Oracle® Cloud

Migrating Oracle Java Cloud Service on OCI-C to Oracle WebLogic Server for OCI Using tar

F86789-04
March 2024

Oracle Cloud Migrating Oracle Java Cloud Service on OCI-C to Oracle WebLogic Server for OCI Using tar,

F86789-04

Primary Author: Oracle Corporation

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Contents

Preface

1 About Migrating Oracle Java Cloud Service

2 Prepare to Migrate

3 Migrate an Instance

Preface

Migrating Oracle Java Cloud Service Instances to Oracle WebLogic Server for OCI Using the Tape Archive File Format (TAR) explains how to migrate your existing Oracle Java Cloud Service instances to Oracle WebLogic Server for Oracle Cloud Infrastructure (Oracle WebLogic Server for OCI), if your Java Cloud Service instance uses Oracle Cloud Infrastructure Classic (OCI-C).

Topics:

- Documentation Accessibility
- Diversity and Inclusion

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1 About Migrating Oracle Java Cloud Service

These topics help you learn about the benefits to migrating your existing Oracle Java Cloud Service instances to Oracle WebLogic Server for Oracle Cloud Infrastructure (Oracle WebLogic Server for OCI), and also provide an overview of the migration process and tools.

> **Note:**
>
> Your Oracle Java Cloud Service instances may be running on Oracle Cloud Infrastructure (OCI) or Oracle Cloud Infrastructure Classic (OCI-C). To determine if your instances are on OCI or OCI-C, see the Oracle Support note 2984700.1.

Topics:

- Why Migrate to Oracle WebLogic Server for OCI
- Migration Scope
- About Oracle WebLogic Server for OCI
- Compare Oracle Java Cloud Service to Oracle WebLogic Server for OCI
- Migrate Using tar

Why Migrate to Oracle WebLogic Server for OCI

Oracle encourages you to migrate your existing cloud resources to Oracle WebLogic Server for OCI.

Oracle Cloud Infrastructure is Oracle's modern cloud platform that's based on the latest cloud technologies and standards. It provides more consistent performance and better features at lower costs. 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.

Oracle WebLogic Server for OCI allows you to quickly provision a WebLogic domain in Oracle Cloud Infrastructure and at the same time provides you the flexibility to customize your WebLogic domain. Following are some of the advantages to migrate to Oracle WebLogic Server for OCI:

- Supports WebLogic Server major release versions after 12.2.1.4.0.
- Enables efficient control of Oracle Cloud Infrastructure resources.
- Offers flexibility to add or remove patches.
- Allows the management of the operating system and domain after provisioning, without any restrictions.
- Allows the use of any supported method to backup, scale, or patch a domain.
- Allows the selection of the compartment where you want to create the resources.

- Allows full customization of the Oracle-managed load balancer.
- Supports cloning instances.
- Includes optional autoscaling feature.
- Supports a documented disaster recovery process. See Oracle WebLogic Server for Oracle Cloud Infrastructure Disaster Recovery.
- Allows WebLogic Server domains without Java Required Files (JRF).
- Integrates with multiple Oracle Cloud Infrastructure services to monitor, trace, deploy, scale and autoscale your WebLogic domain, and other Oracle Cloud Infrastructure services to manage and patch your WebLogic domain resources.
- Offers the option to pay per OCPU/Hour for Oracle WebLogic Server for OCI images with the entitlement to install any Oracle WebLogic Server version.

Migration Scope

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

Currently, Oracle does *not* support the migration of Oracle Java Cloud Service instances running Oracle WebLogic Server 11g or 12.1.3 releases.

If all the following prerequisites are met, then proceed with the migration process described in this guide:

- The Oracle Java Cloud Service uses Oracle Database Cloud Service (DBCS).
- The version of the Oracle Java Cloud Service is 12.2.1.x or later.

This guide does not include detailed procedures on the configuration of basic Oracle WebLogic Server for OCI security, network, and storage resources that might be required to support your new WebLogic Server domain. Instead, this guide provides references to the Oracle WebLogic Server for OCI documentation as appropriate.

> **Note:**
>
> The instructions provided in this document apply to any Oracle Java Cloud Service instance that uses DBCS. Although the instructions in this guide apply to migrating your existing Oracle Java Cloud Service instance in Oracle Cloud Infrastructure Classic to a new instance in Oracle WebLogic Server for OCI, you may use these instructions for older instances of OCI that use DBCS.

About Oracle WebLogic Server for OCI

Oracle WebLogic Server for OCI is available in the Oracle Cloud Infrastructure Marketplace as a set of applications. After launching one of these applications, you use a simple wizard interface to configure and provision an Oracle WebLogic Server domain along with supporting cloud resources such as compute instances, networks, and load balancers.

After launching a domain using the Marketplace applications, you track and monitor its progress as a stack using Resource Manager in Oracle WebLogic Server for OCI. A stack also provides a convenient method of deleting the cloud resources for a domain when you no longer require them.

Like Oracle Java Cloud Service, you can administer the domain and deploy Java EE applications to it just like on-premises domains. Use standard Oracle WebLogic Server tools such as the administration console, WebLogic Deploy Tool (WDT), and WebLogic Scripting Tool (WLST). You can also administer the operating system on the compute instances using a secure shell (SSH) client and standard Linux tools.

See About the Components of Oracle WebLogic Server for Oracle Cloud Infrastructure in *Using Oracle WebLogic Server for OCI*.

Compare Oracle Java Cloud Service to Oracle WebLogic Server for OCI

Understand the basic Oracle WebLogic Server for OCI security, network, and storage concepts, and compare them to their equivalent concepts in Oracle Java Cloud Service.

The following table compares the functionality of Oracle Java Cloud Service to Oracle WebLogic Server for OCI:

Oracle Java Cloud Service	Oracle WebLogic Server for OCI
Supports Oracle WebLogic Server 12.2.1.3, and 12.2.1.4.	Supports Oracle WebLogic Server 12.2.1.3, 12.2.1.4, and 14.1.1.0.
Will not support major version new releases of Oracle WebLogic Server.	Will support major version new releases of Oracle WebLogic Server.
All domains include the Java Required Files (JRF) components and require a database.	Creates basic and JRF-enabled WebLogic Server 12c domains.
Must use Oracle Java Cloud Service to backup, scale, or patch a domain. Certain changes to the operation system and domain are not supported (see Administration Best Practices).	Can use any supported method to backup, scale, or patch a domain; the documentation provides recommendations and best practices. No restrictions on managing the operating system or domain after provisioning.
Can provision an Oracle-managed load balancer in Oracle Cloud Infrastructure, or a user-managed load balancer running Oracle Traffic Director.	Can provision an Oracle-managed load balancer in Oracle Cloud Infrastructure.
Supports limited customization of the Oracle-managed load balancer.	Supports full customization of the Oracle-managed load balancer.
Uses Oracle Identity Cloud Service for authentication. A security application is created in Oracle Identity Cloud Service for each domain.	Use Oracle Identity Cloud Service for authentication. Must create a confidential application in Oracle Identity Cloud Service prior to creating a domain. Confidential application, enterprise application, and App Gateway are created in Oracle Identity Cloud Service for each domain.
Supports domain name as `<domain>.<tenancy>.jcs.ocp.oraclecloud.com` and a certificate for `*.jcs.ocp.oraclecloud.com`.	Does not get domain name and only a self-sign certificate.

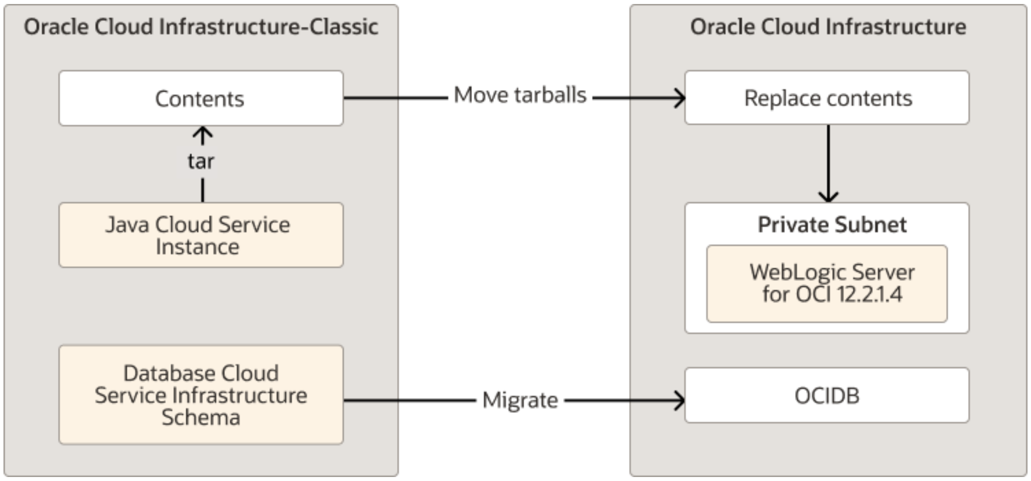
Oracle Java Cloud Service	Oracle WebLogic Server for OCI
Provides root certificates and DNS domain names for the load balancer.	Not supported.

Migrate Using tar

Use the procedures in this guide to migrate service instances to Oracle WebLogic Server for OCI by using `tar`.

The following diagram shows the migration topology for a Oracle Java Cloud Service instance. The migration target is a domain created with Oracle WebLogic Server for OCI.

Figure 1-1 Migration Topology


Oracle Cloud Infrastructure-Classic
Contents
tar
Java Cloud Service Instance
Database Cloud Service Infrastructure Schema
Move tarballs
Migrate
Oracle Cloud Infrastructure
Replace contents
Private Subnet
WebLogic Server for OCI 12.2.1.4
OCIDB

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

1. Migrate the Oracle Database Cloud Service system to an OCI Database System that preferably uses Logical Volume Manager (LVM) for storage.
2. Create a Oracle WebLogic Server for OCI 12.2.1.4 instance in a private subnet.
3. Tar the following contents from the Oracle Java Cloud Service instance:
    - `/u01/app/oracle/middleware`
    - `/u01/data/domains`
    - `/u01/jdk`
4. Move the tarballs to the Oracle WebLogic Server for OCI compute instances.
5. Replace the required contents on the Oracle WebLogic Server for OCI compute instances.
6. Replace the host names and database connection strings in the domain with values that point to the Oracle WebLogic Server for OCI host names and OCI database, respectively.

7. If you use Oracle Identity Cloud Service (IDCS):
   - Replace the client id and secret in the IDCS Authentication Provider.
   - Set up the protected paths in the IDCS application gateway.

2 Prepare to Migrate

Before you migrate your service instances to Oracle WebLogic Server for OCI, understand how the migration affects your existing instances, identify the necessary compute shapes, and create the network and databases to support your migrated service instances.

Topics:

About Downtime Requirements

Two JRF instances must not be running at the same time. You can limit the downtime by not stopping the WebLogic servers on your Oracle Java Cloud Service instance until you are ready to start the WebLogic servers on the Oracle WebLogic Server for OCI instance.

You can discover and export the configuration from your source Oracle WebLogic Server domain while it is running. The migration process 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 WebLogic Server for OCI.

Select Oracle WebLogic Server for OCI Shapes

Identify the compute shapes that provide similar IaaS resources in Oracle WebLogic Server for OCI to the shapes that you're currently using for your service instances in Oracle Java Cloud Service.

A compute shape defines the IaaS resources, such as OCPUs and memory, that are available to a specific node in a service instance.

To ensure that a migrated service instance has the same performance characteristics as the original instance, and can support an equivalent workload, choose Oracle WebLogic Server for OCI shapes that most closely map to the Oracle Java Cloud Service 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 WebLogic Server for OCI 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.

For information on mapping the OCPU count for your OCI-C instance shapes to the OCPU count you will select for the Flex shape used in your Oracle WebLogic Server for OCI instance, see Cloud Classic Pricing.

Configure Security Rules for the Network

If your instance communicates with other resources in Oracle Cloud or on the Internet, create or update the security lists for your target Oracle Cloud Infrastructure network.

A security list is assigned to subnets within your virtual cloud network (VCN). It includes ingress and egress rules that specify the types of traffic allowed in and out of the instances within the subnet. You can update an existing security list, or create a new one and assign it to a subnet.

You might need to create security rules if your Oracle WebLogic Server for OCI instance communicates with external resources, including these Oracle WebLogic Server resources:

- JavaMail Session
- Foreign Java Naming and Directory Interface (JNDI) Provider
- Foreign Java Message Service (JMS) Server
- Messaging Bridge
- Store-and-Forward
- WebLogic Diagnostic Framework (WLDF) REST Action

See Security Lists in the Oracle Cloud Infrastructure documentation.

Oracle recommends that you create a private subnet for Oracle WebLogic Server for OCI in the VCN used by Oracle Java Cloud Service.

Set up a private subnet that can be used for creating your Oracle WebLogic Server for OCI instance in the same VCN as the Oracle Java Cloud Service instance. This allows you to quickly set up a connection to the same database that Oracle Java Cloud Service uses and allow SSH connectivity between the Oracle WebLogic Server for OCI instance VMs and the Oracle Java Cloud Service VMs, so files can be transferred between them.

Recommended Simplified Network Configurations

The following configurations show a Oracle Java Cloud Service set up with a public subnet since this is the most common configuration.

> **Note:**
>
> Public subnets are allowed for Oracle WebLogic Server for OCI instances, but not recommended.

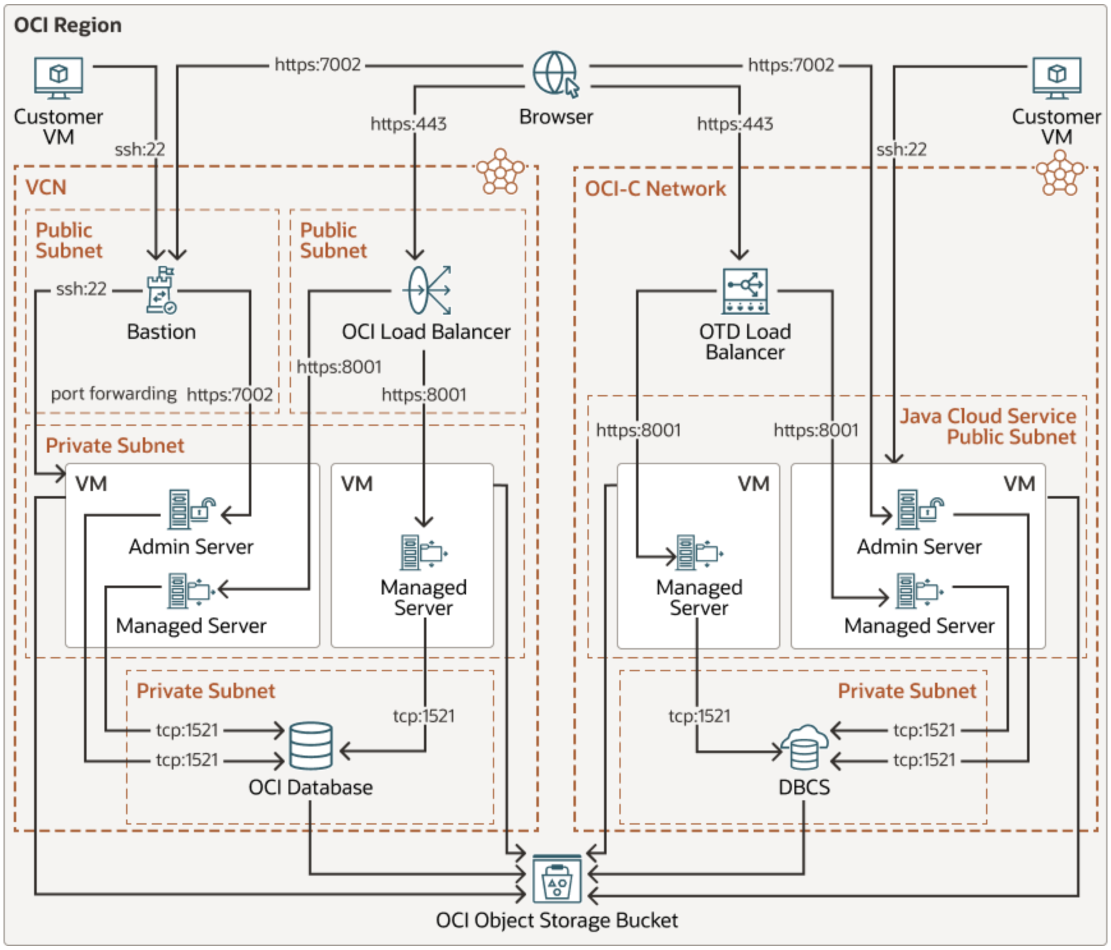
- Oracle Java Cloud Service with OTD
- Oracle Java Cloud Service with IDCS

Oracle Java Cloud Service with OTD

There is no Oracle Traffic Director (OTD) option in Oracle WebLogic Server for OCI. Instead, the OCI load balancer must be used to route traffic to the WebLogic managed servers.

The following diagram illustrates an Oracle Java Cloud Service instance using OTD and the Oracle WebLogic Server for OCI instance recommended to replace it. This illustration represents the state of both instances during the migration process, prior to deleting the Oracle Java Cloud Service instance.

Figure 2-1 Oracle Java Cloud Service with OTD



The private subnet for the OCI database and the public subnet for JCS must already exist. Additionally, you should create the following subnets:

- Oracle WebLogic Server for OCI private subnet
- OCI load balancer public subnet
- Bastion host public subnet

Use the following tables to configure the security lists for the these three subnets:

Table 2-1 Oracle WebLogic Server for OCI Private Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	Bastion public subnet, TCP	22	SSH access
Stateful Ingress	Bastion public subnet, TCP	7002	Administration server SSL port
Stateful Ingress	OCI load balancer public subnet, TCP	8001	Managed server port
Stateful Ingress	Oracle Java Cloud Service public subnet, TCP	22	SSH access, so domain and binaries can be copied over
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	22	SSH access between the compute instances of Oracle WebLogic Server for OCI
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	9071	Used for provisioning and scaling
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	9073	Used for WebLogic health check
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	5556	Used for accessing node manager

Table 2-2 OCI Load Balancer Public Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	Internet (0.0.0.0/0), TCP	443	SSL access

Table 2-3 Bastion Public Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	Hosts with access to bastion, TCP	22	SSH access, port forwarding

Use the following table to add access for the Oracle WebLogic Server for OCI private subnet to the OCI database public subnet:

Table 2-4 Access to OCI Database Public Subnet

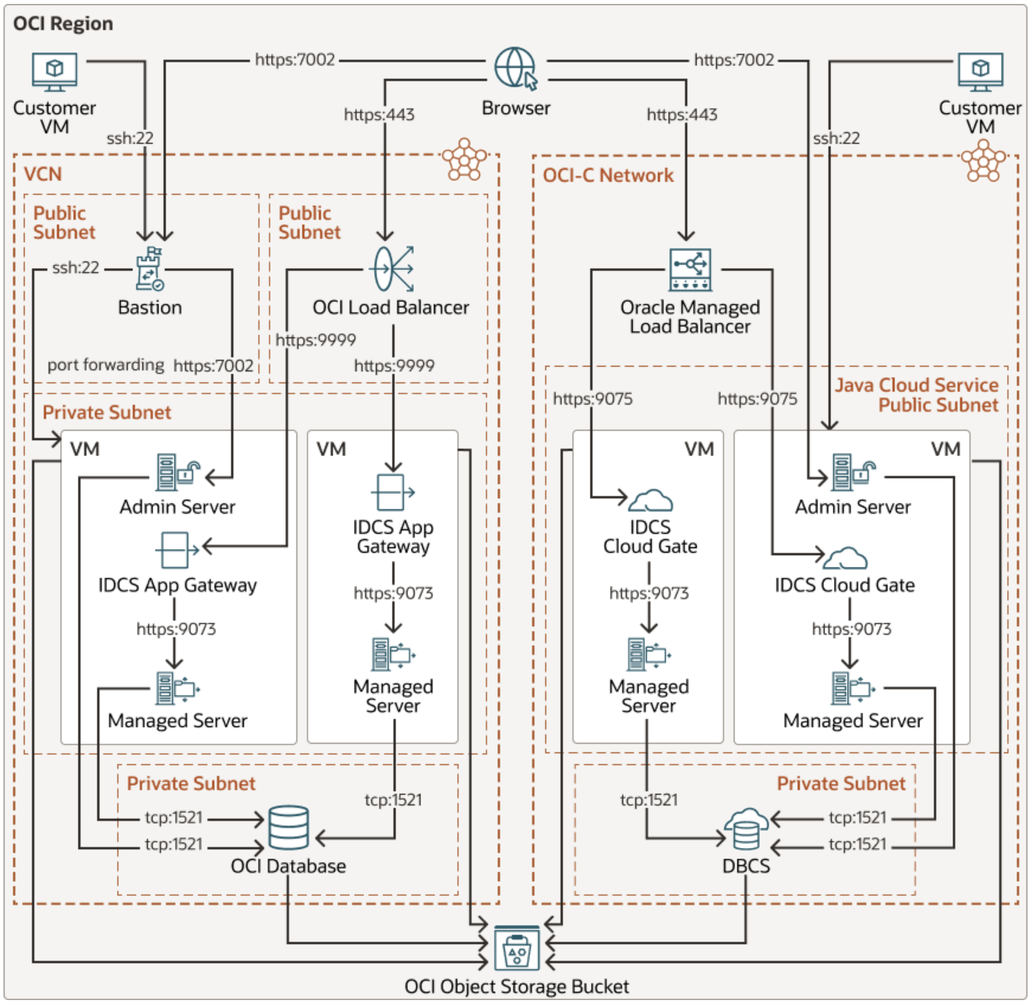
Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	WLS for OCI private subnet, TCP	1521	DB access

Oracle Java Cloud Service with IDCS

> **Note:**
>
> It is not possible to use the same IDCS application that is used in Oracle Java Cloud Service for Oracle WebLogic Server for OCI, because Oracle WebLogic Server for OCI uses an IDCS app gateway. In the same way, you cannot use the same load balancer that is used in Oracle Java Cloud Service for Oracle WebLogic Server for OCI because the load balancer in Oracle Java Cloud Service is managed by Oracle while the resources in Oracle WebLogic Server for OCI are managed by the user.

The following diagram illustrates a Oracle Java Cloud Service instance using IDCS and the Oracle WebLogic Server for OCI instance recommended to replace it. This illustration represents the state of both instances during the migration process, prior to deleting the Oracle Java Cloud Service instance.

Figure 2-2   Oracle Java Cloud Service with IDCS



The private subnet for the OCI database, the public subnet for the OCI load balancer, and the public subnet for JCS must already exist. Additionally, you should create the following subnets:

- Oracle WebLogic Server for OCI private subnet
- Bastion host public subnet

Use the following tables to configure the security lists for the these three subnets:

Table 2-5   Oracle WebLogic Server for OCI Private Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	Bastion public subnet, TCP	22	SSH access
Stateful Ingress	Bastion public subnet, TCP	7002	Administration server SSL port

Table 2-5 (Cont.) Oracle WebLogic Server for OCI Private Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	OCI load balancer public subnet, TCP	9999	IDCS app gateway port
Stateful Ingress	Oracle Java Cloud Service public subnet, TCP	22	SSH access, so domain and binaries can be copied over
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	22	SSH access between the compute instances of Oracle WebLogic Server for OCI
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	9071	Used for provisioning and scaling
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	9073	IDCS app gateway for accessing the Managed Server
Stateful Ingress	Oracle WebLogic Server for OCI private subnet (self), TCP	5556	Used for accessing node manager

Table 2-6 Bastion Public Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	Hosts with access to bastion, TCP	22	SSH access, port forwarding

Use the following table to add access for the Oracle WebLogic Server for OCI private subnet to the OCI database public subnet:

Table 2-7 Access to OCI Database Public Subnet

Rule Type	Source CIDR and Protocol	Default Destination Port	Description
Stateful Ingress	WLS for OCI private subnet, TCP	1521	DB access

Prepare for Migration

Before you start the migration process, you need to complete the prerequisites steps in Before You Begin with Oracle WebLogic Server for OCI.

When performing the prerequisites steps, ensure to that you set the Oracle Java Cloud Service WebLogic administration user name and password as the OCI secret.

This ensures that your password with the WebLogic Server global `Admin` role in the Oracle Java Cloud Service instance is stored as an OCI secret to support running the required `start` or `stop` scripts. See Default Global Roles.

1. Make a note of the user name and password for a user with the WebLogic Server administration role in the Oracle Java Cloud Service.
2. When you get to the prerequisite step Create Secrets for Passwords, add the password to the OCI secrets.

3

Migrate an Instance

Use the following section to migrate your Oracle WebLogic Server domain resources and applications from your existing Oracle Java Cloud Service instance in Oracle Cloud Infrastructure to a new domain in Oracle WebLogic Server for OCI.

> **Note:**
>
> Migration is supported from Oracle Java Cloud Service 12.2.1.x release to any of the existing versions of Oracle WebLogic Server for OCI.

In this procedure,

- Oracle Java Cloud Service is the source instance that you would be migrating from.
- Oracle WebLogic Server for OCI is the target instance that you would be migrating to.

Topics:

- Migrate the DBCS Data to an OCI Database System
- Create an Oracle WebLogic Server for OCI Instance
- Stop the WebLogic Server Processes
- Replace the Domain and Binaries
- Update the Domain Content
- Start the WebLogic Server Processes
- Migrate IDCS

Migrate the DBCS Data to an OCI Database System

Use the instructions provided in this section to migrate the Pluggable Database (PDB) in a Database Cloud Service (DBCS) instance used by Oracle Java Cloud Service to an Oracle Cloud Infrastructure Database System so that it can be used by an Oracle WebLogic Server for OCI instance.

At a high level, the migration process comprises the following tasks:

1. Create an 12.2.*x* OCI Database System.
2. Gather the credentials and other database information that will be needed by the Oracle Data Pump.
3. Create a listener alias for the PDB in the 12.2.*x* Oracle Cloud Infrastructure Database System, if one does not exist.
4. Use the Oracle Data Pump to export the PDB contents from the DBCS instance into a dump file.

5. Use the Oracle Data Pump to import the dump file into a PDB in the 12.2.*x* OCI Database System.

> **Note:**
>
> The instructions provided in this section have been verified with a 12.2.*x* version of the Oracle Cloud Infrastructure Database System with Oracle Database Enterprise Edition. You may require to amend these steps if you plan to use a different version of the database system.

Create an Oracle Cloud Infrastructure Database System

Create a 12.2.*x* Oracle Cloud Infrastructure Database System using the Oracle Cloud Infrastructure Console (see Creating a DB System) or use an existing 12.2.*x* Oracle Cloud Infrastructure Database System. You must create the database system in the same region as the Oracle WebLogic Server for OCI instance to which you plan to migrate the Oracle Java Cloud Service instance.

To simplify the migration, if you are creating a new Oracle Cloud Infrastructure Database System, follow these guidelines:

- Under **Configure storage**, change the storage to **Logical Volume Manager**.
  - This storage option is consistent with the storage option that DBCS uses.
  - If you choose Oracle Grid Infrastructure, you will need to perform additional actions for the Oracle Data Pump import to succeed. See Oracle Cloud Infrastructure Database System with Oracle Grid Infrastructure.
- Create the Oracle Cloud Infrastructure Database System in the same VCN that you will use for the WebLogic Server in Oracle WebLogic Server for OCI.

Gather Information from the Oracle Database Cloud Service

Obtain the following information from the Oracle Database Cloud Service:

- The password for the `sys` user.
- The password for the user who will the create dump file (`system` user is preferred).
- The database node to which the `opc` user can `ssh`.
- The directory where you will place the dump file.
  - This directory must be accessible by the `oracle` user.
  - The examples provided in the instructions use `/home/oracle` as the location of the dump file.
- The name of the PDB that includes the Java Required Files (JRF) schemas.
  If you are unsure of the PDB name, `ssh` to one of the nodes in the database as the `opc` user and run the following commands:

  ```
  sudo su - oracle
  sqlplus
  <At the prompt enter>:
  sys as sysdba
  ```

```
<enter the password when prompted>
show pdbs;
exit;
```

- The listener alias for the PDB. See Ensure that the Listener Alias for the Pluggable Database is Defined.

Gather Information from the Oracle Cloud Infrastructure Database System

Obtain the following information from the Oracle Cloud Infrastructure Database System:

- The password for the `sys` user.
- The password for the user who will the create dump file (`system` user is preferred).
- The database node to which the `opc` user can `ssh`.
- The directory where you will place the dump file.
  - This directory must be accessible by the `oracle` user.
  - The examples provided in the instructions use `/home/oracle` as the location of the dump file.
- The name of the PDB into which you will import the Java Required Files (JRF) schemas. If you are unsure of the PDB name, `ssh` to a node on the Oracle Cloud Infrastructure Database System as the `opc` user and run the following commands:

```
sudo su - oracle
sqlplus
<At the prompt enter>:
sys as sysdba
<enter the password when prompted>
show pdbs;
exit;
```

- The listener alias for the PDB. See Ensure that the Listener Alias for the Pluggable Database is Defined.

Ensure that the Listener Alias for the Pluggable Database is Defined

> **Note:**
>
> By default, the Oracle Cloud Infrastructure Database System nodes do not have the listener alias configured for the PDB.

You must determine if an alias exists, and create one if it is not found.

1. `ssh` to one of the nodes in the Oracle Cloud Infrastructure Database System and check the status of the listener.

   ```
   sudo su - oracle
   
   
   lsnrctl status | grep -i pdb_name
   ```

   For example:

   ```
   sudo su - oracle
   
   
   lsnrctl status | grep -i PDB1
   ```

2. Retrieve the service name, a value surrounded by quotes, and includes the host domain name. You can locate the domain name of the host in the DB System Details page of the Oracle Cloud Infrastructure Console.

   This is an example of the output:

   ```
   Service "pdb1.sometext.vcnfqdn" has 1 instance(s)
   ```

3. List out the contents of the `tnsnames.ora` file.

   ```
   cat /u01/app/oracle/product/db_version/db_home/network/admin/
   tnsnames.ora
   ```

   For example:

   ```
   cat /u01/app/oracle/product/12.2.0/dbhome_1/network/admin/
   tnsnames.ora
   ```

4. Identify if there is an alias defined for the service name from the output of the `cat` command used in Step 3. If you find an alias, make a note of it. If you do not see an alias for the service name, open the `tnsnames.ora` file in a text editor, such as Vim, and add the listener alias.

   ```
   vi /u01/app/oracle/product/db_version/db_home/network/admin/
   tnsnames.ora
   
   
   PDB1 =
     (DESCRIPTION =
       (ADDRESS = (PROTOCOL = TCP)(HOST = db_host_name)(PORT =
   db_port))
       (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = service_name_retrieved_from_lsnrctl_status)
       )
     )
   ```

> **Note:**
>
> You can retrieve the host and port values from other entries in the `tnsnames.ora` file.

Example of an alias named `PDB1` for the service name `pdb1.sometext.vcnfqdn`:

```
vi /u01/app/oracle/product/12.2.0/dbhome_1/network/admin/tnsnames.ora


PDB1 =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = my.dbhost.vcnfqdn)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = pdb1.sometext.vcnfqdn)
    )
  )
```

5. Save the file and exit the Vim editor.
6. Verify the `sqlplus` connectivity with the database user name that you use to import the dump file. The `system` user name is preferred and used in the examples where the listener alias is defined as `PDB1`:

   ```
   sqlplus
   <At the prompt enter>:
   system@PDB1
   <enter password when prompted>
   ```

   If this step fails, you may need to exit `sqlplus` and run the `lsnrctl reload` command and try again.

Identify the Directory for the JRF Tablespaces

When Oracle Java Cloud Service creates the Java Required Files schemas, it creates the tablespaces for the schemas in the `/u02/app/oracle/oradata` location, under a uniquely named directory. The Oracle Data Pump import utility does not create this directory and will throw errors if it is not present. Therefore, you must make a note of the directory hosting the tablespaces.

1. `ssh` to one of the nodes in the DBCS and list the content:

   ```
   ls /u02/app/oracle/oradata
   ```

   Example of the output:

   ```
   pdb_zjp_phx
   ```

2. Save this directory name for later use.

> **Note:**
>
> In many versions of the DBCS system, there will not be a subdirectory. In such a case, you will see the `<RCU_prefix_name>__ias_opss.dbf` file and several other `*.dbf` files with the RCU prefix name. If this is the case, you will not need to record anything.

Export the Pluggable Database from the DBCS System

To export the PDB:

1. `ssh` to one of the nodes in the DBCS system and list the contents.
2. Become the `oracle` user and access `sqlplus`.

   ```
   sudo su - oracle
   sqlplus
   <At the prompt enter>:
   sys as sysdba
   <enter password when prompted>
   ```

3. Switch to the PDB.

   ```
   alter session set container = pdb_name_for_DBCS;
   ```

   For example:

   ```
   alter session set container = PDB1;
   ```

4. Create an alias to the path where the dump file will be placed.

   ```
   create or replace directory dump_file_alias as
   'path_to_place_dump_file';
   ```

   For example:

   ```
   create or replace directory dbcs_pdb1_dump as '/home/oracle';
   ```

5. Grant the right to a user to write to this path alias.

   ```
   grant read, write on directory dump_file_alias to db_user;
   ```

   For example:

   ```
   grant read, write on directory dbcs_pdb1_dump to system;
   ```

6. Grant full right to a user to export the database and exit.

   ```
   grant exp_full_database to db_user;


   exit;
   ```

   For example:

   ```
   grant exp_full_database to system;


   exit;
   ```

7. Run the `export` command from outside the `sqlplus` prompt as the `oracle` user. Provide the password for the `database` user when prompted.

   ```
   expdp db_user@pdb_name_for_DBCS directory=dump_file_alias
   dumpfile=dump_file_name logfile=log_file_name full=y
   ```

   For example:

   ```
   expdp system@PDB1 directory=dbcs_pdb1_dump dumpfile=dbcs_pdb1_dump.dmp
   logfile=dbcs_pdb1_dump.log full=y
   ```

   The dump process will take time. If the job completes successfully, the last line of the output will include the text `successfully completed`.

8. Record the size of the dump file to ensure that the Oracle Cloud Infrastructure Database System will have enough space to accept the file.

   ```
   du -sh path_to_place_dump_file/dump_file_name
   ```

   For example:

   ```
   du -sh /home/oracle/dbcs_pdb1_dump.dmp
   ```

9. Save the file size for use in the next step. See Step 2 of Move the Dump File to the Oracle Cloud Infrastructure Database System.

Move the Dump File to the Oracle Cloud Infrastructure Database System

If you have access to the Oracle Cloud Infrastructure Database System, then, as the `opc` user, use the `scp` command to copy the files. Otherwise, use the Oracle Cloud Infrastructure Object Storage to upload the dump file to a bucket that is accessible to the Oracle Cloud Infrastructure Database System. See Uploading Objects to a Bucket or Folder.

Verify that there is enough space in the target Oracle Cloud Infrastructure Database System.

1. `ssh` to one of the nodes in the Oracle Cloud Infrastructure Database System.

   ```
   sudo su - oracle
   ```

   ```
   df -h dump_file_location
   ```

   For example:

   ```
   sudo su - oracle
   ```

   ```
   df -h /home/oracle
   ```

2. Compare the amount of free space free to the size of the dump file determined at the end exporting the PDB. See Step 9 of Export the Pluggable Database from the DBCS System. If the space is insufficient, increase the space for the storage volume before proceeding.

   If you uploaded the dump file to an Oracle Cloud Infrastructure Object Storage bucket, get the dump file from the bucket and place it in the directory that you chose at the time of gathering information from the Oracle Cloud Infrastructure Database System (see Gather Information from the Oracle Cloud Infrastructure Database System). In the examples, this location will be `/home/oracle`. See Downloading an Object from a Bucket or Folder.

   After you copy the file, ensure that you grant the `oracle` user `read` access on the dump file and `execute` access on the parent directories of the file.

Prepare the OCI Database System to Import the JRF Tablespaces and Schemas

Oracle recommends using the Logical Volume Manager as the storage option for the Oracle Cloud Infrastructure Database System into which you will import the JRF tablespaces and schemas. However, the Oracle Grid Infrastructure option is also supported.

Follow the instructions depending on the type of storageyou plan to use:

- Oracle Cloud Infrastructure Database System with the Logical Volume Manager
- Oracle Cloud Infrastructure Database System with Oracle Grid Infrastructure

Oracle Cloud Infrastructure Database System with the Logical Volume Manager

Create the directory in which the JRF tablespaces exist on the DBCS system (see Identify the Directory for the JRF Tablespaces). The Oracle Data Pump import utility will not create this directory and will throw errors if it is not present. If there was no subdirectory, which can be the case for many versions of DBCS, then you can skip this step.

For example:

```
cd /u02/app/oracle/oradata


mkdir pdb_zjp_phx
```

Oracle Cloud Infrastructure Database System with Oracle Grid Infrastructure

Oracle Grid Infrastructure uses Automatic Storage Management (ASM). The Oracle Data Pump import utility attempts to recreate the database files according to the location where they are stored. Since DBCS uses a file system, as opposed to ASM, the Oracle Data Pump import utility will create the dump file with the location of the files based on where they are physically stored on the disk. Therefore, if you attempt to import this dump file into the Oracle Grid Infrastructure, the import will fail to import any of the JRF tablespaces or schemas.

To avoid this failure, you must recreate the tablespaces and schemas for each Oracle WebLogic Server JRF domain that uses the DBCS system.

`ssh` to the Oracle Java Cloud Service administration host for the domain (the host name will end in `wls-1`) as the `opc` user and follow these instructions:

1. `ssh` to the Oracle Java Cloud Service administration host for the domain (the host name will end in `wls-1`) as the `oracle` user:

   ```
   sudo su -oracle


   vi /tmp/create_sql_for_jrf_schemas.py
   ```

2. Generate the SQL script to create the tablespaces and schemas.

   a. Copy the following `create_sql_for_jrf_schemas.py` script to the `/tmp` directory:

      ```
      import os
      import sys
      from xml.dom import minidom
      import weblogic.security.internal.SerializedSystemIni as
      SerializedSystemIni
      import weblogic.security.internal.encryption.ClearOrEncryptedService
      as ClearOrEncryptedService

      def get_node_text(node):
          nodelist = node.childNodes
          for node in nodelist:
              if node.nodeType == node.TEXT_NODE:
                  return node.data

      if len(sys.argv) != 2:
          print("Supply the domain path as the only argument.")
          sys.exit(1)

      domain_path = sys.argv[1]
      # Get the schema prefix and pass from OPSS data source. The
      ```

```
assumption is that all the values for the JRF schemas are the
same.
ds_name = 'opss-data-source'
readDomain(domain_path)
cd('/JDBCSystemResource/' + ds_name + '/JdbcResource/' + ds_name
+ '/JDBCDriverParams/NO_NAME_0/Properties/NO_NAME_0/Property/
user')
schema_prefix = get('Value').split('_')[0].upper()

# wlst does not allow reading of the password so parse directly
from the config file
doc = minidom.parse(domain_path + '/config/jdbc/opss-datasource-
jdbc.xml')
driver_params = doc.getElementsByTagName('jdbc-driver-params')
[0].childNodes
for param in driver_params:
    if param.nodeName == "password-encrypted":
        system_ini =
SerializedSystemIni.getEncryptionService(domain_path)
        encryption_service = ClearOrEncryptedService(system_ini)
        pass_phrase =
encryption_service.decrypt(get_node_text(param))
        break
    if param.nodeName == "password":
        pass_phrase = get_node_text(param)

current_dir = os.getcwd()
sql_file = open(current_dir + '/create_jrf_data.sql', 'w')
# Create the tablespaces
sql_file.write("create tablespace " + schema_prefix + "_IAS_OPSS
datafile '+DATA' SIZE 62914560 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_IAU
datafile '+DATA' SIZE 62914560 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_MDS
datafile '+DATA' SIZE 104857600 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_IAS_UMS
datafile '+DATA' SIZE 104857600 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_STB
datafile '+DATA' SIZE 10485760 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_WLS
datafile '+DATA' SIZE 62914560 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
sql_file.write("create tablespace " + schema_prefix + "_IAS_TEMP
datafile '+DATA' SIZE 62914560 AUTOEXTEND ON NEXT 8192 MAXSIZE
32767M;\n")
# Create a required role stored in those tablespaces
sql_file.write("create role APEX_GRANTS_FOR_NEW_USERS_ROLE;\n")
# Create the schema users
sql_file.write("create user " + schema_prefix + "_OPSS
identified by " + pass_phrase + ";\n")
```

```
sql_file.write("create user " + schema_prefix + "_IAU identified by "
+ pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_UMS identified by "
+ pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_WLS_RUNTIME
identified by " + pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_WLS identified by "
+ pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_MDS identified by "
+ pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_IAU_VIEWER
identified by " + pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_IAU_APPEND
identified by " + pass_phrase + ";\n")
sql_file.write("create user " + schema_prefix + "_STB identified by "
+ pass_phrase + ";\n")
# Grant unlimited quota to most important schema users on specific
tablespaces
sql_file.write("alter user " + schema_prefix + "_OPSS quota unlimited
on " + schema_prefix + "_IAS_OPSS;\n")
sql_file.write("alter user " + schema_prefix + "_UMS quota unlimited
on " + schema_prefix + "_IAS_UMS;\n")
sql_file.write("alter user " + schema_prefix + "_WLS_RUNTIME quota
unlimited on " + schema_prefix + "_WLS;\n")
sql_file.write("alter user " + schema_prefix + "_WLS quota unlimited
on " + schema_prefix + "_WLS;\n")
sql_file.write("alter user " + schema_prefix + "_MDS quota unlimited
on " + schema_prefix + "_MDS;\n")
sql_file.write("alter user " + schema_prefix + "_STB quota unlimited
on " + schema_prefix + "_STB;\n")
sql_file.write("commit;\n")
sql_file.close()
```

b. As the `oracle` user, execute the script:

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh /tmp/
create_sql_for_jrf_schemas.py <path to your domain>
```

The script will place a `create_jrf_data.sql` file in your current directory.

3. Execute the SQL script on the Oracle Grid Infrastructure Oracle Cloud Infrastructure Database System.

   a. Copy this script or the contents of the script to the Oracle Grid Infrastructure Oracle Cloud Infrastructure Database System.

   b. Make the script accessible to the `oracle` user.

   c. Log into the database as a user with the `sysdba` role.

```
sudo su - oracle
sqlplus
<At the prompt enter>:
sys as sysdba
<enter the password when prompted>
```

   d. Switch to the target PDB on the Oracle Cloud Infrastructure Database System.

   ```
   alter session set container = pdb_name_for_OCI_DB;
   @<path to generated sql file>
   ```

   For example:

   ```
   alter session set container = PDB1;
   @/home/oracle/create_jrf_data.sql
   ```

4. Repeat these steps for each Oracle Java Cloud Service instance that uses the DBCS instance.

Import the PDB from DBCS into the Oracle Cloud Infrastructure Database System

To import the PDB:

1. `ssh` to one of the nodes in the Oracle Cloud Infrastructure Database System and list the contents.
2. Become the `oracle` user and access `sqlplus`.

   ```
   sudo su - oracle
   sqlplus
   <At the prompt enter>:
   sys as sysdba
   <enter password when prompted>
   ```

3. Switch to the PDB on the Oracle Cloud Infrastructure Database System.

   ```
   alter session set container = pdb_name_for_OCI_DB;
   ```

   For example:

   ```
   alter session set container = PDB1;
   ```

4. Create an alias to the path where the dump file will be placed.

   ```
   create or replace directory dump_file_alias as
   'path_to_place_dump_file';
   ```

   For example:

   ```
   create or replace directory dbcs_pdb1_dump as '/home/oracle';
   ```

5. Grant the right to a user to write to this path alias.

   ```
   grant read, write on directory dump_file_alias to db_user;
   exit;
   ```

For example:

```
grant read, write on directory dbcs_pdb1_dump to system;
exit;
```

6. Run the `import` command from outside the `sqlplus` prompt as the `oracle` user. Provide the password for the `database` user when prompted.

   ```
   impdp db_user@tns_alias directory=dbcs_pdb1_dump
   dumpfile=dbcs_pdb1_dump.dmp logfile=log_file_name full=y
   ```

   For example:

   ```
   impdp system@PDB1 directory=dbcs_pdb1_dump dumpfile=dbcs_pdb1_dump.dmp
   logfile=dbcs_pdb1_dump.log full=y
   ```

   The command should complete with few errors because certain object types such as tablespace temp and tablespace users already exist. This is an expected behavior.

7. Search for `ORA-` in the log file to examine the errors.

   For example:

   ```
   cat /home/oracle/dbcs_pdb1_dump.log | grep ORA-
   ```

   Check that the errors end with `already exists` or are some other expected errors such as `"USER:"SYS" failed due to insufficient privileges`. The main errors to look for are with the JRF schemas. For example, an error that includes `_OPSS` in it.

8. Before exiting, construct the connect string that will be used by examining the `tnsnames.ora` entries.

   ```
   cat /u01/app/oracle/product/db_version/db_home/network/admin/tnsnames.ora
   ```

   For example:

   ```
   cat /u01/app/oracle/product/12.2.0/dbhome_1/network/admin/tnsnames.ora
   ```

9. Locate the listener alias.

   For example:

   ```
   cat /u01/app/oracle/product/12.2.0/dbhome_1/network/admin/tnsnames.ora


   PDB1 =
     (DESCRIPTION =
       (ADDRESS = (PROTOCOL = TCP)(HOST = my.dbhost.vcnfqdn)(PORT = 1521))
       (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = pdb1.sometext.vcnfqdn)
   ```

```
          )
       )
```

10. Record the `HOST:PORT/SERVICE_NAME` as the connect string. You will need this when you are migrating your Oracle Java Cloud Service domain contents to Oracle WebLogic Server for Oracle Cloud Infrastructure.

    Using this example, the connect string will be:

    ```
    my.dbhost.vcnfqdn:1521/pdb1.sometext.vcnfqdn
    ```

Create an Oracle WebLogic Server for OCI Instance

From OCI Marketplace create either a **Oracle WebLogic Suite UCM** or **Oracle WebLogic Server Enterprise Edition UCM** instance corresponding to the WebLogic edition you used in your Oracle Java Cloud Service. Irrespective of the WebLogic version used in your Oracle Java Cloud Service instance, create a Oracle WebLogic Server for OCI 12.2.1.4 instance.

Create an Oracle WebLogic Server for OCI instance by following the steps documented in Create a Domain in *Using Oracle WebLogic Server for OCI*, along with the following customizations:

- Select the **WLS Instance Advanced Configuration** option and then change the **WebLogic Managed Server External Port** and **WebLogic Managed Server External SSL Port** to match those used in Oracle Java Cloud Service.
  By default, Oracle Java Cloud Service uses ports `8001` and `8002`, respectively.
- If your Oracle Java Cloud Service instance used OTD, then select **Provision Load Balancer**.
- If your Oracle Java Cloud Service instance used Oracle Identity Cloud Service (IDCS), then select **Provision Load Balancer** and **Enable Authentication Using Identity Cloud Service**.
- Although all Oracle Java Cloud Services use JRF, you *do not* need to select the **Provision with JRF** option, since the domain migrated from the Oracle Java Cloud Service instance points to the existing JRF database schemas, not the new schemas created with the JRF option.
- Oracle WebLogic Server for OCI creates all block volumes at `50 GB`. If you have block volumes for the Oracle Java Cloud Service domain or common volumes with larger sizes, set the same sizes for the corresponding Oracle WebLogic Server for OCI data and middleware volumes prior to migration.
- Oracle WebLogic Server for OCI creates only a data and block volume. If you created additional volumes for your Oracle Java Cloud Service instance, add those volumes using the OCI console after completing the provisioning process. See Adding a Block Volume in the Oracle Cloud Infrastructure documentation.

Stop the WebLogic Server Processes

Stop all the WebLogic Server processes on both the Oracle WebLogic Server for OCI and Oracle Java Cloud Service instances.

In **Oracle WebLogic Server for OCI**:

1. As an `opc` user, log in to the each non-administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o stop
   # run jps to confirm that there are no more processes are running. If
   there are run kill -9 on each.
   jps
   ```

2. As an `opc` user, log in to the Administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o stop
   # run jps to confirm that there are no more processes are running. If
   there are run kill -9 on each.
   jps
   ```

In **Oracle Java Cloud Service**:

Using the Oracle WebLogic Server Administration Console, stop the WebLogic servers on the Oracle Java Cloud Service instance.

Oracle recommends that you stop the WebLogic servers in your Oracle Java Cloud Service prior to beginning the migration process. Use the Oracle WebLogic Server Administration Console to stop all the WebLogic servers on the Oracle Java Cloud Service instance. However, if you have minimal downtime requirements, you can wait to stop the WebLogic servers in your Oracle Java Cloud Service at the point in which you start the WebLogic Server processes for the migrated Oracle WebLogic Server for OCI.

Replace the Domain and Binaries

Replace the domain and binaries in the Oracle WebLogic Server for OCI instance.

Complete the following steps:

1. Access the VM running the WebLogic administration server in the Oracle Java Cloud Service instance and run the following commands to preserve the WebLogic domain and binaries:

   ```
   sudo su - oracle
   cd /u01/data/domains
   tar cvf /tmp/domain.tar .
   cd /u01/app/
   tar cvf /tmp/mw.tar .
   cd /u01/jdk/
   tar cvf /tmp/jdk.tar .
   ```

2. Log in as an `oracle` user and `ssh` to each additional VM in the Oracle Java Cloud Service instance, and preserve the WebLogic domain contents.

You do not have to save the binaries from each VM because the binaries are the same on each VM.

```
cd /u01/data/domains
tar cvf /tmp/domain.tar .
```

3. Use OCI Object Storage to upload the tar files to a bucket that is accessible to the compute instances in the Oracle WebLogic Server for OCI deployment. See Uploading Objects to a Bucket or Folder.
4. Access the Oracle WebLogic Server for OCI instance as the `opc` user.
5. On each VM in the Oracle WebLogic Server for OCI instance, run the following command to move the contents out of the current domain, middleware home, and JDK home:

   ```
   sudo su - oracle
   mkdir /tmp/domain_bak
   mv /u01/data/domains/* /tmp/domain_bak/
   mkdir /tmp/mw_bak
   mv /u01/app/* /tmp/mw_bak/
   ```

6. Download the `domain.tar`, `mw.tar`, and `jdk.tar` files from object storage into the `/tmp` directory. See Downloading an Object from a Bucket or Folder.

   ```
   sudo su - oracle
   scp -o UserKnownHostsFile=/dev/null -o
   StrictHostKeyChecking=no /tmp/domain.tar 10.1.1.1:/tmp/
   scp -o UserKnownHostsFile=/dev/null -o
   StrictHostKeyChecking=no /tmp/mw.tar 10.1.1.1:/tmp/
   scp -o UserKnownHostsFile=/dev/null -o
   StrictHostKeyChecking=no /tmp/jdk.tar 10.1.1.1:/tmp/
   ```

7. `ssh` to the Oracle WebLogic Server for OCI instance as the `opc` user and on each VM run the following to `untar` the source instance domain:

   ```
   sudo su - oracle
   tar xvf /tmp/domain.tar -C /u01/data/domains/
   tar xvf /tmp/mw.tar -C /u01/app/
   mkdir /u01/app/oracle/jdk
   tar xvf /tmp/jdk.tar -C /u01/app/oracle/jdk/
   ```

> **Note:**
>
> Oracle WebLogic Server for OCI creates only a data and block volume. If you created additional volumes for your Oracle Java Cloud Service instance, add a new volume and transfer the data using the above steps.

Update the Domain Content

To update the domain content by using the cloning scripts, complete the following steps:

1. As a `oracle` user, run the following commands on each VM in the Oracle WebLogic Server for OCI instance:

   ```
   sudo su - oracle
   python3 /opt/scripts/cloning/update_service_config.py metadata 1
   ```

   This ensures that the metadata used by restart and scaling scripts on Oracle WebLogic Server for OCI has the correct information.

   > **Note:**
   >
   > Oracle Java Cloud Service limits the domain objects to 8 characters. However, Oracle WebLogic Server for OCI does not have this limitation.

2. Change the host names in the domain configuration files to the host name of the VMs in the Oracle WebLogic Server for OCI instance.
   As a `oracle` user, run the following command on each VM in the Oracle WebLogic Server for OCI instance:

   ```
   sudo su - oracle
   python3 /opt/scripts/cloning/update_service_config.py hostname 1
   ```

3. If the Demo Identity certificates have the incorrect host name, then, as a `oracle` user, run the following command on each VM in the Oracle WebLogic Server for OCI instance

   ```
   sudo su - oracle
   /opt/scripts/cloning/generate_demo_certs.sh
   ```

4. Ensure that there is a marker file present so Compute instance reboot restarts the servers:

   ```
   touch /u01/data/domains/<jcs_domain_name>/provCompletedMarker
   ```

   where `<jcs domain name>` can be found by running:

   ```
   ls /u01/data/domains/ | grep _domain
   ```

Replace the Database Connections

Before performing this step, ensure that you have completed the migration of DBCS to the Oracle Cloud Infrastructure Database System. See Migrate the DBCS Data to an OCI Database System.

1. Log in or `sudo` as the user owning the Oracle WebLogic Server files and locate your WebLogic domain configuration's config directory. If *$DOMAIN_HOME* is empty, set *DOMAIN_HOME* to the domain home of your Oracle WebLogic Server installation.

   ```
   sudo su - oracle
   ```

   ```
   cd $DOMAIN_HOME/config
   ```

2. Back up the `jdbc` directory and the `fmwconfig` configuration files:

   ```
   cp -rp ./jdbc /tmp/
   ```

   ```
   cp -rp ./fmwconfig/jps-config*.xml /tmp/
   ```

3. From the command line, get the existing connection string:

   ```
   grep url jdbc/opss-datasource-jdbc.xml
   ```

   An example of the output:

   ```
   <url>jdbc:oracle:thin:@//dbcshostfqdn:1521/pdb1.svcname</url>
   ```

4. Extract the value between `<url>jdbc:oracle:thin:@//` and `</url>`. Continuing with the example, this is the value you would extract:

   ```
   dbcshostfqdn:1521/pdb1.svcname
   ```

5. Replace the connection string retrieved (oldconnect) with the service value you fetched at the end of migrating DBCS to the Oracle Cloud Infrastructure Database System (newconnect). See Import the PDB from DBCS into the Oracle Cloud Infrastructure Database System.

   ```
   sed -i 's;oldconnect;newconnect;g'./jdbc/*.xml
   ```

   For example:

   ```
   sed -i 's;dbcshostfqdn:1521/pdb1.svcname;my.dbhost.vcnfqdn:1521/
   pdb1.sometext.vcnfqdn;g'./jdbc/*.xml
   ```

6. Replace the connection string in the `fmwconfig` configuration files using the connection string (newconnect) from Step 5.

   ```
   sed -i 's;oldconnect;newconnect;g'./fmwconfig/jps-config*.xml
   ```

   For example:

   ```
   sed -i 's;dbcshostfqdn:1521/pdb1.svcname;my.dbhost.vcnfqdn:1521/
   pdb1.sometext.vcnfqdn;g'./fmwconfig/jps-config*.xml
   ```

Start the WebLogic Server Processes

If you did not stop the WebLogic servers on the Oracle Java Cloud Service instance, then stop the WebLogic servers on the Oracle Java Cloud Service instance now. Oracle does not support running two WebLogic domains using JRF simultaneously. Use the WebLogic Server Administrative Console to stop all the servers.

Complete the following steps on the migrated Oracle WebLogic Server for OCI instance:

1. As an `opc` user, log in to the Administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o start
   # run jps to confirm that the processes are running.
   jps
   ```

2. As an `opc` user, log in to the each non-administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o start
   # run jps to confirm that the processes are running.
   jps
   ```

Migrate IDCS

The Oracle Identity Cloud Service (IDCS) confidential application created by Oracle WebLogic Server for OCI must replace the IDCS application created for Oracle Java Cloud Service.

Complete the following steps:

1. Update the client and secret in the IDCS authentication provider.
   a. From the IDCS console, expand the **Navigation Drawer**, and then click **Applications**.
   b. Click the enterprise application associated with your domain.
      The name of the application is: `<`*`stack`*`>_confidential_idcs_app_<`*`timestamp`*`>`

      Example: `myweblogic_confidential_idcs_app_2019-08-01T01:02:01.123456`.
   c. Click the **Configuration** tab.
   d. Retrieve the `Client ID` and `Client Secret` values.
   e. Use the WebLogic Server Administration Console to update the client and secret in the IDCS authentication provider:
      i. Log in to WebLogic Server Administration Console.
         See Access the WebLogic Console in *Using Oracle WebLogic Server for OCI*.
      ii. Navigate to **security realms**.
      iii. Select the realm. By default this is `myrealm`.
      iv. Click on the **Providers** tab.

   v. Select **IDCSIntegrator**.
   vi. Click on the **Provider Specific** tab.
   vii. Click **Lock and Edit**.
   viii. Change the **Client Id** and **Client Secret** to the values retrieved from the IDCS confidential application.

2. Add the `Cloudgate` role.

   The IDCS confidential application created for the Oracle WebLogic Server for OCI instance sets up a more restrictive default role, *Authenticator Client*, than is set up in Oracle Java Cloud Service. To keep the same level of permissions so that the Enterprise Manager Fusion Middleware Control console, update the client application to add the *Cloud Gate* role. See Integrate OPSS User and Group APIs with Identity Cloud Service.

3. Restart the WebLogic Server processes on the Oracle WebLogic Server for OCI instance:

   a. As an opc user, log in to the Administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh
   # run jps to confirm that the processes are running.
   jps
   ```

   b. As an opc user, log in to the each non-administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh
   # run jps to confirm that the processes are running.
   jps
   ```

4. Move any protected paths.

   If you set up any server context paths to be protected by IDCS, then you must migrate them. See Migrate Oracle Identity Cloud Service Roles and Policies.

4

Post-Migration

After successfully migrating your Oracle Java Cloud Service instances to Oracle WebLogic Server for OCI, perform the following post-migration tasks:

- If you were using custom SSL certificates on Oracle Traffic Director (OTD), then set up SSL certificates on the OCI Load Balancer.
- If you use a hosting provider to manage DNS make sure to reset the `CNAME` records at your hosting provider to point to the new IP addresses of the OCI load balancer and WebLogic VMs.
- Verify that all the deployed applications are functioning as expected.
  If you have enabled Oracle WebLogic Server Administration Console communication, 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.
- It is recommended that you stop your Oracle Java Cloud Service instance using the Oracle Java Cloud Service console. With the instance stopped you will not be charged for the Oracle Java Cloud Service instance. If you discover any issues with the Oracle WebLogic Server for OCI set up you can always switch back to the Oracle Java Cloud Service by starting it up again. See Before You Begin with Oracle WebLogic Server for OCI.
- After monitoring the migrated Oracle WebLogic Server for OCI for a few days or weeks, if your applications are all running as expected, you should delete your Oracle Java Cloud Service instance. See Delete an Oracle Java Cloud Service Instance.
- If you migrated your Oracle Java Cloud Service instance to Oracle WebLogic Server for OCI, and the WebLogic Server version that you migrated is 12.2.1.2 or 12.2.1.3, you must upgrade the WebLogic Server versions post-migration.
  - If the WebLogic Server version is 12.2.1.3, see Upgrade a Domain in *Using Oracle WebLogic Server for OCI*.
  - If the WebLogic Server version is 12.2.1.2, see Upgrade a Domain.

Upgrade a Domain

If you migrated your Oracle Java Cloud Service instance to WebLogic Server version 12.2.1.2 for Oracle WebLogic Server for OCI, you must perform an in-place upgrade to upgrade the WebLogic Server version.

Topics:

- Stop the WebLogic Server Processes
- Perform Prerequisite Tasks
- Set up VNC Server
- Perform Readiness Check

- Upgrade Infrastructure Schemas
- Reconfigure the Domain
- Upgrade the Domain
- Restart Servers
- Post Upgrade
- Roll Back an Upgrade

Stop all WebLogic Server Processes

Stop all the WebLogic server processes on the Oracle WebLogic Server for OCI instance.

1. As an `opc` user, log in to the each non-administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o stop
   # run jps to confirm that there are no more processes are running.
   If there are run kill -9 on each.
   jps
   ```

2. As an `opc` user, log in to the Administration VM in the instance and run the following command:

   ```
   sudo su - oracle
   /opt/scripts/restart_domain.sh -o stop
   # run jps to confirm that there are no more processes are running.
   If there are run kill -9 on each.
   jps
   ```

Perform Prerequisite Tasks

Before upgrading your service instance, back up the database and the domain directories.

To perform prerequisite tasks:

> **Note:**
>
> You need to perform steps 1 to 3 in case the upgrade fails and you want to roll back the service instance to its original state.

1. Back up the Oracle Cloud Infrastructure database to object storage. See Create an On-Demand Full Backup of a Database in Oracle Cloud Infrastructure documentation.

2. On each VM in the Oracle WebLogic Server for OCI instance, back up the domain directory on your service node.

   ```
   sudo su - oracle
   mkdir /u01/data/old_domain_version
   cp -r /u01/data/domains/* /u01/data/old_domain_version/
   ```

3. On each VM in your Oracle WebLogic Server for OCI, move the middleware home contents to a temporary location to replace the older binaries with the WebLogic Server 12.2.1.4 binaries.

   ```
   sudo su - oracle
   mkdir /tmp/old_mw_version
   mv /u01/app/* /tmp/old_mw_version/
   ```

4. Copy the 12.2.1.4.0 binaries located in `/tmp/mw_bak/` to `/u01/app/`.

   > **Note:**
   >
   > When migrating your Oracle Java Cloud Service instances to Oracle WebLogic Server for OCI, you would have moved the 12.2.1.4 binaries to `/tmp/mw_bak/` location. See step 3e in Replace the Domain and Binaries.

   ```
   sudo su - oracle
   cp -r /tmp/mw_bak/* /u01/app/
   ```

Set up VNC Server

To use the Reconfiguration Wizard and Upgrade Assistant (Fusion Middleware tools) during the upgrade process, you need a graphical user interface (GUI) environment. The instructions in this step explain how to set up a VNC server and use port forwarding through a bastion host. If you are familiar with using X11, then X11 forwarding can be used to forward the GUI to your local desktop and you can skip this step.

> **Note:**
>
> This step is *not applicable* for a non-JRF instance.

1. Log in to administration compute instance as an `opc` user.
2. Run the following command to install the GUI packages on the target 12.2.1.4 instance:

   ```
   sudo bash
   yum group install "Server with GUI"
   # enter 'y' when prompted.
   ```

3. As an `opc` user, run the following command to set up the VNC server for the `oracle` user on the administration compute instance:

   ```
   sudo bash
   yum install tigervnc-server -y
   exit
   sudo su - oracle
   vncpasswd
   # enter the password and confirmation password.
   # respond with "n" when prompted if this should be a view only
   password
   exit
   sudo bash
   cp /lib/systemd/system/vncserver@.service /etc/systemd/system/
   vncserver@\:1.service
   vi /etc/systemd/system/vncserver@\:1.service
    Replace <USER> with oracle
   systemctl daemon-reload
   systemctl enable vncserver@\:1.service
   systemctl start vncserver@\:1.service
   systemctl status vncserver@\:1.service
   # Confirm it is running

   # Set up the firewall to allow the VNC server port to be accessed:
   iptables -I INPUT -m state --state NEW -p tcp --destination-port
   5901 -j ACCEPT
   ```

4. Sign in to the Oracle Cloud Infrastructure Console and update the subnet to have a security list that enables inbound access to the VNC server port `5901`.

   > **Note:**
   >
   > If you have not already set up your bastion VM to be able to access VMs as the `opc` user, then place your private key pem for the opc user on the disk. This is used for port forwarding.

5. Create the tunnel by accessing the bastion host as the `opc` user:

   ```
   ssh -i <privatekey.ppk> -L <vnc_port_on_wlsm-private-ip>:<wlsvm-
   private-ip>:<port_on_bastion> <wlsvm-private-ip>
   ```

   Following is an example, where the IP address of the administration compute instance is `10.1.1.1` and the private key for the `opc` user is in `~/.ssh/id_rsa`

   ```
   ssh -i ~/.ssh/id_rsa -L 5901:10.1.1.1:5901 10.1.1.1
   ```

Windows instructions for launching GUI

1. Install and launch PuTTY.
2. For **Host Name**, type the bastion IP address.
3. For **Saved Sessions**, type `bastion`.

4. Under **Category**, go to **Connection** > **Data**.
5. For **Auto-logion username**, type `opc`.
6. Under **Category**, go to **Connection** > **Data** > **SSH** > **Tunnels**.
7. Type the following values for the respective fields:
   - Source port: `5901`
   - Destination: `localhost:5901`
8. Click **Add**.
9. Under **Category**, go to **Connection** > **Data** > **SSH** > **Auth**.
10. For **Private key file for authentication**, browse and select the xperiment private key that you have created.
11. Under the **Category**, select **Session**.
12. Select **Save** and then select **Open** to establish the connection
13. Verify that you connected successfully to the putty session.
14. Install a VNC Viewer and set up a new connection to use `localhost:5901` to verify that you can connect correctly.

> **Note:**
>
> Ensure that you have set up the `vncserver` as the `oracle` user, as this creates a session with the `oracle` user even though you have port forwarding via the `opc` user ssh keys.

Perform Readiness Check

Perform a readiness check to determine if your service instance is ready for upgrade.

> **Note:**
>
> This step is *not applicable* for a non-JRF instance.

1. By using the VNC viewer session, start the Upgrade Assistant.

   ```
   export USER_MEM_ARGS=-Djava.security.egd=file:/dev/urandom
   /u01/app/oracle/middleware/oracle_common/upgrade/bin/ua -readiness
   ```

   Setting `USER_MEM_ARGS` to use the `/dev/urandom` device reduces the time it takes to run the Oracle Fusion Middleware upgrade tools.
2. Use the Upgrade Assistant to perform a readiness check. See Upgrade from a previous 12c release to 12.2.1.4 in *Upgrading to the Oracle Fusion Middleware Infrastructure*
3. On the **Readiness Check Type** screen, select the domain-based readiness check.

The domain-based readiness check enables the Upgrade Assistant to discover and select all upgrade-eligible schemas or component configurations in the domain specified in the Domain Directory field.

4. On the **End of Readiness** screen in the Upgrade Assistant, review the results of the readiness check (`Readiness Success` or `Readiness Failure`).

    - If the readiness check is successful, click **View Readiness Report** to review the complete report. Oracle recommends that you review the Readiness Report before you perform the upgrade even when the readiness check is successful. Use the **Find** option to search for a particular word or phrase within the report. The report also indicates where the completed Readiness Check Report file is located.
    - If the readiness check encounters an issue or error, click **View Log** to review the log file, identify and correct the issues, and then restart the readiness check.

Upgrade Infrastructure Schemas

This step helps to identify if you have an earlier version of infrastructure database schemas or have installed other Oracle products.

> **Note:**
>
> This step is *not applicable* for a non-JRF instance.

1. Start the Upgrade Assistant if you have not already done so. For example:

    ```
    export USER_MEM_ARGS=-Djava.security.egd=file:/dev/./urandom
    /u01/app/oracle/middleware/oracle_common/upgrade/bin/ua
    ```

2. Upgrade the schemas. See Upgrading Schemas Using the Upgrade Assistant in *Upgrading to the Oracle Fusion Middleware Infrastructure*:
3. On the Selected Schemas screen, select **All Schemas Used by a Domain**, and then enter a domain directory name in the **Domain Directory** field.

    The **All Schemas Used by a Domain** selection allows the Upgrade Assistant to discover and select all components that have a schema available to upgrade in the domain specified in the **Domain Directory** field. This is also known as a domain-assisted schema upgrade. In addition, the Upgrade Assistant prepopulates connection information on the schema input screens.

4. On the Upgrade Progress screen in the Upgrade Assistance, monitor the schema upgrade progress.
5. Finish the schema upgrade process.

    - If the schema upgrade succeeds, click **Close** to complete the upgrade and close the wizard.
    - If the upgrade fails, click **View Log** to view and troubleshoot the errors. The logs are available in the following directory:

        ```
        /u01/app/oracle/middleware/oracle_common/upgrade/logs
        ```

6. Remedy the database connection failure if one occurs. See Problems with Database Connectivity When Upgrading the Infrastructure Schema Database in *Administering Oracle Java Cloud Service*.

7. Verify the schema upgrade was successful by checking that the schemas in `schema_version_registry` have been properly updated. See Problems with Database Connectivity When Upgrading the Infrastructure Schema Database in *Upgrading to the Oracle Fusion Middleware Infrastructure*.

   One way to verify the schema upgrade is to use SQL*Plus commands to obtain data from the `SCHEMA_VERSION_REGISTRY`.

   a. Find the Oracle Java Cloud Service instance's schema prefix in the Upgrade Assistant log file at `/u01/app/oracle/middleware/oracle_common/upgrade/logs`.

   b. Connect to the database as a user having Oracle DBA privileges and run the following commands from SQL*Plus to get the current version numbers.

```
sqlplus / as sysdba
SQL> connect <user_name>/<password>@<host_name>:<port>/<service_name>
as sysdba
SQL> SELECT MRC_NAME,COMP_ID,OWNER,VERSION,STATUS,UPGRADED FROM
SCHEMA_VERSION_REGISTRY WHERE MRC_NAME like 'SP1556690734';
```

   Example output for the `SCHEMA_VERSION_REGISTRY`:

MRC_NAME	COMP_ID	OWNER	VERSION	STATUS	UPGRADED
SP1556690734	IAU	SP1556690734_IAU	12.2.1.2.0	VALID	Y
SP1556690734	IAU_APPEND	SP1556690734_IAU_APPEND	12.2.1.2.0	VALID	N
SP1556690734	IAU_VIEWER	SP1556690734_IAU_VIEWER	12.2.1.2.0	VALID	Y
SP1556690734	MDS	SP1556690734_MDS	12.2.1.3.0	VALID	Y
SP1556690734	OPSS	SP1556690734_OPSS	12.2.1.0.0	VALID	Y
SP1556690734	STB	SP1556690734_STB	12.2.1.3.0	VALID	Y
SP1556690734	UCSUMS	SP1556690734_UMS	12.2.1.0.0	VALID	N
SP1556690734	WLS	SP1556690734_WLS	12.2.1.0.0	VALID	N

Reconfigure the Domain

> **Note:**
>
> - This step is *not applicable* for a non-JRF instance.
> - Running the reconfiguration wizard is not required for an upgrade from 12.2.1.3. However, if it is not run and you do not replace 12.2.1.3 with 12.2.1.4 in config.xml, then you will encounter the Incorrect Version Numbers After a Reduced Downtime Upgrade issue.

1. Start the Reconfiguration Wizard as user `oracle` with the following logging options, with `log_file` as the absolute path of the log file you'd like to create for the domain reconfiguration session. This can be helpful if you need to troubleshoot the reconfiguration process.

   For example:

   ```
   /u01/app/oracle/middleware/oracle_common/common/bin/reconfig.sh -
   log_priority=all -log="/u01/reconfig0212.log"
   ```

2. Perform the reconfiguration tasks as described in *Upgrading to the Oracle Fusion Middleware Infrastructure*. See Reconfiguring the Domain with the Reconfiguration Wizard.
3. On the Advanced Configuration screen of the Reconfiguration Wizard, select **Deployment and Services**.
4. Target the `wsm-pm` app to the cluster containing the managed severs.
5. Click **Reconfig**.
6. Check the End of Configuration screen to learn whether the reconfiguration process completed successfully or failed.
   - If the reconfiguration is successful, **Oracle WebLogic Server Reconfiguration Succeeded** is displayed. The location of the domain that was reconfigured as well as the Administration Server URL (including the listen port) are displayed as well.
   - If the reconfiguration process did not complete successfully, an error message is displayed which indicates the reason. Take appropriate action to resolve the error.

Upgrade the Domain

> **Note:**
>
> This step is *not applicable* for a non-JRF instance.

1. Start the Upgrade Assistant, for example:

   ```
   export USER_MEM_ARGS=-Djava.security.egd=file:/dev/./urandom
   /u01/app/oracle/middleware/oracle_common/upgrade/bin/ua
   ```

2. Use the Upgrade Assistant to upgrade the domain configurations. See Upgrading the Domain Configurations with the Upgrade Assistant in *Upgrading to the Oracle Fusion Middleware Infrastructure*.
3. On the All Configurations screen, select **All Configurations Used by a Domain** and specify your domain location in the **Domain Directory** field. Enter the domain directory directly or click **Browse** to select a valid domain directory.
4. On the Upgrade Summary page, review the summary of the options you have selected for the component configuration upgrade, and then click **Upgrade** to start the upgrade process.
5. View the Upgrade Progress page to monitor the upgrade.
6. View the results and finish the upgrade.
   - If the upgrade succeeds, the Upgrade Success page is displayed. Click **Close** to complete the upgrade and close the wizard.
   - If the upgrade fails, the Upgrade Failure screen is displayed. Click **View Log** to view and troubleshoot the errors. The logs are available at

     ```
     ORACLE_HOME/oracle_common/upgrade/logs
     ```

Restart Servers

Access the 12.2.1.4 instance as the `opc` user, and on each VM run the following command:

```
sudo su - oracle
/opt/scripts/restart_domain.sh
```

Post Upgrade

If the upgrade was successful, complete following tasks if they apply to your instance:

- If you use a Hosting Provider to manage DNS, then reset the `CNAME` records at your Hosting Provider to point to the new IP addresses of the load balancer and WebLogic VMs.
- Destroy the source 12.2.1.2 instance. See Destroy Stack Resources in *Using Oracle WebLogic Server for OCI*.

> **WARNING:**
>
> Do not Delete a JRF Database Schema or Delete the Identity Cloud Service Resources in *Using Oracle WebLogic Server for OCI* as these resources are required in the upgraded instance.

- Run the following commands to remove the UI libraries. This stops the VNC server, removes the VNC server package, and removes the `Server with GUI` group packages.

  ```
  # Remove VNC server
  sudo su - oracle
  vncserver -list
  # Locate the X Display value and kill this - typically this is :1
  vncserver -kill :1
  # Remove the Linux service
  exit
  sudo bash
  systemctl stop vncserver@\:1.service
  systemctl disable vncserver@\:1.service
  # Uninstall package
  yum remove tigervnc-server -y
  # Remove GUI
  yum group remove "Server with GUI"
  ```

Roll Back an Upgrade

If the upgrade fails, you can roll back the upgrade to restore the WebLogic Server to its previous version.

1. Shutdown all WebLogic Server processes, including node manager, on each node. See Stop all WebLogic Server Processes.
2. Restore the backup of the Oracle Cloud Infrastructure database instance from object storage.

   See Restore a Database Using a Specific Backup from Object Storage in Oracle Cloud Infrastructure documentation.
3. Replace the domain contents in domain directory (`/u01/data/domains/*`) on each node with those you saved in `/u01/data/old_domain_version/`. See Perform Prerequisite Tasks.
4. Replace the binaries in `/u01/app/*` with those saved in `/tmp/old_mw_version`. See Perform Prerequisite Tasks.
5. Restart the node manager and the WebLogic Server processes. See Start the WebLogic Server Processes.