Template
        JTAMigratableTarget:
            Cluster: MyTarget_cluster
            UserPreferredServer: MyTarget_server_1
    ...
MigratableTarget:
    MyTarget_server_1 (migratable):
        Cluster: MyTarget_cluster
        UserPreferredServer: MyTarget_server_1
    ...
ServerTemplate:
    MyTarget_cluster_Template:
        Cluster: MyTarget_cluster
        JTAMigratableTarget:
            Cluster: MyTarget_cluster
    ...
UnixMachine:
    MyTarget_machine_1:
    ...
JDBCSystemResource:
    'MyDataSource':
        Target: MyTarget_cluster
    ...
CoherenceClusterSystemResource:
    DataGridConfig:
        Target: MyTarget_cluster
    ...
Application:
    MyApp:
        Target: MyTarget_cluster
```

5. Find and remove the following applications from the model file, if they exist:

- `OraJaaSmon`
- `sample-app`

In the following example, remove the highlighted lines.

```
Application:
    OraJaaSmon:
        SourcePath: wlsdeploy/applications/OraJaaSmon.war
        ModuleType: war
        StagingMode: nostage
        Target: MyTarget_adminserver
    'sample-app':
        SourcePath: 'wlsdeploy/applications/sample-app.war'
        ModuleType: war
        StagingMode: stage
        Target: MyTarget_cluster
```

6. Find and remove all occurrences of the following attributes from the model file:
   - `ListenAddress`
   - `NodeManagerPasswordEncrypted`
   - `CredentialEncrypted`
7. For each server in the model file, find and remove the `PublicAddress` attribute from the following default `NetworkAccessPoint` nodes:
   - `channel-dep`
   - `SecuredExternAdmin`
   - `ExternAdmin`
   - `SecuredExternContent`
   - `ExternContent`

   In the following example, the highlighted line should be removed.

```
Server:
    MyInstan_adminserver:
        ...
        NetworkAccessPoint:
            'channel-dep':
                ...
                PublicAddress: 203.0.113.10
```

8. Find the `PublicAddress` attribute of any custom `NetworkAccessPoint` nodes in the model file (not in the list above), and replace the current value with the corresponding public IP address that is assigned to your target instance.

   Example:

```
Server:
    MyInstan_adminserver:
        ...
        NetworkAccessPoint:
            MyChannel:
```

```
...
PublicAddress: <target_IP>
```

9. For each server in the model file, find the `Arguments` attribute within the `ServerStart` node:
   - If you configured any custom startup arguments for a server in your source instance, then replace the current value of `Arguments` with the custom arguments only.
   - If you did not configure any custom startup arguments for a server, then remove the entire `Arguments` line.

   In the following example, the server has custom startup arguments:

   ```
   MyInstan_server_1:
       ...
       ServerStart:
           Arguments: '-Dmy.custom.arg=true'
   ```

10. Create a file named `wdt.properties`.
11. If your source instance is configured to use custom identity and trust keystore files, then update the model file with the keystore passwords.
    a. Enter the required passwords for your keystores and private keys as properties in the `wdt.properties` file.

       Example:

       ```
       keystore1.password=<your_password>
       trustkeystore1.password=<your_password>
       privatekey1.password=<your_password>
       ```

    b. For each server in your model file, find the following attributes, and replace the current placeholder values with references to the corresponding properties:
       - `CustomIdentityKeyStorePassPhraseEncrypted`
       - `CustomTrustKeyStorePassPhraseEncrypted`

       Example:

       ```
       Server:
           MyInstan_server_1:
               ...
               CustomIdentityKeyStorePassPhraseEncrypted:
       '@@PROP:keystore1.password@@'
               CustomTrustKeyStorePassPhraseEncrypted:
       '@@PROP:trustkeystore1.password@@'
       ```

    c. For each server in your model file, find the `ServerPrivateKeyPassPhraseEncrypted` attribute in the `SSL` node, and then replace the current placeholder values with a reference to the corresponding property.

Example:

```
Server:
    MyInstan_server_1:
        ...
        SSL:
            ServerPrivateKeyPassPhraseEncrypted:
'@@PROP:privatekey1.password@@'
```

d. Add the following attributes to the `SSL` node for your administration server, if they are not already present:

- `Enabled: true`
- `ListenPort: 9072`

Example:

```
Server:
    MyInstan_adminserver:
        ...
        SSL:
            Enabled: true
            ListenPort: 9072
            ServerPrivateKeyPassPhraseEncrypted:
'@@PROP:privatekey1.password@@'
```

e. For each managed server in your model file, add the following attributes to the `SSL` node, if they are not already present:

- `Enabled: true`
- `ListenPort: 9074`

Example:

```
Server:
    MyInstan_server_1:
        ...
        SSL:
            Enabled: true
            ListenPort: 9074
            ServerPrivateKeyPassPhraseEncrypted:
'@@PROP:privatekey1.password@@'
```

12. If your source instance includes custom Java Database Connectivity (JDBC) data sources, then provide the location and password of the new application databases in Oracle Cloud Infrastructure.

a. Identify the data sources found within the `JDBCSystemResource` node in your model file.

b. Enter the required passwords for your data sources as properties in the `wdt.properties` file.

Example:

```
datasource1.password=<your_password>
datasource2.password=<your_password>
```

c. For each data source in your model file, find the `PasswordEncrypted` attribute and replace the current placeholder value with a reference to the corresponding property.

Example:

```
JDBCSystemResource:
    MyDataSource1:
        ...
        JdbcResource:
            ...
            JDBCDriverParams:
                URL: ...
                PasswordEncrypted: '@@PROP:datasource1.password@@'
```

d. For each data source in your model file, find the `URL` attribute and replace the current value with the URL to the corresponding Oracle Cloud Infrastructure Database.

The following table shows the URL format to use, depending on the Oracle Database version, and whether you created a Virtual Machine (VM) or Bare Metal database type.

Database Version	Database Type	URL Format
12c	VM	`jdbc:oracle:thin:@//<db_hostname>-scan.<db_domain>:<db_port>/<pdb_name>.<db_domain>`
12c	Bare Metal	`jdbc:oracle:thin:@//<db_hostname>.<db_domain>:<db_port>/<pdb_name>.<db_domain>`
11g	VM	`jdbc:oracle:thin:@//<db_hostname>-scan.<db_domain>:<db_port>/<db_unique_name>.<db_domain>`
11g	Bare Metal	`jdbc:oracle:thin:@//<db_hostname>.<db_domain>:<db_port>/<db_unique_name>.<db_domain>`

If you did not specify a PDB name when you created an Oracle Cloud Infrastructure Database that is running Oracle Database 12c, the default name is `<db_name>_pdb1`.

The following example shows a Virtual Machine database named `myappdb`, that is running Oracle Database 12c, and contains a PDB named `pdb1`:

```
JDBCDriverParams:
    URL: jdbc:oracle:thin:@//myappdb-
```

```
scan.mydbsubnet.myvcn.oraclevcn.com:1521/
pdb1.mydbsubnet.myvcn.oraclevcn.com
```

13. Within the `SecurityConfiguration` node in your model file, remove the `Adjudicator` node and any child nodes, if they exist.

    In the following example, the highlighted lines should be removed.

```
SecurityConfiguration:
    ...
    Realm:
        myrealm:
            Adjudicator:
                DefaultAdjudicator:
                    DefaultAdjudicator:
```

14. Use a Secure Copy (SCP) client to upload the model file, archive file, and properties file to the Administration Server node in your *target* instance.

```
scp -i <privatekey> <source_domain>.* opc@<target_admin_IP>:/tmp
scp -i <privatekey> wdt.properties opc@<target_admin_IP>:/tmp
```

Update the Oracle WebLogic Server Domain on the Target Instance

Run the Oracle WebLogic Server Deploy Tooling on your target Oracle Java Cloud Service instance to update its domain configuration and to deploy your applications.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *target* instance as the `opc` user.

```
ssh -i <privatekey> opc@<target_admin_IP>
```

2. Change the owner of the archive, model, and properties files to the `oracle` user.

```
sudo chown oracle:oracle /tmp/source_domain.*
sudo chown oracle:oracle /tmp/wdt.properties
```

3. Switch to the `oracle` user.

```
sudo su - oracle
```

4. Navigate to the `/u01/weblogic-deploy` directory.

```
cd /u01/weblogic-deploy
```

5. Copy the input files to the current directory.

```
cp /tmp/<source_domain>.* .
cp /tmp/wdt.properties .
```

6. Run the `validateModel.sh` command and specify the following parameters:

- The location of your MIDDLEWARE_HOME directory
- The names of the model, archive and properties files
- The JRF domain type

Format:

```
/u01/weblogic-deploy/bin/validateModel.sh -oracle_home /u01/app/oracle/
middleware/ -model_file <source_domain>.yaml -archive_file
<source_domain>.zip -variable_file wdt.properties -domain_type JRF
```

Example:

```
/u01/weblogic-deploy/bin/validateModel.sh -oracle_home /u01/app/oracle/
middleware/ -model_file MyInstan_domain.yaml -archive_file
MyInstan_domain.zip -variable_file wdt.properties -domain_type JRF
```

7. Verify that the validateModel.sh command completed successfully. Correct any errors.

```
####<timestamp> <INFO> <validate> <__perform_model_file_validation>
<WLSDPLY-05403>
<Validation of /u01/weblogic-deploy/<source_domain>.yaml completed with
0 error(s), 0 warning(s) and 0 info(s) items>
validateModel.sh completed successfully (exit code = 0)
```

8. Run the updateDomain.sh command and specify the following parameters:
   - The locations of your DOMAIN_HOME and MIDDLEWARE_HOME directories
   - The names of the model, archive, and properties files
   - The JRF domain type

   Format:

```
/u01/weblogic-deploy/bin/updateDomain.sh -domain_home /u01/data/domains/
<target_domain> -oracle_home /u01/app/oracle/middleware/ -model_file
<source_domain>.yaml -archive_file <source_domain>.zip -variable_file
wdt.properties -domain_type JRF
```

   Example:

```
/u01/weblogic-deploy/bin/updateDomain.sh -domain_home /u01/data/domains/
MyInstan_domain -oracle_home /u01/app/oracle/middleware/ -model_file
MyInstan_domain.yaml -archive_file MyInstan_domain.zip -variable_file
wdt.properties -domain_type JRF
```

9. Verify that the updateDomain.sh command completed successfully with no errors.

```
updateDomain.sh completed successfully (exit code = 0)
```

   Log files are in the /u01/weblogic-deploy/logs directory.

10. Disconnect from the Administration Server node.

Copy Supporting Files to the Target Instance

Identify and copy any files to your target Oracle Java Cloud Service instance that are not managed by Oracle WebLogic Server Deploy Tooling.

Oracle WebLogic Server Deploy Tooling automatically finds and archives the following types of files in your source instance's domain configuration. It also adds these files to your target instance's domain configuration:

- Application deployments
- Library deployments
- Custom keystores

Other files that your applications or domain resources require are not automatically managed by Oracle WebLogic Server Deploy Tooling, including files that are located outside the `DOMAIN_HOME` directory. You must manually copy these files to the target instance.

1. If the Managed Servers in your source instance are configured to use custom identity and trust keystore files, then copy the keystore files from the Administration Server node to the Managed Server nodes.

   Oracle WebLogic Server automatically stages application files to target Managed Server nodes, but does not do the same for keystore files.

   a. Use a Secure Shell (SSH) client to connect to the Administration Server node in your *target* instance.

      ```
      ssh -i <privatekey> opc@<target_admin_IP>
      ```

   b. Switch to the `oracle` user.

      ```
      sudo su - oracle
      ```

   c. Use SSH to connect from the Administration Server node to the host name of the Managed Server node.

      Example:

      ```
      ssh myinstance-wls-2
      ```

   d. Navigate to the `DOMAIN_HOME` directory.

      ```
      cd /u01/data/domains/<target_domain>
      ```

   e. Use a Secure Copy (SCP) client to download the archive file from the Administration Server node.

      Format:

      ```
      scp <target_admin_hostname>:/u01/weblogic-deploy/
      <source_domain>.zip .
      ```

Example:

```
scp myinstance-wls-1:/u01/weblogic-deploy/MyInstan_domain.zip .
```

f. Extract the archive file to the current directory.

```
unzip <source_domain>.zip
```

g. Disconnect from the Managed Server node.

h. Repeat Step 1 for any other Managed Servers that use custom keystores.

2. Use SSH to connect to the Administration Server node in your *source* instance.

```
ssh -i <privatekey> opc@<source_admin_IP>
```

3. Switch to the `oracle` user.

```
sudo su - oracle
```

4. Identify any supporting files that need to be copied to the target instance.

5. Copy the files to the `/tmp` directory.

Example:

```
cp /u01/myfiles/app.properties /tmp
```

> **Note:**
>
> If you have multiple files to transfer, then consider adding them to a single archive file.

6. Change the owner of the files to the `opc` user.

Example:

```
exit
sudo chown opc:opc /tmp/app.properties
```

7. Disconnect from the node.

8. Use SCP to download the files from the Administration Server node in your source instance to your local computer.

Example:

```
scp -i <privatekey> opc@<source_admin_IP>:/tmp/app.properties .
```

9. Use SCP to upload the files to the Administration Server node in your *target* instance.

Example:

```
scp -i <privatekey> app.properties opc@<target_admin_IP>:/tmp
```

10. Use SSH to connect to the Administration Server node in your target instance.

    ```
    ssh -i <privatekey> opc@<target_admin_IP>
    ```

11. Change the owner of the files to the `oracle` user.

    Example:

    ```
    sudo chown oracle:oracle /tmp/app.properties
    ```

12. Switch to the `oracle` user.

    ```
    sudo su - oracle
    ```

13. Move the files to the same location that they were found on the source instance.

    Example:

    ```
    mkdir /u01/myfiles
    mv /tmp/app.properties /u01/myfiles
    ```

14. Disconnect from the node.

Configure Node Manager SSL on the Target Instance

If you configured your source Oracle Java Cloud Service instance to use custom identity or trust keystores, then you must manually configure the Node Manager on each node in the target instance to use the custom keystores.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *target* instance as the `opc` user.

    ```
    ssh -i <privatekey> opc@<target_admin_IP>
    ```

2. Switch to the `oracle` user.

    ```
    sudo su - oracle
    ```

3. Using the model file and properties file, identify the SSL configuration for the servers on this node:
    - The identity keystore file, path, and password
    - The trust keystore file, path, and password
    - The key alias and password

    Example:

    ```
    Server:
        ...
        MyInstan_server_1:
            ...
            CustomIdentityKeyStoreFileName: wlsdeploy/servers/
    MyInstan_server_1/identity.jks
            CustomTrustKeyStoreFileName: wlsdeploy/servers/
    ```

```
MyInstan_server_1/trust.jks
        CustomIdentityKeyStorePassPhraseEncrypted:
'@@PROP:keystore1.password@@'
        CustomTrustKeyStorePassPhraseEncrypted:
'@@PROP:trustkeystore1.password@@'
        ...
        SSL:
            ServerPrivateKeyAlias: server_cert
            ServerPrivateKeyPassPhraseEncrypted:
'@@PROP:privatekey1.password@@'
```

4. Edit the `nodemanager.properties` file located under the `DOMAIN_HOME` directory.

   ```
   vi $DOMAIN_HOME/nodemanager/nodemanager.properties
   ```

5. Add the following lines to the end of the file. Specify the full path to the keystore files.

   ```
   KeyStores=CustomIdentityAndCustomTrust
   CustomIdentityKeystoreType=jks
   CustomIdentityKeyStoreFileName=/u01/data/domains/<target_domain>/
   wlsdeploy/servers/<target_server_name>/<identity_keystore_file>
   CustomIdentityKeyStorePassPhrase=<identity_keystore_password>
   CustomIdentityPrivateKeyPassPhrase=<key_password>
   CustomIdentityAlias=<key_alias>
   CustomTrustKeystoreType=jks
   CustomTrustKeyStoreFileName=/u01/data/domains/<target_domain>/wlsdeploy/
   servers/<target_server_name>/<trust_keystore_file>
   CustomTrustKeyStorePassPhrase=<trust_keystore_password>
   ```

   Example:

   ```
   KeyStores=CustomIdentityAndCustomTrust
   CustomIdentityKeystoreType=jks
   CustomIdentityKeyStoreFileName=/u01/data/domains/MyInstan/wlsdeploy/
   servers/MyInstan_adminserver/myidentity.jks
   CustomIdentityKeyStorePassPhrase=<identity_keystore_password>
   CustomIdentityPrivateKeyPassPhrase=<key_password>
   CustomIdentityAlias=server_cert
   CustomTrustKeystoreType=jks
   CustomTrustKeyStoreFileName=/u01/data/domains/MyInstan/wlsdeploy/
   servers/MyInstan_adminserver/mytrust.jks
   CustomTrustKeyStorePassPhrase=<trust_keystore_password>
   ```

6. Edit the `setDomainEnv.sh` file located under the `DOMAIN_HOME` directory.

   ```
   vi $DOMAIN_HOME/bin/setDomainEnv.sh
   ```

7. Add the following line to the end of the file.

   ```
   export WLST_PROPERTIES="${WLST_PROPERTIES} -
   Dweblogic.ssl.JSSEEnabled=true -Dweblogic.security.SSL.enableJSSE=true -
   Dweblogic.security.SSL.ignoreHostnameVerification=true -
   Dweblogic.security.TrustKeyStore=CustomTrust -
   ```

```
Dweblogic.security.CustomTrustKeyStoreFileName=/u01/data/domains/
<target_domain>/wlsdeploy/servers/<target_server_name>/
<trust_keystore_file> -Dweblogic.security.CustomTrustKeyStoreType=JKS"
```

Example:

```
export WLST_PROPERTIES="${WLST_PROPERTIES} -
Dweblogic.ssl.JSSEEnabled=true -Dweblogic.security.SSL.enableJSSE=true -
Dweblogic.security.SSL.ignoreHostnameVerification=true -
Dweblogic.security.TrustKeyStore=CustomTrust -
Dweblogic.security.CustomTrustKeyStoreFileName=/u01/data/domains/
MyInstan/wlsdeploy/servers/MyInstan_adminserver/mytrust.jks -
Dweblogic.security.CustomTrustKeyStoreType=JKS"
```

8. Connect to all Managed Server nodes in the target instance that use custom keystores, and then repeat Steps 4 to 7.

   Example:

   ```
   ssh myinstance-wls-2
   vi $DOMAIN_HOME/nodemanager/nodemanager.properties
   vi $DOMAIN_HOME/bin/setDomainEnv.sh
   exit
   ```

9. Disconnect from the Administration Server node.

Start All Oracle WebLogic Server Processes on the Target Instance

After you update the domain configuration on the target Oracle Java Cloud Service instance, you must restart all Oracle WebLogic Server and Node Manager processes.

The Administration Server must be running before you start any Managed Servers.

- Start the Administration Server on the first node and verify that it started successfully.
- Start the Managed Servers on all nodes.

If you previously shut down the server processes by using the `kill` command, then Node Manager restarts them for you automatically. Otherwise, you must start the server processes manually.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *target* instance as the `opc` user.

   ```
   ssh -i <privatekey> opc@<target_admin_IP>
   ```

2. Switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. Start the Node Manager process on this node.

- If your service instance is running Oracle WebLogic Server 12c, then run the following command.

  ```
  /u01/data/domains/<target_domain>/bin/startNodeManager.sh >nm.out
  2>&1 &
  ```

- If your service instance is running Oracle WebLogic Server 11g, then run the following command.

  ```
  /u01/app/oracle/middleware/wlserver_10.3/server/bin/
  startNodeManager.sh >nm.out 2>&1 &
  ```

4. List the IDs of all running Java processes on this node.

   ```
   jps -l
   ```

5. Verify that the Node Manager and server processes are running.

   - `weblogic.NodeManager`
   - `weblogic.Server`

   The first node on your service instance has two `weblogic.Server` processes: the Administration Server and the first Managed Server.

6. If Node Manager started, but the servers did not, then you must start the servers manually.

   a. Launch the WebLogic Scripting Tool (WLST).

      ```
      source $DOMAIN_HOME/bin/setDomainEnv.sh
      $MIDDLEWARE_HOME/oracle_common/common/bin/wlst.sh
      ```

   b. Connect to the Node Manager on this node.

      ```
      nmConnect(username="<nm_user>", password="<nm_password>",
      domainName="<target_domain>", domainDir="/u01/data/domains/
      <target_domain>", nmType="ssl", host="<target_hostname>",
      port="5556", verbose="false")
      ```

      Example:

      ```
      nmConnect(username="weblogic", password="<nm_password>",
      domainName="MyInstan_domain", domainDir="/u01/data/domains/
      MyInstan_domain", nmType="ssl", host="myinstance-wls-1",
      port="5556", verbose="false")
      ```

      By default, the Node Manager credentials are the same as those you specified when you created the target instance.

   c. Restart the servers on this node.

      ```
      nmStart('<target_server>')
      nmServerStatus('<target_server>')
      ```

Both the Administration Server and the first Managed Server run on the first node in your service instance. The Administration Server must be running before you start any Managed Servers.

Example:

```
nmStart('MyInstan_adminserver')
nmServerStatus('MyInstan_adminserver')
nmStart('MyInstan_server_1')
nmServerStatus('MyInstan_server_1')
```

d. Exit WLST.

```
exit()
```

7. Use SSH to connect from the Administration Server node to the host name of each Managed Server node in this instance, and then repeat Steps 3 to 6.

   Example:

   ```
   ssh myinstance-wls-2
   ```

8. Disconnect from the Administration Server node.
9. Verify that you can sign in to the Oracle WebLogic Server Administration Console.

   ```
   https://<target_admin_IP>:7002/console
   ```

Create the Trust Service Identity Asserter on the Target Instance

After you update the domain configuration on the target Oracle Java Cloud Service instance, you must manually create the Trust Service Identity Assertion provider in the Oracle WebLogic Server security realm.

The Oracle WebLogic Server Deploy Tooling does not automatically migrate the Trust Service Identity Assertion provider from your source instance to your target instance. Create the provider if it is missing from your target instance.

1. Access the Oracle WebLogic Server Administration Console for your *target* instance.

   ```
   https://<target_admin_IP>:7002/console
   ```

2. Sign in to the console as the Oracle WebLogic Server system administrator.
3. From the Change Center panel, click **Lock & Edit**.
4. From the Domain Structure panel, click **Security Realms**.
5. Click **myrealm**.
6. Click the **Providers** tab.
7. Click the **Authentication** tab if it's not already selected.
8. Click **New**.

9. For **Name**, enter `Trust Service Identity Asserter`.
10. For **Type**, select **TrustServiceIdentityAsserter**.
11. Click **OK**.
12. Click **Reorder**.
13. Select **Trust Service Identity Asserter**, and then click **Move selected items to top of list**.
14. Click **OK**.
15. Click **Activate Changes**.

Troubleshoot Migration Problems

If you encounter problems migrating your Oracle Java Cloud Service instance to Oracle Cloud Infrastructure, inspect the log files for the migration tools and servers. After correcting the problems, you can restore the target instance to its initial state, and then try the migration again.

1. Use a Secure Shell (SSH) client to connect to the Administration Server node on the *target* instance as the `opc` user.

   ```
   ssh -i <privatekey> opc@<target_admin_IP>
   ```

2. Switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. Check for warnings or errors in the Oracle WebLogic Server Deploy Tooling log files, which are located in the `/u01/weblogic-deploy/logs` directory.
4. Check for warnings or errors in the Oracle WebLogic Server domain log file, which is located at `/u01/data/domains/<target_domain>/servers/target_server/logs/<target_domain>.log`.
5. Fix any problems that you identify.

   For example, edit the `<source_domain>.yaml` model file.
6. Access the Oracle Java Cloud Service console.
7. Click the name of the service instance that you want to restore.
8. On the Overview page, click the **Administration** tile.
9. Click the **Backup** tab.
10. Under **Available Backups**, beside the backup that you want to restore, click **Menu**  , and then select **Restore**.
11. For **Notes**, enter any free-form text to provide additional information about the restoration. For example, describe why you are restoring the service instance.
12. Click **Restore**.
13. When prompted for confirmation, perform one of the following steps:

- If the selected backup has an associated database backup, select the check box to confirm that you have already restored the database, and then click **Continue with Restore**.
- Click **Yes, Restore Service**.

14. To check the status of the restore operation, periodically click **Refresh** .
15. Perform these tasks again.
    - Stop All Oracle WebLogic Server Processes on the Target Instance
    - Install the Oracle WebLogic Server Deploy Tooling (target instance only)
    - Update the Oracle WebLogic Server Domain on the Target Instance
    - Start All Oracle WebLogic Server Processes on the Target Instance
    - Create the Trust Service Identity Asserter on the Target Instance

4

Complete the Post-Migration Tasks

After successfully migrating your Oracle Java Cloud Service instances from Oracle Cloud Infrastructure Classic to Oracle Cloud Infrastructure, test your applications thoroughly, and then perform cleanup and other optional configuration tasks.

Topics:

Test the Target Instance

Verify that your Java applications and other Oracle WebLogic Server resources are accessible and function correctly on the target Oracle Java Cloud Service instance.

Be sure to thoroughly run all application test cases. Also verify that you can access the WebLogic Server Administration Console.

If your instance includes custom data sources that access your application databases, you can test database connectivity directly from the WebLogic Server Administration Console. Select a data source, click the **Monitoring** tab, and then click the **Testing** tab.

Migrate FastConnect and VPN Connections to Oracle Cloud Infrastructure

Use Oracle Cloud Infrastructure to create a connection between your private, on-premises network and a network in Oracle Cloud.

A Virtual Private Network (VPN) uses a public network to create a secure connection between two private networks. Oracle supports two connectivity solutions for a Virtual Cloud Network (VCN) in Oracle Cloud Infrastructure:

- Oracle Cloud Infrastructure FastConnect - Create dedicated, high-speed, virtual circuits for production systems that communicate with your on-premises network using the Border Gateway Protocol (BGP). This service is equivalent to Oracle Cloud Infrastructure FastConnect Classic.
- IPSec VPN - Create secure connections with your on-premises network using the IPSec protocol. This solution replaces VPN as a Service (VPNaaS) and Corente in Oracle Cloud Infrastructure Classic.

When migrating from Oracle Cloud Infrastructure Classic, update the existing BGP or VPN configuration in your on-premises network to use either Oracle Cloud Infrastructure FastConnect or IPSec VPN. Alternatively, if you require connectivity to instances in both Oracle Cloud Infrastructure and Oracle Cloud Infrastructure Classic

during the migration process, create a separate BGP or VPN configuration in your on-premises network.

In Oracle Cloud Infrastructure, creating a connection to your on-premises network includes these tasks:

- Create a Dynamic Routing Gateway (DRG) in the VCN.
- Create a route table in the VCN that directs external traffic to the DRG.
- Assign the route table to a subnet in the VCN.

Refer to these topics in the Oracle Cloud Infrastructure documentation:

- FastConnect
- IPSec VPN

Clean Up Resources in Oracle Cloud Infrastructure Classic

After testing your target Oracle Java Cloud Service instance, you can delete the source instance and supporting cloud resources in Oracle Cloud Infrastructure Classic.

Delete these Oracle Cloud Infrastructure Classic resources to avoid costs for services that you no longer use.

1. Access the Oracle Java Cloud Service console.
2. Delete the source Oracle Java Cloud Service instances that you created in Oracle Cloud Infrastructure Classic.
    a. Click **Manage this instance** for the service instance, and then select **Delete**.
    b. Enter the **Database Administrator User Name** and **Database Administrator User Password** for the infrastructure schema database.

       Alternatively, select **Force Delete** if you plan to delete this database as well.
    c. Click **Delete**.
3. Click **IP Reservations**.
4. Delete any IP reservations that you created for your source Oracle Java Cloud Service instances.
    a. Click **Delete** for the IP reservation.
    b. When prompted for confirmation, click **OK**.
5. Access the Oracle Database Cloud Service console (Database Classic).
6. Delete the Oracle Database Cloud Service instances that you created in Oracle Cloud Infrastructure Classic to support your source Oracle Java Cloud Service instances.

   Do not delete a database if it is still in use by other services.
    a. Click **Manage this instance** for the database instance, and then select **Delete**.
    b. When prompted for confirmation, click **Delete**.
7. Click **IP Reservations**.

8. Delete any IP reservations that you created for your Oracle Database Cloud Service instances.

    a. Click **Delete**  for the IP reservation.

    b. When prompted for confirmation, click **OK**.

9. Access the Oracle Cloud Infrastructure Object Storage Classic console (Storage Classic).

10. Delete the object storage containers that you created in Oracle Cloud Infrastructure Classic to support your source Oracle Java Cloud Service instances.

    Do not delete a container if it is still in use by other services.

    a. Click the delete icon  for the container.

    b. When prompted for confirmation, click **OK**.