Oracle® Cloud

Using Oracle WebLogic Server for OCI

F84648-23
May 2025

Oracle Cloud Using Oracle WebLogic Server for OCI,

F84648-23

Primary Author: Oracle Corporation

Contents

Preface

Documentation Accessibility x

Diversity and Inclusion x

1 Get Started

About Oracle WebLogic Server for OCI 1-1

About the Components of Oracle WebLogic Server for OCI 1-2

- Oracle WebLogic Server 1-3
- OS Management Hub 1-4
- OCI Certificate Service 1-4
- JRF Domain 1-5
- Marketplace 1-5
- Resource Manager 1-5
- Compute 1-6
- Virtual Cloud Network 1-7
- Load Balancer 1-7
- Database 1-8
- Vault 1-12
- Identity 1-12
- Logging 1-14
- Application Performance Monitoring 1-14
- Application Performance Monitoring Dashboard 1-14
- Autoscaling 1-16

About Oracle WebLogic Server for OCI Versions and Retirement Policy 1-22

- Retirement Policy 1-23

Before You Begin 1-24

- Understand Service Requirements 1-25
- Create a Compartment 1-25
- Create Policies 1-25
  - User Group Policies 1-26
  - Dynamic Group Policies 1-29
  - Dynamic Group Policies for Autoscaling 1-34
- Create an Encryption Key 1-38

Create Secrets for Passwords 1-38
Create an SSH Key 1-39
Create a Virtual Cloud Network 1-39
Create a Private Endpoint 1-40
Create a Network Security Group 1-40
Create a Private Subnet for the Oracle WebLogic Server Nodes 1-43
Configure the Load Balancer 1-47
Create a Subnet for the Load Balancer 1-51
Create a Subnet for the Bastion Node 1-52
Create a Subnet for the Mount Target 1-53
Create a Database 1-53
Create a Confidential Application 1-55
Create an Application Performance Monitoring Domain 1-57
Create an Auth Token 1-57
Create a Certificate Authority 1-57
Create an OS Management Hub Profile 1-58
Validate Existing Network Setup 1-58
Using the Validation Script 1-59

2 Create a Stack

About Creating a Domain 2-1
Create a Domain 2-5
Launch a Stack 2-7
Specify Stack Information 2-7
Specify Stack Configuration 2-8
Configure Network Parameters 2-12
Configure WebLogic Domain Parameters 2-14
Configure WebLogic Domain Parameters for Secured Production Mode 2-15
Configure Database Parameters and VCN Peering 2-16
Configure WebLogic Compute Instance Parameters 2-21
Configure OS Management Hub Profile 2-22
WebLogic Cloud Steps 2-23
For Autoscaling section 2-33
Configure a Resource Manager Private Endpoint 2-34
Configure a Bastion 2-35
Configure a Load Balancer 2-36
Configure WebLogic Authentication 2-37
Configure Application Performance Monitoring 2-38
Configure Autoscaling 2-38
Configure File System 2-39
Configure Tags 2-41

Create the Domain Stack 2-41
Use Your New Domain 2-42
View the Cloud Resources for a Stack 2-42

3 Manage a Domain

About Managing a Domain 3-1
About the Default Ports 3-2
About the Security Checkup Tool 3-4
Access the Compute Instance 3-6
Access the WebLogic Server Administration Console 3-7
Access the WebLogic Console in a Public Subnet 3-7
Access the WebLogic Console in a Private Subnet 3-8
Access by Using the Bastion Instance 3-8
Access by Using the Bastion Service 3-11
Access the WebLogic Console Through a Load Balancer 3-12
Access the WebLogic Remote Console 3-14
Access the WebLogic Remote Console in a Public Subnet 3-14
Access the WebLogic Remote Console in a Private Subnet 3-15
Create SSH Tunneling Using the Bastion Instance 3-16
Create SSH Tunneling Using the Bastion Service 3-17
Configure Admin Server Provider using Proxy 3-18
Access the Fusion Middleware Control Console 3-19
Access the Fusion Middleware Control Console in a Public Subnet 3-20
Access the Fusion Middleware Control Console in a Private Subnet 3-20
Access the Sample Application 3-22
Access the Sample Application Through a Public Load Balancer 3-22
Access the Sample Application Through a Private Load Balancer 3-23
Access the Sample Application Without a Load Balancer 3-24
Access the Sample Application in a Private Subnet 3-25
Access the Sample Application Using Identity Cloud Service 3-26
Start and Stop a Domain 3-27
Add IDCS After Creating a Domain 3-28
Back Up and Restore a Domain 3-30
About Volume Backups 3-30
About Database Backups 3-31
Create a Backup 3-31
Restore a Backup 3-32
Restore to a Previous Domain 3-32
Recover from an Operating System Failure 3-34
Repair a Boot Volume 3-35
Add a Certificate to the Load Balancer 3-36

Enable OS Management Hub After Creating a Domain 3-37
Renew a Certificate When Secured Production Mode Is Enabled 3-37
Secure a Domain Using Identity Cloud Service 3-39
    Create WebLogic Administrator Groups 3-40
    Update WebLogic Administrator Roles 3-40
    Update Protected Application Resources 3-41
    Update Application Deployment Descriptors 3-43
Secure Web Services Using Identity Cloud Service 3-44
    Deploy a Sample Web Service Client Application 3-45
    Get Information About Identity Cloud Service 3-49
    Configure OAuth for the Web Services Provider 3-51
    Update the Confidential Application for the Web Services Provider 3-54
    Configure OAuth for the Web Services Client 3-54
    Update the Confidential Application for the Web Services Client 3-57
    Test the Sample Web Service Client 3-57
Integrate OPSS User and Group APIs with Identity Cloud Service 3-58
Upgrade the Oracle Identity Cloud Service App Gateway Version 3-59
Configure Session Persistence 3-65
    Enable Session Affinity or Sticky Sessions 3-66
Update the Password Secret OCID, Policy, and User Name 3-66
Update Password for the weblogic User 3-67
About Deleting Secrets and Policies 3-68
Update the Infrastructure Schema Password 3-69
    Update the Data Source Password if the WebLogic Administration Server Is Running 3-69
    Update the Data Source Password if the WebLogic Administration Server Is Not Running 3-72
Manage Data Sources 3-75
    Create a Data Source for an Oracle Autonomous Database 3-75
        Download the Oracle Autonomous Database Wallet 3-76
        Configure a Data Source for an Oracle Autonomous Database 3-78
    Create a Data Source for a DB System Database 3-80
        Verify the PDB Name 3-80
        Configure a Data Source for a DB System Database 3-81
Connect to a Domain Using Oracle JDeveloper 3-82
    Add an Ingress Rule 3-83
    Create a Network Channel 3-83
    Create a Connection to the Domain Through Oracle JDeveloper 3-84
Configure SSL for WebLogic Server 3-85
    Create Keystores and Certificates for WebLogic Server 3-85
    Add the Oracle Identity Cloud Service Certificate to the Trust Keystore 3-87
    Associate Keystores and SSL Certificate with WebLogic Server 3-88
    Configure Node Manager to Use the SSL Certificate 3-89
    Configure SSL for New Servers After Scaling Out 3-92

Replace Expiring Certificates 3-92
Upgrade a Domain 3-92
Backup the Database 3-93
Create a 12.2.1.4 Instance 3-93
Stop the WebLogic Server Processes on the Source 12.2.1.3 Instance 3-93
Replace the Domain on the Target 12.2.1.4 Instance with Cloning 3-94
Set up VNC Server 3-95
Perform Readiness Check 3-97
Upgrade Infrastructure Schemas 3-97
Reconfigure the Domain 3-99
Upgrade the Domain 3-100
Restart Servers 3-100
Post Upgrade 3-100
Roll Back Upgrade 3-101

4 Scale a Stack

Add or Remove WebLogic Server Nodes 4-1
Considerations for Scaling Out 4-2
Considerations for Scaling In 4-3
Add UCM WebLogic Server Node to a BYOL Stack 4-4
Manage Autoscaling Resources 4-5
Create Alarm Definitions 4-5
Update Alarm Definitions 4-6
Configure Function Application 4-7
Reenable Autoscaling for a Stack 4-8
Change the Shape of the Existing Compute Instances 4-9
Change Reserved Public IP Usage 4-10
Add a Load Balancer 4-10
Remove the Load Balancer 4-12

5 Clone a Stack

Move a WebLogic Server Instance 5-2
Clone a JRF or non-JRF Instance Using a Script 5-2
Copy a WebLogic Server Instance 5-5
Clone a non-JRF Instance 5-6
Scale Out the Cloned Instance 5-10

6 Delete a Stack

Delete a JRF Database Schema 6-1

Delete the Identity Cloud Service Resources 6-3
Delete Autoscaling Resources 6-4
Destroy Stack Resources 6-4
Delete the Stack 6-5
Delete the Database Security List 6-6
Delete the Database Network Security Group 6-7
Delete the Identity Cloud Service Resources Manually 6-7

7 Troubleshoot

Check Known Issues 7-1
Delete RCU Fails for Autonomous Database in 14.1.2.0 Domains 7-2
Clone Script Failed 7-2
Cleanup Resources of a Deleted Instance 7-2
Missing uuid for /u01/data or /u01/app on Reboot 7-16
Error in Mounting the Volume During Provisioning 7-17
Unable to Access the WebLogic Console From the Internet 7-18
Unable to Access the Sample Application Using the Load Balancer 7-19
Unable to Access the Fusion Middleware Control Console or the Enterprise Manager Console From the Internet 7-20
Load Balancer Does Not Send Cookie X-Oracle-BMC-LBS-Route 7-21
Reapply Fails for Load Balancer Configuration 7-21
Autoscaling Failed to Create Functions 7-22
Management Agents Are Not Deleted on Instance Termination 7-23
Enterprise Manager Console Is Not Loading 7-24
Scale Out Fails on the Administration Compute Instance 7-24
Warnings for Schema Password 7-25
Security Checkup Tool Warnings 7-25
Running python3 Command Fails 7-28
Unrecognized Arguments When Using the Patching Utility Tool 7-28
Get Additional Help and Contact Support 7-28
Unable to Fetch the Password Expiry Date for the OPSS User 7-30
OS Management Service Agent Not Enabled 7-31
OS Management Monitored Resources are Not Available 7-32
Stack Creation Failed 7-32
- View the Stack Log Files 7-37
- Modify the Stack Configuration 7-37
- Cannot Launch a Stack in Marketplace 7-38
- Cannot Determine Home Region 7-38
- Cannot Find Dynamic Group and Secrets Policy 7-38
- Maximum Number of Dynamic Groups Has Exceeded 7-39
- Unable to Get Secret Content or Decrypted Credential 7-39

Unable to Get Decrypted Credential When Creating a Stack in a Private Subnet 7-40
Failed to Download Oracle Autonomous Database Wallet 7-40
Failed to Validate DB Connectivity 7-41
Invalid or Overlapping Network CIDR 7-41
Job Is Still Running or Has Timed Out 7-42
Failed to Check Database Port Is Open for Exadata DB System 7-42

8 Patches

About the Patching Utility Tool 8-13
Patch Management Using the Patching Utility Tool 8-14
View Patching Tool Version 8-14
Configure Initial Setup 8-15
List Patches 8-17
List Pending Patches 8-19
View Patch Details 8-20
Download Patches 8-20
Apply Patches 8-21
Roll Back Patches 8-26
Upgrade Patching Tool 8-28
Patch Management Using the Patching Utility Tool on Windows 8-28

A License Information

B Configure SSL for a Domain

Create an HTTPS Listener for the Load Balancer B-1
Add a Certificate to the Load Balancer B-3
Update the App Gateway for HTTPS B-4

C Script Files

Script File to Create Policies for Autoscaling Functions Post Provisioning C-1
Script File to Delete Resources C-8

D Migrate Terraform Scripts

Preface

Using Oracle WebLogic Server for Oracle Cloud Infrastructure Oracle WebLogic Server for OCI explains how to create an Oracle WebLogic Server domain in Oracle Cloud Infrastructure Marketplace, and then administer the domain and deploy Java EE applications.

Topics:

- Documentation Accessibility
- Diversity and Inclusion

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `https://www.oracle.com/corporate/accessibility/`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `https://support.oracle.com/portal/` or visit `Oracle Accessibility Learning and Support` 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 Get Started

Learn about the architecture and features of Oracle WebLogic Server for Oracle Cloud Infrastructure (Oracle WebLogic Server for OCI), and perform any prerequisite tasks.

> **Note:**
>
> If you are using Oracle WebLogic Server for OCI release 22.4.1 or earlier, see Using Oracle WebLogic Server for OCI (Release 22.4.1 or earlier).

Topics

- About Oracle WebLogic Server for OCI
- About the Components of Oracle WebLogic Server for OCI
- About Oracle WebLogic Server for OCI Versions and Retirement Policy
- Before You Begin

About Oracle WebLogic Server for OCI

Use Oracle WebLogic Server for OCI to quickly create your Java Enterprise Edition (Java EE) application environment in Oracle Cloud Infrastructure, including an Oracle WebLogic Server domain, in a fraction of the time it would normally take on-premises.

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

You can track and monitor the progress of an Oracle WebLogic Server for OCI stack in Resource Manager. As your workload requirements change, you can use Resource Manager to scale the domain by adding or removing managed servers. A stack also provides a convenient method of deleting the cloud resources for a domain when you no longer require them.

After creating an Oracle WebLogic Server domain, you can administer the domain and deploy Java EE applications to it just like on-premises domains. Use standard Oracle WebLogic Server tools like the administration console 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.

Oracle WebLogic Server for OCI can also create a domain that includes the Java Required Files (JRF) components. A JRF-enabled domain:

- Supports the Oracle Application Development Framework (ADF)
- Can be administered and monitored using the Oracle Fusion Middleware Control console
- Connects to an existing database in Oracle Cloud Infrastructure

After creating a JRF-enabled domain with Oracle WebLogic Server for OCI, you can install additional Oracle Fusion Middleware products onto the domain if you have existing on-premise licenses for these products.

Oracle WebLogic Server for OCI supports these Oracle WebLogic Server editions:

- Oracle WebLogic Server Standard Edition
- Oracle WebLogic Server Enterprise Edition
- Oracle WebLogic Suite

Oracle WebLogic Server for OCI supports these Oracle WebLogic Server releases:

- Oracle WebLogic Server 14c (14.1.2.0) - See Understanding WebLogic Server
- Oracle WebLogic Server 14c (14.1.1.0) - See Understanding WebLogic Server
- Oracle WebLogic Server 12c (12.2.1.4) - See Understanding WebLogic Server

> **Note:**
>
> From release 21.4.3 (December 9, 2021) onwards, you cannot provision a domain in Oracle Oracle WebLogic Server for OCI for Oracle WebLogic server versions 11g (10.3.6.0) and 12c (12.2.1.3) from the Marketplace.

Oracle WebLogic Server for OCI supports these billing options:

- Universal Credits (also called UCM) - You are billed for the cost of the Oracle WebLogic Server Enterprise Edition or Oracle WebLogic Suite license (based on OCPUs per hour) in addition to the cost of the compute resources.
- Bring Your Own License (BYOL) - You reuse your existing on-premises Oracle WebLogic Server Standard Edition, Oracle WebLogic Server Enterprise Edition, or Oracle WebLogic Suite licenses in Oracle Cloud. You are billed only for the cost of the compute resources.

Oracle WebLogic Server Standard Edition is available only as BYOL.

Oracle WebLogic Server for OCI requires Oracle Cloud Infrastructure Vault (formerly known as Key Management) in order to securely store the passwords for your domains. See Oracle Cloud Infrastructure Vault FAQ.

About the Components of Oracle WebLogic Server for OCI

Learn about the Oracle Cloud Infrastructure components that comprise Oracle WebLogic Server for OCI.

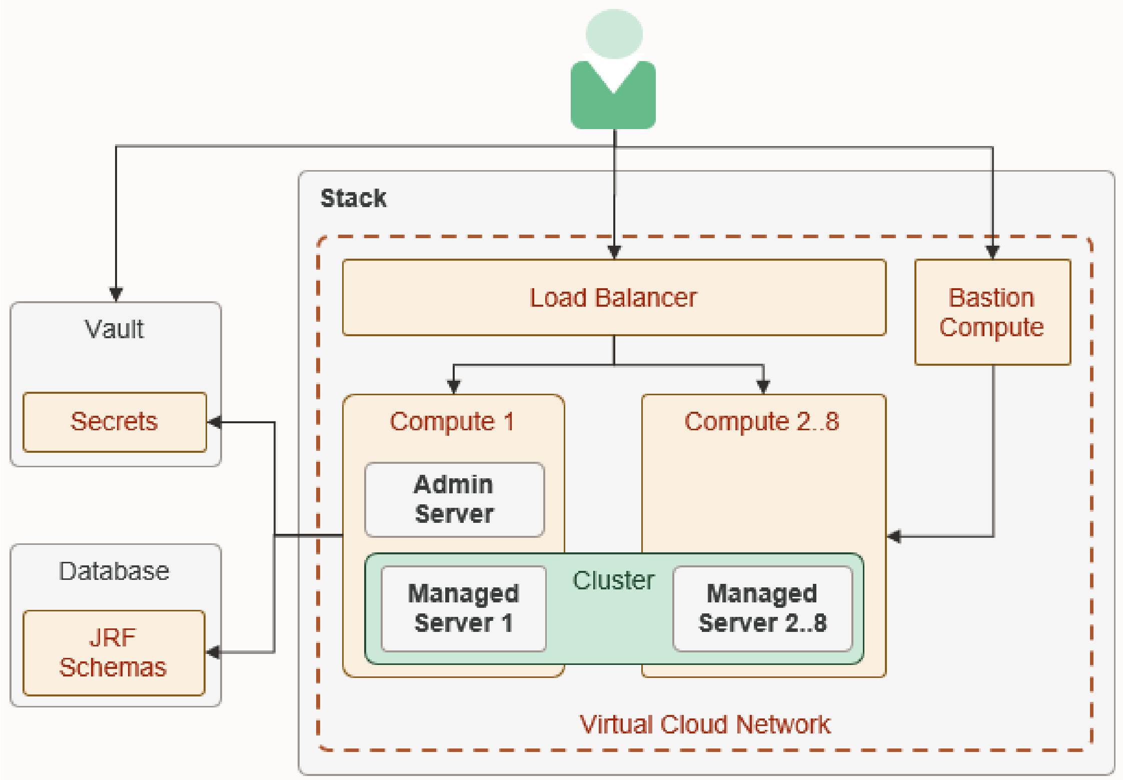
Topics

- Oracle WebLogic Server
- OCI Certificate Service
- JRF Domain
- Marketplace
- Resource Manager
- Compute
- Virtual Cloud Network

- Load Balancer
- Database
- Vault
- Identity
- Logging
- Application Performance Monitoring
- Application Performance Monitoring Dashboard
- Autoscaling

The following diagram illustrates the components of a typical Oracle WebLogic Server for OCI deployment.

Figure 1-1 Components of a Typical Deployment



Oracle WebLogic Server

An Oracle WebLogic Server domain consists of one administration server and one or more managed servers to host your Java application deployments.

Oracle WebLogic Server for OCI supports these Oracle WebLogic Server editions:

- Oracle WebLogic Server Standard Edition
- Oracle WebLogic Server Enterprise Edition
  - Includes all features and benefits of Oracle WebLogic Server Standard Edition

  - Includes clustering for high availability and scalability of Java resources and applications
  - Includes Oracle Java SE Advanced (Java Mission Control and Java Flight Recorder) for diagnosing problems in development and production
- Oracle WebLogic Suite
  - Includes all features and benefits of Oracle WebLogic Server Enterprise Edition
  - Includes Oracle Coherence for increased performance and scalability
  - Includes Active Gridlink for RAC for advanced database connectivity

Oracle WebLogic Server for OCI does not provision a cluster in domains running WebLogic Server Standard Edition.

Oracle WebLogic Server for OCI supports Oracle WebLogic Server 12.2.1.4.0, 14.1.1.0.0 and 14.1.2.0.0 releases. See About Oracle WebLogic Server for OCI for specific version information.

Oracle WebLogic Server for OCI can create these domain configurations:

- A basic domain that does not require a database configured as described in Lock Down WebLogic Server in the *Securing a Production Environment for Oracle WebLogic Server* requirements using a custom identity and custom trust keystore with certificates generated from the OCI Certificate Service.
- A basic domain that does not require a database.
- A domain that includes the Java Required Files (JRF) components and also requires a database.
- A domain that includes the Java Required Files (JRF) components, requires a database, and is configured as described in Lock Down WebLogic Server in the Securing a Production Environment for Oracle WebLogic Server requirements using a custom identity and custom trust keystore with certificates generated from the OCI Certificate Service (supported only in 14.1.2.0.0).

OS Management Hub

Oracle WebLogic Server for OCI uses OS Management Hub (OMH) to monitor and manage OS updates for Compute instances it creates.

The OS Management Hub Agent is enabled by the stack. As part of enabling the agent, the Compute instance is registered as a managed instance using a profile created by the stack or from an existing profile set for the stack. See Registering an Instance in the Oracle Cloud Infrastructure documentation.

See Listing Instances in the Oracle Cloud Infrastructure documentation for details on how to access the registered managed instances created by the stack and perform tasks such as updating OS software packages.

OCI Certificate Service

Oracle WebLogic Server for OCI uses OCI Certificate Authorities to sign certificates used for SSL configurations in domains created with Secured Production Mode enabled.

For more information, see OCI Certificate Service.

JRF Domain

The Java Required Files (JRF) option must be selected if your applications were developed with Oracle Application Development Framework (ADF) or use Oracle Web Services Manager (WSM) to access and secure REST and SOAP endpoints.

For more information about ADF, see Faster and Simpler Java-based Application Development. For more information about WSM, see the documentation for Oracle Web Services Manager.

> **Note:**
>
> These Fusion Middleware components are considered part of Oracle JRF: Oracle Application Development Framework, Oracle Fusion Middleware Audit Framework, Fabric Common, Infrastructure Security, Java Object Cache, JMX Framework, JPS, MDS, OJSP.Next, Oracle Web Services, Oracle Web Services Manager, Oracle TopLink, UCP, and XDK.

Marketplace

Oracle WebLogic Server for OCI is accessed as a collection of applications in the Oracle Cloud Infrastructure Marketplace.

Oracle Cloud Infrastructure Marketplace is an online store that's available in the Oracle Cloud Infrastructure console. When you launch an Oracle WebLogic Server for OCI application from Marketplace, it prompts you for some basic information, and then directs you to Resource Manager to complete the configuration of your Oracle WebLogic Server domain and supporting cloud resources.

Choose an Oracle WebLogic Server for OCI application that meets your functional and licensing requirements.

See Overview of Marketplace in the Oracle Cloud Infrastructure documentation.

Resource Manager

Oracle WebLogic Server for OCI uses Resource Manager in Oracle Cloud Infrastructure to provision the cloud instances and networks that support your Oracle WebLogic Server domain.

Resource Manager is an Oracle Cloud Infrastructure service that uses Terraform to provision, update, and destroy a collection of related cloud resources as a single unit called a stack. Resource Manager supports most resource types in Oracle Cloud Infrastructure, but a stack in Oracle WebLogic Server for OCI is comprised of these components:

- A compute instance running the administration server and the first managed server
- A compute instance for each additional managed server in the domain
- A bastion compute instance that provides administrative access to a domain on a private subnet
- A virtual cloud network (VCN), including subnets, route tables, and network security groups or security lists (optional)
- A load balancer (optional)

See Overview of Resource Manager in the Oracle Cloud Infrastructure documentation.

Compute

The servers that make up an Oracle WebLogic Server domain run on one or more Oracle Cloud Infrastructure Compute instances.

Oracle WebLogic Server for OCI creates Oracle Linux compute instances, and automatically installs the Oracle WebLogic Server software and creates the domain configuration on these instances.

> **Note:**
>
> The Oracle Linux version of the compute instances is 8.7.

You select the Oracle WebLogic Server edition and version you want to provision. If you plan to run Oracle WebLogic Server Standard Edition and require more than 4 nodes, then create a domain that runs Oracle WebLogic Server 12c Standard Edition. For all other editions and versions, the maximum is 8 nodes, which can be scaled out to 30 when you edit the domain.

You assign a shape to a domain, which determines the number of CPUs and the amount of memory allocated to each compute instance in the domain. If you create a domain in a private subnet, you can assign a different shape to the bastion compute instance.

> **NOT_SUPPORTED:**
>
> Some shapes might not be available in all regions.

You also assign a secure shell (SSH) public key to the compute instances for a domain. You can access and administer the operating system on the compute instances by using an SSH client and the matching private key.

In OCI, each Compute instance is assigned to an Availability Domain (AD). An availability domain represents a data center within an Oracle Cloud Infrastructure region. Each availability domain contains three fault domains. Oracle WebLogic Server for OCI automatically distributes the compute instances across these availability domains for high availability. If only a single AD is available, the VMs are spread across fault domains.

> **Note:**
>
> In a regional subnet, if you use shapes with service limits that are set for an availability domain, then for high availability the fault domains are used.

See Overview of the Compute Service and Regions and Availability Domains in the Oracle Cloud Infrastructure documentation.

Each Oracle WebLogic Server for OCI compute instance is comprised of three volumes:

- *Boot* volume - This volume contains the Linux operating system. To restore a boot volume backup, you must create a new compute instance.

> **Note:**
>
> Restoring from a boot volume is not recommended. It must be performed only as a last option.

- *Block* volume (*Domain* volume) - This volume has the contents of the `/u01/data` folder, including the domain configuration.
- *Block* volume (*Middleware* volume) - This volume has the contents of the `/u01/app` folder. The *Middleware* volume is introduced in instances created after release 20.3.3 (September 29, 2020).

See Overview of Block Volume Backups in the Oracle Cloud Infrastructure documentation.

Virtual Cloud Network

Oracle WebLogic Server for OCI assigns compute instances and load balancers to specific subnets in a virtual cloud network (VCN).

A VCN in Oracle Cloud Infrastructure covers a single, contiguous CIDR block of your choice. A subnet is a subdivision of a VCN that consists of a contiguous range of IP addresses that do not overlap with other subnets in the VCN. A VCN includes one or more subnets, route tables, gateways, DHCP options, and network security groups or security lists.

Oracle WebLogic Server for OCI can automatically create a VCN and subnets for a new Oracle WebLogic Server domain, or you can create your own VCN and subnets before creating a domain. By default subnets span an entire region in Oracle Cloud Infrastructure.

By default subnets are public. Any compute instances assigned to a private subnet can not be directly accessed from outside of Oracle Cloud. To enable the administration of compute instances in a private subnet, Oracle WebLogic Server for OCI can create a separate public subnet and bastion compute instance. Oracle WebLogic Server for OCI can also create a service gateway in a VCN so that compute instances can access other cloud services like Key Management and Oracle Autonomous Database, without using the public Internet.

If you already have an existing bastion to provide public access to the compute instances, or if you already have a VPN connection to your on-premise network, then you can delete the bastion instance created by Oracle WebLogic Server for OCI.

> **Note:**
>
> Configuring a bastion is optional.
>
> If you do not configure a bastion, no status is returned for provisioning. You must check the status of provisioning by connecting to each compute instance and confirm that the `/u01/provStartMarker` file exists with details found in the file `/u01/logs/provisioning.log` file. See Configure a Bastion.

See Overview of Networking in the Oracle Cloud Infrastructure documentation.

Load Balancer

Oracle Cloud Infrastructure Load Balancing routes requests it receives from clients to the managed servers in your Oracle WebLogic Server domain.

When you create a domain, Oracle WebLogic Server for OCI can automatically create a load balancer in Oracle Cloud Infrastructure and configure it to distribute traffic across the servers in your domain. Using a load balancer is recommended if your cluster size is greater than one.

By default, the load balancer is public. You can also provision a public load balancer with a reserved public IP. If you create a domain in a private subnet, then you can provision a public or private load balancer.

A private load balancer does not have a public IP address and cannot be accessed from outside of Oracle Cloud, unless you have configured a virtual private network (VPN) between your VCN and your on-premise data center.

A load balancer consists of primary and standby instances but it is accessible from a single public IP address. If the primary instance fails, traffic is automatically routed to the standby instance.

If your region includes multiple availability domains (AD), the load balancer supports two networking options:

- Assign the load balancer to one regional subnet
- Assign the load balancer to two AD-specific subnets

Session persistence is a method to direct all requests originating from a single logical client to a single backend server. By default, session persistence is enabled on the load balancer with the `Enable Load balancer cookie persistence` option, but you can update the load balancer after creating a domain.

See these topics in the Oracle Cloud Infrastructure documentation:

- Overview of Load Balancing
- VPN Connect
- Session Persistence

Database

To create an Oracle WebLogic Server domain that includes the Java Required Files (JRF) components, you must provide an existing database in Oracle Cloud Infrastructure.

> **Note:**
>
> - You cannot create an Oracle WebLogic Server domain that includes the Java Required Files (JRF) components for Oracle WebLogic Server 14.1.1.0 as this version does not support JRF.
> - Oracle Application Express (APEX) is not supported.

Choose one of these database options:

- Oracle Autonomous Database
  - Both dedicated and shared infrastructure autonomous database options are supported.
  - See Overview of the Autonomous Database in the Oracle Cloud Infrastructure documentation.

> **Note:**
>
> From 22.1.2 release (February 24, 2022) onwards, Free-Tier autonomous database is supported.

- Oracle Cloud Infrastructure Database
  - Bare metal, virtual machine (VM), and Exadata DB systems are supported.
  - For a 1-node VM DB system, you can use the fast provisioning option to create the database. Oracle WebLogic Server for OCI supports using Logical Volume Manager as the storage management software for a 1-node VM DB system.
  - For Oracle WebLogic Server 12.2.1.4.0 and 14.1.2.0.0, you can also specify a database connection string. This database connection string can be used only with existing VCN. To know the database connection string details, see Database Connect String for Database Version and Type in Configure Database Parameters and VCN Peering.
  - See Overview of the Database Service in the Oracle Cloud Infrastructure documentation.

Oracle WebLogic Server for OCI supports the same database versions and drivers as those for on-premise WebLogic Server installations. Refer to the following excel files (`xls`) at Oracle Fusion Middleware Supported System Configurations:

- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.2.0.0)
- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.1.0.0)
- System Requirements and Supported Platforms for Oracle Fusion Middleware 12c (12.2.1.4.0)

> **Note:**
>
> From release 21.4.3 (December 9, 2021) onwards, you cannot provision a domain in Oracle Oracle WebLogic Server for OCI for Oracle WebLogic server versions 11g (10.3.6.0) and 12c (12.2.1.3) from the Marketplace.

When you create a domain and associate it with an existing database, Oracle WebLogic Server for OCI does the following:

- Provisions the schemas to support the JRF components in the selected database
- Provisions data sources in the domain that provide connectivity to the selected database
- Deploys the JRF components and libraries to the domain

If you use an Oracle Cloud Infrastructure Database, the type of data sources that are created in the domain depend on the WebLogic Server edition and the number of database nodes.

- GridLink data sources for Oracle WebLogic Suite and a 2-node RAC DB system
- Multi data sources for Oracle WebLogic Server Enterprise Edition and a 2-node RAC DB system
- Generic data sources for all other configurations

See Understanding JDBC Resources in WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

> **Note:**
>
> If you use database connection string, then Oracle WebLogic Server for OCI creates a single instance datasource. However, you can update the data source for Oracle WebLogic Suite with Active GridLink data source and data source for Oracle WebLogic Server Enterprise Edition with multi data source. See Configuring Active GridLink Connection Pool Features and Configuring JDBC Multi Data Sources.

If you use a private subnet for WebLogic Server and Oracle Autonomous Database, Oracle WebLogic Server for OCI uses a service gateway in the VCN to access the database. The service gateway provides network access to cloud service without using the public Internet. See Access to Oracle Services: Service Gateway in the Oracle Cloud Infrastructure documentation.

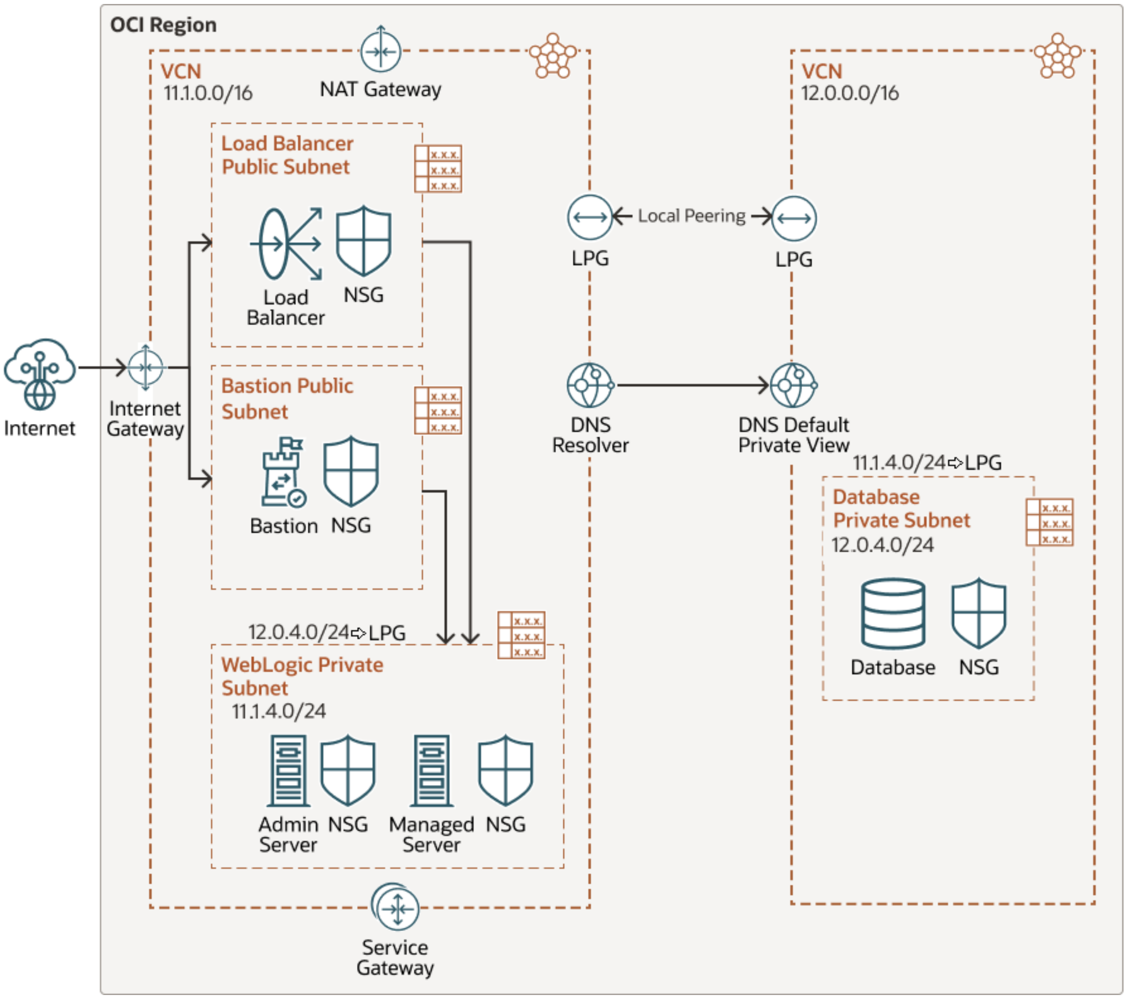
If your database is on a different VCN than the VCN you want to use for WebLogic Server, then Oracle WebLogic Server for OCI creates a connection between the two VCNs so that they are able to communicate.

This configuration is called local VCN peering and is illustrated by the following diagram.

The VCN peering is scoped to the subnet level as shown in the diagram. That is, the routes added to the WebLogic Server subnet forwards traffic to the database subnet only, and the routes added to the database subnet forwards traffic to the WebLogic Server subnet only. By adding default private view of the database VCN to the DNS resolver of the WebLogic Server VCN, all hosts in the database VCN are resolvable by the WebLogic Server VM.

The following diagram uses network security groups (NSGs) for the resources: load balancer, bastion, WebLogic Admin Server, and WebLogic Manager Server. However, instead of NSGs, you can configure security lists for these resources, if you use an existing network.

Figure 1-2   Local VCN Peering



The VCN peering for Oracle Cloud Infrastructure Database and Oracle Autonomous Database in Oracle WebLogic Server for OCI is as follows:

> **Note:**
>
> In case of Autonomous Database, the database must use a private endpoint for VCN peering.

- If you choose to create a virtual cloud network, Oracle WebLogic Server for OCI creates a Local Peering Gateway.
- If you use existing VCN and new subnets for an Oracle WebLogic Server domain, Oracle WebLogic Server for OCI creates a Local Peering Gateway, if your database VCN is on a different VCN than the WebLogic Server VCN.
- If you use an existing VCN and existing subnet, you must peer the VCNs manually before creating the stack if your database VCN is on a different VCN than the WebLogic Server VCN.
- If you use Oracle Cloud Infrastructure Database with connection string, you must peer the VCNs manually before creating the stack, if your database VCN is on a different VCN than the WebLogic Server VCN.
  To peer the VCNs manually, see Manual VCN Peering.

If you want multiple JRF-enabled domains to use the same database, then you can use local VCN peering. For this case, you must create the local peering gateway for each domain and modify the route table., and ensure that the CIDRs of the VCNs in the WebLogic subnet of the different stacks do not overlap.

To support VCN peering, you must perform the following prerequisite tasks before creating a stack:

- Create an local peering Gateway (LPG) in the database VCN.
- Add a route to the current route table of the database subnet, to direct traffic to the CIDR of the WebLogic subnet to the LPG.
- Open the database port to the WebLogic subnet CIDR.
- If you use an existing VCN and existing subnet, or existing VCN and new subnets, you must add the default private view of the database VCN to the DNS resolver of the existing VCN. See Add a DNS view to the DNS Resolver.

Local VCN peering cannot be used to connect WebLogic Server to an Oracle Cloud Infrastructure Database in a different region. See Local VCN Peering in the Oracle Cloud Infrastructure documentation.

Vault

Oracle Cloud Infrastructure Vault (formerly known as Key Management) enables you to manage sensitive information using vaults, keys, and secrets when creating an Oracle WebLogic Server domain.

A vault is a container for encryption keys and secrets. A standard vault is hosted on a hardware security module (HSM) partition with multiple tenants, and uses a more cost-efficient, key-based metric for billing purposes. A virtual private vault provides greater isolation and performance by allocating a dedicated partition on an HSM.

Secrets store credentials such as required passwords for a new domain. You use an encryption key in a vault to encrypt and import secret contents to the vault. Secret contents are based64-encoded. Oracle WebLogic Server for OCI uses the same key to retrieve and decrypt secrets when creating the domain.

Parameters for a new domain include:

- The secret for the password for the default Oracle WebLogic Server administrator
- The secret for the administrator password for an existing database, if you are creating a domain that includes the Java Required Files (JRF) components
- The secret for the client secret for an existing confidential application, if you are creating a domain that uses Oracle Identity Cloud Service for authentication

By default, Oracle WebLogic Server for OCI creates a dynamic group and root policy to allow compute instances to access your keys and secrets.

See:

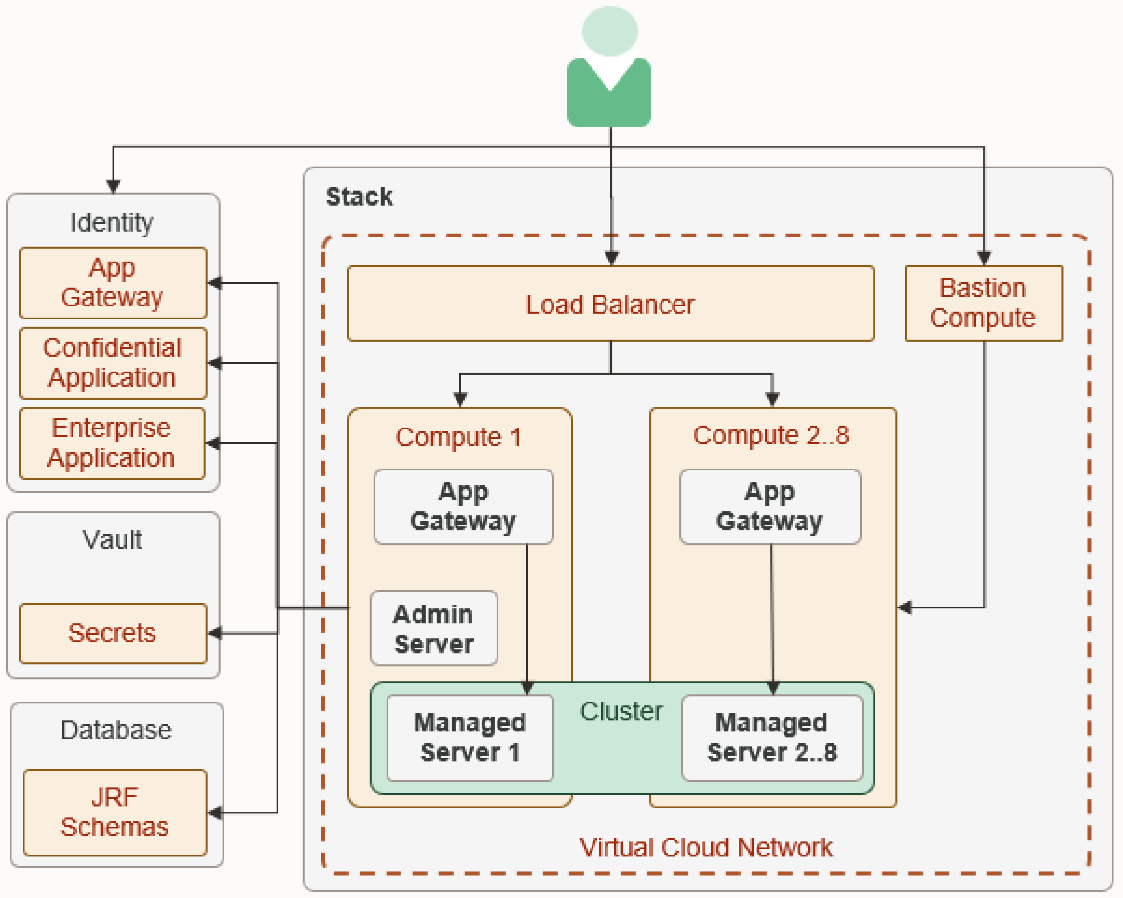
- Overview of Vault in the Oracle Cloud Infrastructure documentation
- Oracle Cloud Infrastructure Vault FAQ

Identity

Oracle Identity Cloud Service provides Oracle Cloud administrators with a central security platform to manage the relationships that users have with your applications.

By default, the Oracle WebLogic Server domain is configured to use the local WebLogic Server identity store to maintain administrators, application users, groups, and roles. These security elements are used to authenticate users, and to also authorize access to your applications and to tools like the WebLogic Server Administration Console.

Oracle WebLogic Server for OCI can configure a domain running WebLogic Server to use Oracle Identity Cloud Service for authentication. The following diagram illustrates this configuration.



This configuration is supported only for Oracle Cloud accounts that include Oracle Identity Cloud Service 19.2.1 or later.

Oracle WebLogic Server for OCI configures an App Gateway in Oracle Identity Cloud Service. It also provisions each compute instance in the domain with the App Gateway software appliance. The App Gateway acts as a reverse proxy, intercepts HTTP requests to the domain, and ensures that the users are authenticated with Oracle Identity Cloud Service.

Oracle WebLogic Server for OCI creates two security applications in Oracle Identity Cloud Service to support the domain. A confidential application allows the domain to securely access the identity provider using the OAuth protocol. An enterprise application defines the URLs that are protected by the App Gateway.

If you enable integration with Oracle Identity Cloud Service for a domain, then you must also enable a load balancer for the domain.

See About Oracle Identity Cloud Service Concepts in *Administering Oracle Identity Cloud Service*.

Logging

Oracle WebLogic Server for OCI uses the Logging Service to view the server logs such as errors or warnings in the Oracle Cloud Infrastructure console.

The logging resources, Custom Logs, Log Groups, and Unified Agent Configuration, are created during stack provisioning. See Logging Concepts in the Oracle Cloud Infrastructure documentation.

Application Performance Monitoring

Application Performance Monitoring service monitors the performance of administration and managed servers in Oracle WebLogic Server for OCI domain and displays the server metrics in the Application Performance Monitoring dashboard. It also helps to understand workloads and alerts the user for any issues in the logs.

See About Application Performance Monitoring in the Oracle Cloud Infrastructure documentation.

Application Performance Monitoring Dashboard

You can use the Oracle-defined Application Performance Monitoring (APM) dashboard for Oracle WebLogic Server for OCI to view the WebLogic metrics that are exported using APM Java Agent. This Oracle-defined Application Performance Monitoring (APM) dashboard is called **WebLogic Domains**.

You must access the Dashboards page to view the **WebLogic Domains** dashboard. To access the dashboard, see Access Dasboards.

On the Dashboards page, when you click **WebLogic Domains**, you can view the dashboard and the following filter options:

- Compartment - The compartment where the APM domain resides.
- APM Domain - The APM domain where your metrics is located.
- WebLogic Domain - The name of the WebLogic domain for which you can view the metrics. This option lists all the domains of the selected APM domain. The servers that belongs to the selected WebLogic domain are displayed in the **WebLogic Server** filter.

  > **Note:**
  >
  > By default, Oracle WebLogic Server for OCI supports only one domain. However, you can create additional domains.

- WebLogic Server - The name of the server for which you can view the metrics. This option lists only the servers that belong to the selected WebLogic domain. But, if **All servers** option is selected, the widgets shows metrics for all the servers.

The **WebLogic Domains** dashboard page includes default widgets that can be used to monitor metrics such as CPU load, heap percentage, and stuck threads.

But, if you want to add widgets to the **WebLogic Domains** dashboard, you must first create a custom dashboard by creating a copy of the **WebLogic Domains** dashboard, and then add widgets to your newly created custom dashboard. See Customize an Oracle-defined Dashboard.

The following table lists the widgets that are available by default in the **WebLogic Domains** dashboard.

Table 1-1 Widgets Displayed by Default in the WebLogic Domains Dashboard

Widget Name	Description	Type
WebLogic CPU load (System and Process)	Displays the process and system CPU load (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric of all selected servers.	Line chart, aggregated
WebLogic Free Heap percentage	Displays the free heap percentage for the selected server. If multiple servers are selected, the widget displays the mean of the load of all selected servers.	Line chart, aggregated
WebLogic Free Heap percentage (by Server)	Displays the free heap percentage for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic Process CPU load (by Server)	Displays the process CPU for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic Stuck threads (by Server)	Displays the stuck threads for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic Queue Length (by Server)	Displays the queue length for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server

The following table lists the widgets that you can add to your newly created custom dashboard.

Table 1-2 Widgets for Your Custom Dashboard

Widget Name	Description	Type
WebLogic GC Young Time (by Server)	Displays the garbage collection time for the young generation for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic GC Old Time (by Server)	Displays the garbage collection time for the old generation for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic Servers	Displays a table with the following information for the selected servers: • Heap Free Percentage Average • Last WebLogic Server State • Process CPU Load Average • System CPU Load Average	Table
WebLogic Execute Thread Total Count (by Server)	Displays the `WeblogicThreadPoolExecuteThreadTotalCount` for the selected server. If multiple servers are selected, the widget displays one line for each selected server	Line chart, grouped by server
WebLogic Heap Committed (by Server)	Displays the amount of heap memory committed for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server

Table 1-2 (Cont.) Widgets for Your Custom Dashboard

Widget Name	Description	Type
WebLogic Heap Used (by Server)	Displays the amount of heap memory used for the selected server. If multiple servers are selected, the widget displays one line for each selected server.	Line chart, grouped by server
WebLogic Threadpool	Displays the number of stuck threads, execute thread total count and queue length (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric for all selected servers.	Line chart, aggregated
WebLogic GC Time	Displays the garbage collection total time for young and old generation (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric for all selected servers.	Line chart, aggregated
WebLogic GC Count	Displays the garbage collection total count for young and old generation (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric for all selected servers.	Line chart, aggregated
WebLogic Heap Usage	Displays the amount of heap used memory and heap committed memory (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric for all selected servers.	Line chart, aggregated
WebLogic Non Heap Usage	Displays the amount of non-heap used memory and non-heap committed memory (one line per metric) for the selected server. If multiple servers are selected, the widget displays the mean of each metric for all selected servers	Line chart, aggregated
WebLogic Last Known Status	Displays the last known status of the selected server (e.g. Running, admin).	SingleValue
WebLogic Open Sockets	Displays the last known value of open sockets of the selected server.	SingleValue
WebLogic Active Threads	Displays the last known value for active threads of the selected server, based on the formula: `"( WeblogicThreadPoolExecuteThreadTotalCountLast.value - (WeblogicThreadPoolStandbyCountLast.value + WeblogicThreadPoolExecuteThreadIdleCountLast.value + WeblogicThreadPoolStuckCountLast.value))"`	SingleValue

Autoscaling

Oracle WebLogic Server for OCI uses autoscaling to automatically manage the size and lifecycle state of your WebLogic compute instances.

Before you enable autoscaling:

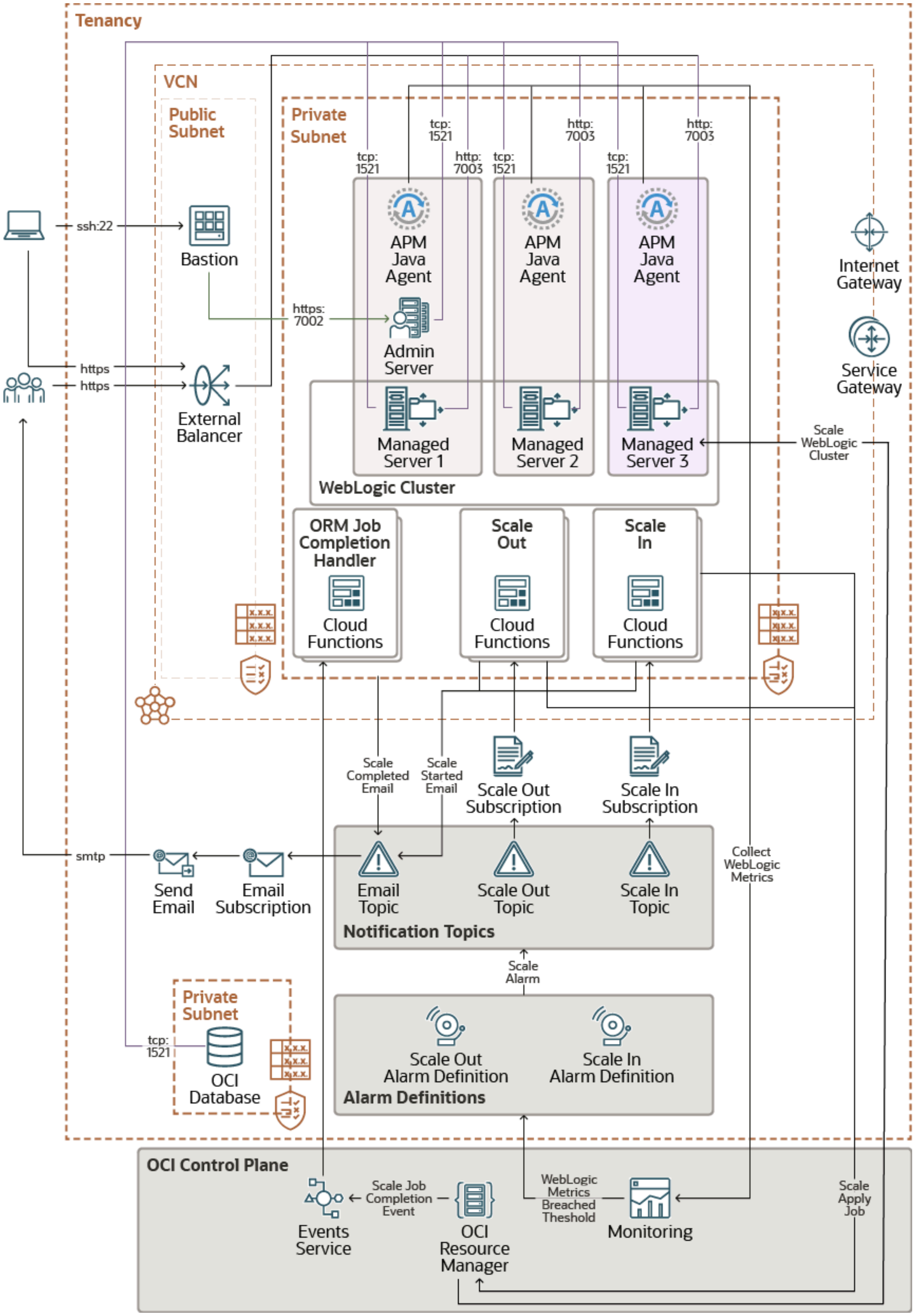
- You must create an Application Performance Monitoring domain or use an existing APM domain. See Create an APM Domain in the Oracle Cloud Infrastructure documentation.

> **Note:**
>
> Application Performance Monitoring always free domain is not supported for autoscaling.

- Create a new auth token for a user with access to Oracle Cloud Infrastructure Registry, or use an existing auth token. See Managing User Credentials in the Oracle Cloud Infrastructure documentation.
- Create policies for an Oracle Cloud Infrastructure user who is not an administrator. See Dynamic Group Policies for Autoscaling.

The following diagram illustrates the network configuration for autoscaling in Oracle WebLogic Server for OCI deployment.

Figure 1-3   Network Configuration

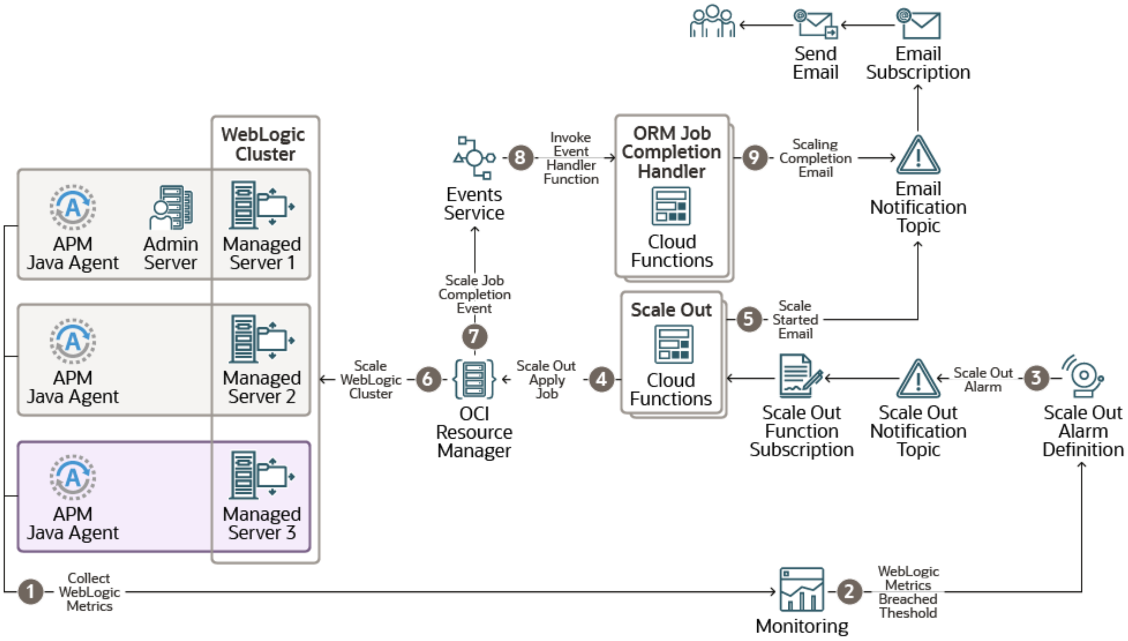


During stack creation for a domain, Oracle WebLogic Server for OCI allows you to configure metric-based autoscaling based on WebLogic Monitoring metrics.

> **Note:**
>
> You can configure autoscaling for Oracle WebLogic Server Enterprise Edition and Oracle WebLogic Suite only.

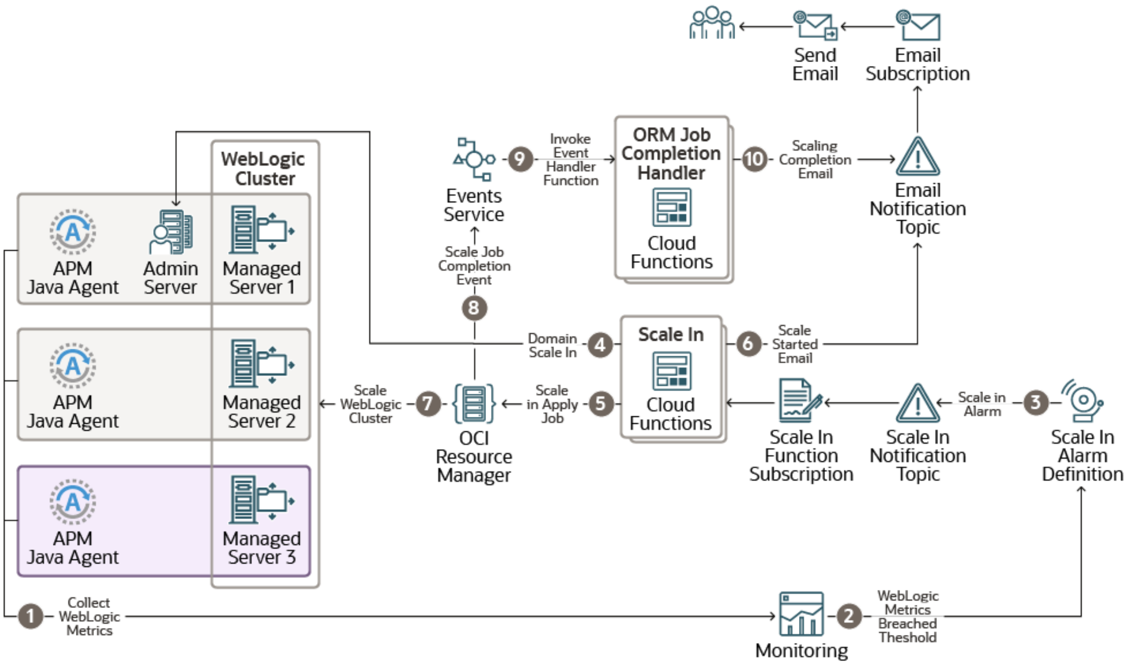
The following diagram illustrates Scale Out flow in Oracle WebLogic Server for OCI deployment.

Figure 1-4 Scale Out Flow



The following diagram illustrates Scale In flow in Oracle WebLogic Server for OCI deployment.

Figure 1-5   Scale In Flow



The resources created for autoscaling are:

- Application Performance Monitoring Agent - Application Performance Monitoring agent is used to collect WebLogic monitoring metrics and enables probes for tracing.
- WebLogic Monitoring metrics - WebLogic Monitoring metrics are collected by the Application Performance Monitoring Agent.
- Alarms Definition - Alarms Definitions are rules defined for metric-based autoscaling. They include the alarm query that evaluates the alarm using Monitoring Query Language (MQL) expression and the notification destination to send messages when the alarm is in the firing state, in addition to other alarm properties. See Managing Alarms in the Oracle Cloud Infrastructure documentation.
  The following Alarm Definitions are created during provisioning:
  - Scale Out Alarm Definition
  - Scale In Alarm Definition

  If the **OCI Policies** check box is selected during provisioning, alarms are enabled.

  If the **OCI Policies** check box is not selected during provisioning, alarms are disabled. You must create dynamic group and policies, and then enable alarms from the Oracle Cloud Infrastructure console post provisioning.

  To create dynamic group and policies, see Dynamic Group Policies for Autoscaling and to enable alarms, see To enable an alarm in Oracle Cloud Infrastructure documentation.
- Notification topic - A topic is a communication channel for sending messages to its subscriptions. See Creating a Topic and Creating a Subscription in the Oracle Cloud Infrastructure documentation.
  The following notification topics are created during provisioning:
  - Scale Out Topic
  - Scale In Topic

  - Email Topic
- Notification subscriptions - The following notification subscriptions are created during provisioning:
  - Scale Out Function Subscription for Scale Out Topic
  - Scale In Function Subscription for Scale In Topic
  - Email Subscription for Email Topic

  See Managing Topics and Managing Subscriptions in the Oracle Cloud Infrastructure documentation.
- Function Application - Function Application is logical group of related functions, under which Scale In Function, Scale Out Function, and Resource Manager Job Completion Handler Function are grouped. See Overview of Functions in the Oracle Cloud Infrastructure documentation.
- Oracle Cloud Infrastructure Registry (OCIR) - OCIR is created using Terraform in root compartment and stores the docker image of the Functions. See Overview of Container Registry in the Oracle Cloud Infrastructure documentation.
  The following repositories are created during provisioning:
  - `<`*`service_prefix_name`*`>_autoscaling_function_repo/scaleout`
  - `<`*`service_prefix_name`*`>_autoscaling_function_repo/scalein`
  - `<`*`service_prefix_name`*`>_autoscaling_function_repo/orm_job_completion_handler`
- Functions - Functions trigger scale in and scale out operations when an alarm event is created upon reaching a threshold for the monitored metrics. See Overview of Functions in the Oracle Cloud Infrastructure documentation.
  The following functions are created during provisioning:
  - Scale Out Function
  - Scale In Function
  - Resource Manager Job Completion Handler Function
- Event Rule - The Event Rule defines event matching rule for Resource Manager `Job Create - End` event in the stack compartment. See Overview of Events in the Oracle Cloud Infrastructure documentation.
- Log Group and Logs - The log group contains one or more logs. The log resources created during provisioning are:
  - Function Application Log - This contains logs from all functions within the function application.
  - Event Rule Invoke Log - This contain logs for Resource Manager Job Completion event rule invocations.

  See Logging Overview in the Oracle Cloud Infrastructure documentation.

In Oracle WebLogic Server for OCI, you can apply metric-based autoscaling to WebLogic server instances. You can select a performance metric and set thresholds that this performance metric must reach to trigger an autoscaling event. The performance metrics in metric-based autoscaling are:

- CPU Load - The current CPU load of the JVM process.
- Used Heap Percent - The percentage of JVM heap that is used.
- Queue Length - The number of pending requests in the priority queue.

- Stuck Threads - The number of stuck threads in the thread pool.

See Metrics in the Oracle Cloud Infrastructure documentation.

After you select a performance metric and define the alarm minimum and maximum thresholds for the selected metric, two alarm definitions are triggered, Scale Out Alarm Definition and Scale In Alarm Definition, and a notification is sent to the Scale Out Topic or the Scale In Topic. Then, the Scale Out Function or the Scale In Function is invoked (based on the topic that receives the notification), and this function scales out or scales in the instances. Messages are sent out as emails to the notification address when the instance is scaled out or scaled in, and also after the scaling is complete with the job status.

See Notifications in the Oracle Cloud Infrastructure documentation.

The limitations with Autoscaling are:

- Autoscaling is not supported with Terraform CLI-based Oracle WebLogic Server for OCI provisioning.
- Autoscaling is not supported for cloning in this release.
- Autoscaling can be selected during provisioning only and cannot be enabled or disabled on reapply.
- Autoscaling is disabled if you select **Do Not Update Domain Configuration for Scale Out** during provisioning as for autoscaling, the domain update must be performed during provisioning.
- Autoscaling does not work as expected if bastion is not configured when WebLogic instances are in private subnet.
  For example, during provisioning, if you configure the bastion and you opt to manually update the domain configuration by not selecting **Do Not Update Domain Configuration for Scale Out**, even if the domain update fails, scale out stack apply job succeeds as it does not detect the domain update failure. So the stacks are available for autoscaling, and successive alarms trigger a scale out and thus end up scaling out to maximum node count of 30 nodes.

About Oracle WebLogic Server for OCI Versions and Retirement Policy

Oracle WebLogic Server for OCI versions adopt the standard Oracle multiple digits system for version numbering.

A version number for Oracle WebLogic Server for OCI contains six decimal places in the form:

```
WLS n.n.n.n.yymmdd.n
```

For example:

```
WLS 12.2.1.4.190716.01
WLS 14.1.1.0.190716.01
WLS 14.1.2.0.250121.01
```

The first 4 decimal places together describe the base version of a WebLogic Server release, such as:

`12.2.1.4` for Oracle WebLogic Server 12c Release 3

`14.1.1.0` for Oracle WebLogic Server 14c Release 1

`14.1.2.0` for Oracle WebLogic Server 14c Release 2

The 5th decimal place is a quarterly patch set release date. For example, `250121` is the January 2025 patch set.

For patch set update (PSU) naming purposes, releases in a quarterly patch set for Oracle WebLogic Server for OCI are named uniquely by the 6th decimal place. The first release is `01`. The number is incremented by one for every patch or every update of that patch set, including one-off patches and updates to the VM image, the WebLogic scripts placed on the VM images, the Terraform scripts, and the Resource Manager schema.

An Oracle WebLogic Server for OCI PSU is created from the Critical Patch Updates (CPUs) of several products, namely, Oracle WebLogic Server, Oracle JDeveloper, Oracle Java Development Kit, Oracle Platform Security Services and Oracle Web Services Manager. While the Oracle WebLogic Server for OCI quarterly patch set release date (*yymmdd*) is mainly derived from a WebLogic Server PSU, the latest PSUs of all those products are included. See Patches.

Retirement Policy

Oracle WebLogic Server for OCI PSUs do not retire based on any fixed time range. Instead, for each WebLogic Server release (12c or 14c), only the last two patch sets (identified by *yymmdd*) are retained in a quarter.

> **Note:**
>
> The Oracle WebLogic Server for OCI image of a version is retained as-is, including the WebLogic Server binaries, the VM image contents, and any bugs that were inherent to the WebLogic scripts on the VM.

For example, suppose at the end of December 2019 there are two July PSUs and two October PSUs for WebLogic Server release 12.2.1.4:

```
12.2.1.4.190716.01
12.2.1.4.190716.02
12.2.1.4.191016.01
12.2.1.4.191016.02
```

When the 2020 January PSU is released (say, `12.2.1.4.200116.01`), the July PSUs are retired and only the PSUs for October 2019 and January 2020 are retained:

```
12.2.1.4.191016.01
12.2.1.4.191016.02
12.2.1.4.200116.01
```

Note the following about all retained Oracle WebLogic Server for OCI versions:

- Bugs fixed in a later version are not back ported into an earlier version.
- A version may be pulled without notice if there is a very serious security or functional issue.

See Known Issues in Oracle WebLogic Server for Oracle Cloud Infrastructure for known problems in a retained version and how to work around them.

Before You Begin

Before you create a domain with Oracle WebLogic Server for OCI, you must complete one or more prerequisite tasks.

Some tasks are required for any type of Oracle WebLogic Server domain that you create with Oracle WebLogic Server for OCI. Other tasks are optional or only applicable for specific domain configurations.

> **Note:**
>
> Before you provision an instance, you can estimate the cost of the resources and services to use in your instance. See Oracle Cloud Cost Estimator.

Required Tasks

- Understand Service Requirements
- Create a Compartment
- Create Policies
- Create an Encryption Key
- Create Secrets for Passwords
- Create an SSH Key

Optional Tasks

- Create a Virtual Cloud Network
- Create a Private Endpoint
- Create a Network Security Group
- Create a Private Subnet for the Oracle WebLogic Server Nodes
- Configure the Load Balancer
- Create a Subnet for the Load Balancer
- Create a Subnet for the Bastion Node
- Create a Subnet for the Mount Target
- Create a Database
- Create a Confidential Application
- Create an Application Performance Monitoring Domain
- Create an Auth Token
- Create a Certificate Authority
- Create an OS Management Hub Profile
- Validate Existing Network Setup

Understand Service Requirements

You require access to several services in order to use Oracle WebLogic Server for OCI.

- Identity and Access Management (IAM)
- Compute, Network, Block Storage
- Vault, Key, Secret
- Resource Manager
- Load Balancing (optional)
- Database (optional)
- Tagging (optional)

Check the service limits for these components in your Oracle Cloud Infrastructure tenancy and, if necessary, request a service limit increase.

In Oracle Cloud Infrastructure Vault (formerly known as Key Management), a standard vault is hosted on a hardware security module (HSM) partition with multiple tenants, and uses a more cost-efficient, key-based metric for billing purposes. A virtual private vault provides greater isolation and performance by allocating a dedicated partition on an HSM. Each type of vault has a separate service limit in your Oracle Cloud Infrastructure tenancy. The limit for secrets spans all vaults.

See:

- Service Limits in the Oracle Cloud Infrastructure documentation
- Oracle Cloud Infrastructure Vault FAQ

Create a Compartment

Create compartments for your Oracle WebLogic Server for OCI resources, or use existing compartments.

This task is typically performed by an administrator.

When you create a domain with Oracle WebLogic Server for OCI, by default the compute instances, networks, and load balancer are all created within a single compartment. You can, however, choose to use two compartments, one compartment just for the compute instances (WebLogic Server and bastion nodes), and another compartment for all the network resources that are created for the domain (including load balancer, virtual cloud network, subnets, route tables, gateways, and network security groups or security lists).

See Managing Compartments in the Oracle Cloud Infrastructure documentation.

Create Policies

Access to Oracle Cloud Infrastructure is controlled through policies.

There are two major groupings of policies that are required by Oracle WebLogic Server for OCI:

- User Group Policies
- Dynamic Group Policies and Dynamic Group Policies for Autoscaling

User Group Policies

Oracle WebLogic Server for OCI runs terraform scripts on behalf of the user who is running the OCI stack. Therefore, when running the stack you must have certain OCI policies set up to create the required OCI resources.

Required Policies

Unless you are an OCI administrator for a tenancy, the following policies must be set up for the OCI Identity Group that you are a member of. You must set up these user group policies prior to creating a Oracle WebLogic Server for OCI instance.

In the following listed policies:

- `MyGroup` is the group name of the user who will create the OCI stack.
- `MyCompartment` and `MySecretCompartment` is name of the compartment in which the resources will be created or updated.

Table 1-3 Required User Group Policies

Policy	Description	Policy Location
`Allow group` *`MyGroup`* `to inspect instance-image in compartment` *`MyCompartment`*	To use the WebLogic custom images from Marketplace	Stack Compartment
`Allow group` *`MyGroup`* `to use app-catalog-listing in compartment` *`MyCompartment`*	To use the Marketplace applications	Stack Compartment
`Allow group` *`MyGroup`* `to manage instance-family in compartment` *`MyCompartment`*	To create Compute Instances	Stack Compartment
`Allow group` *`MyGroup`* `to manage volume-family in compartment` *`MyCompartment`*	To create Block Volumes	Stack Compartment
`Allow group` *`MyGroup`* `to manage orm-family in compartment` *`MyCompartment`*	To create stacks	Stack Compartment
`Allow group` *`MyGroup`* `to inspect secrets in compartment` *`MySecretCompartment`*	To display the list of secrets in the UI	Secret Compartment
`Allow group` *`MyGroup`* `to inspect limits in tenancy`	To determine if resources are available in various compartments	Root Compartment
`Allow group` *`MyGroup`* `to inspect tenancies in tenancy`	To locate the home region for the tenancy	Root Compartment
`Allow group MyGroup to use virtual-network-family in compartment MyNetworkCompartment`	To set the networking configuration when creating compute instances.	Network Compartment

Stack Selection Policies

Unless you are an OCI administrator for your tenancy, there are user group policies that must be set up based on the selections that you make in the Oracle Cloud Infrastructure console or the OCI CLI.

The following table list the policies required for specific selections you intend to make.

In the following listed policies:

- `MyGroup` is the group name of the user who will create the OCI stack.
- `MyCompartment`, `MyNetworkCompartment`, and `MyDBCompartment` is name of the compartment in which the resources will be created or updated.

Table 1-4 User Group Policies Based on Stack Selection

Selection	Policy	Description	Policy Location
Enable OS Management Hub	Allow group *MyGroup* to manage osmh-family in compartment *MyProfileCompartment*	To create OS Management Hub (OMH) profiles and associate OMH software sources to new or existing profiles.	Profile Compartment
	Allow group *MyGroup* to read osmh-software-sources in tenancy	To locate OS Management Hub software sources which are typically defined in the root compartment.	Root Compartment
	Allow group *MyGroup* to read osmh-profiles in tenancy	To associate OS Management Hub software sources to profiles.	Root Compartment
Network • **Create a new VCN** • **Create a New Subnet**	Allow group *MyGroup* to manage virtual-network-family in compartment *MyNetworkCompartment*	To create VCNs and subnets	Network Compartment
Load Balancer • **Add a Load Balancer** • **Enable Authentication Using Identity Cloud Service**	Allow group *MyGroup* to manage load-balancers in compartment *MyNetworkCompartment*	To create a load balancer	Network Compartment
Enable Secured Production Mode	Allow group *MyGroup* to read certificate-authorities in tenancy	To identify the compartment where the certificate authority resides.	Root Compartment
Enable Application Performance Monitoring	Allow group *MyGroup* to read metrics in compartment *MyCompartment* where target.metrics.namespace='oracle_apm_monitoring'	To view and retrieve monitoring metrics	APM Domain Compartment
	Allow group *MyGroup* to read apm-domains in compartment MyAPMDomainCompartment	To list the APM domains in the UI and to identify the compartment associated with the selected APM domain.	APM Domain Compartment
Enable Exporting Logs to OCI Logging Service	Allow group *MyGroup* to manage log-groups in compartment *MyCompartment*	To create log groups	Stack Compartment
	Allow group *MyGroup* to use log-content in compartment *MyCompartment*	To view the logs	Stack Compartment
	Allow group *MyGroup* to manage unified-configuration in compartment *MyCompartment*	To provision agent configurations	Stack Compartment
Enable Autoscaling	Allow group *MyGroup* to read objectstorage-namespaces in tenancy	To read objectstorage namespace for tenancy	Root Compartment
	Allow group *MyGroup* to manage repos in tenancy	To create repository and push images to repository	Root Compartment (this can be moved to compartment if we use repos from compartment)

Table 1-4 (Cont.) User Group Policies Based on Stack Selection

Selection	Policy	Description	Policy Location
	Allow group *MyGroup* to manage functions-family in compartment id *MyCompartmentOCID*	To deploy and manage OCI Functions and function application	Stack Compartment
	Allow group *MyGroup* to manage ons-subscriptions in compartment id *MyCompartmentOCID*	To create and manage topic subscriptions	Stack Compartment
	Allow group *MyGroup* to manage ons-topics in compartment id *MyCompartmentOCID*	To create and manage topics	Stack Compartment
	Allow group *MyGroup* to manage alarms in compartment id *MyCompartmentOCID*	To create and manage Alarms	Stack Compartment
	Allow group *MyGroup* to manage cloudevents-rules in compartment id *MyCompartmentOCID*	To create cloud event rules	Stack Compartment
	Allow group *MyGroup* to use cloud-shell in tenancy	Required for deleting autoscaling resources before stack deletion	Root Compartment
OCI Policies	Allow group *MyGroup* to manage dynamic-groups in tenancy	To create dynamic groups. See Dynamic Group Policies.	Root Compartment
	Allow group *MyGroup* to manage policies in tenancy	To create policies in the root compartment	Root Compartment
Use Resource Manager Private Endpoint	Allow group *MyGroup* to manage orm-family in compartment *MyNetworkCompartment*	To create private endpoints	Network Compartment
	Allow group *MyGroup* to manage virtual-network-family in compartment *MyNetworkCompartment*	To create private endpoints	Network Compartment
Provision with JRF and **Autonomous Transaction Database Processing**	Allow group *MyGroup* to inspect autonomous-transaction-processing-family in compartment *MyDBCompartment*	To display the list of available Autonomous Databases	DB Compartment
Provision with JRF and **Database System**	Allow group *MyGroup* to inspect database-family in compartment *MyDBCompartment*	To display the list of databases available	DB Compartment
	Allow group *MyGroup* to manage virtual-network-family in compartment *MyDBCompartment*	To create a security list to allow ingress from WebLogic Server	DB Compartment
Add File System Storage	Allow group *MyGroup* to manage mount-targets in compartment *MyCompartment*	To create mount target and associate a file system with an existing mount target	Mount Target Compartment
	Allow group *MyGroup* to manage file-systems in compartment *MyCompartment*	To create a shared file system	FSS Compartment
	Allow group *MyGroup* to manage export-sets in compartment *MyCompartment*	To create export-sets	FSS Compartment

Table 1-4 (Cont.) User Group Policies Based on Stack Selection

Selection	Policy	Description	Policy Location
Add File System Storage and **Existing File System**	`Allow group` *`MyGroup`* `to read mount-targets in compartment` *`MyCompartment`*	To read existing mount targets for mounting existing FSS	Mount target Compartment
Add File System Storage and **Existing Mount target**	`Allow group` *`MyGroup`* `to use mount-targets in compartment` *`MyCompartment`*	To read existing mount targets for mounting new FSS	Mount Target Compartment
	`Allow group` *`MyGroup`* `to manage file-systems in compartment` *`MyCompartment`*	To create a shared file system	FSS Compartment
	`Allow group` *`MyGroup`* `to manage export-sets in compartment` *`MyCompartment`*	To create FSS export-sets for mount target	FSS Compartment

Dynamic Group Policies

When Oracle WebLogic Server for OCI starts up a Compute instance, certain scripts make OCI API calls. The Compute instances are granted the privileges by defining an Oracle Cloud Infrastructure dynamic group that includes the Compute instance and assigning that dynamic group to a set of policies. You can choose to create these policies prior to creating the OCI stack or you can choose to let the stack creation terraform scripts create these policies for you by selecting the `OCI Policies` option.

If you want the Oracle WebLogic Server for OCI stack to create the dynamic groups and policies for you, then be sure to have your OCI administrator construct the following user group policies:

Table 1-5 Dynamic Group Policies

Policy	Description	Policy Location
`Allow group` *`MyGroup`* `to manage dynamic-groups in tenancy`	To create dynamic groups	Root Compartment
`Allow group` *`MyGroup`* `to manage policies in tenancy`	To create policies in the root compartment	Root Compartment
`Allow group` *`MyGroup`* `to manage tag-namespaces in tenancy`	To create defined tags in tenancy for a stack (used for dynamic group-based policy setup)	Root Compartment

Create a Dynamic Group

If you choose to create the policies prior to stack creation your OCI administrator must create the OCI dynamic group and the OCI policies.

To create a dynamic group, complete the following steps:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Identity & Security**. Under the **Identity** group, click **Compartments**.
3. Locate the compartment you plan to use for the compute instances. Then in the **OCID** column, copy the compartment's OCID.

An OCID value looks similar to this: `ocid1.compartment.oc1..`*`alongstringoflettersandnumbers123`*.

4. Click **Dynamic Groups**.
5. Click **Create Dynamic Group**.
6. Enter a **Name** and **Description**.
7. For **Rule 1**, create a rule that includes all instances in the named compartment as members of this group.

   ```
   instance.compartment.id = 'WLS_Compartment_OCID'
   ```

   Provide the OCID for the compartment you copied in step 3.

   > **Note:**
   >
   > If domains can be created in more than one compartment, click **Rule Builder** to select **Any of the following** and then use separate lines to add each compartment OCID.

8. Click **Create Dynamic Group**.

If you choose to create the policies yourself, prior to stack creation, your OCI administrator must create the OCI dynamic group and the OCI policies.

- Required policies
- Stack selection policies

Required policies

The following table lists the *required* policies (if the `OCI Policies` option is *not* selected):

Table 1-6 Required policies (if the `OCI Policies` option is *not* selected)

Policy	Description	Policy Location
`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to manage instance-family in compartment` *`MyCompartment`*	To add volumes to Compute instances	Stack Compartment
`Allow dynamic-group MyInstancesPrincipalGroup to manage volume-attachments in compartment MyCompartment`	To attach volumes to Compute instances	Stack Compartment
`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to inspect volumes in compartment` *`MyCompartment`*	To look up volumes, so that, they can be added and attached	Stack Compartment
`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to manage volumes in compartment` *`MyCompartment`*	For future Cloning options to delete unneeded volumes	Stack Compartment

Table 1-6 (Cont.) Required policies (if the `OCI Policies` option is *not* selected)

Policy	Description	Policy Location
Allow dynamic-group *MyInstancesPrincipalGroup* to read secret-bundles in tenancy where target.secret.id = '<OCID_of_wls_password_secret>'	To set up the WLS domain administrator user	Root Compartment

Stack selection policies

The following tables lists the policies that are based on stack selection, if the `OCI Policies` option is not selected:

Table 1-7 Policies based on stack selection if the `OCI Policies` option is not selected

Selection	Policy	Description	Policy Location
Enable OS Management Hub	Allow dynamic-group *MyInstancesPrincipalGroup* to manage osmh-family in compartment *MyCompartment*	To register the Compute instance as an OS Management Hub managed instance.	Stack Compartment
	Allow dynamic-group *MyInstancesPrincipalGroup* to manage osmh-family in compartment *MyProfileCompartment*	To attach the profile with the OS Management Hub managed instance.	Profile Compartment
	Allow dynamic-group *MyInstancesPrincipalGroup* to {OSMH_MANAGED_INSTANCE_ACCESS} in compartment *MyCompartment* where request.principal.id = target.managed-instance.id	To allow the agent on the managed instance to interact with OS Management Hub	Stack Compartment
	Allow dynamic-group *MyInstancesPrincipalGroup* to {MGMT_AGENT_DEPLOY_PLUGIN_CREATE, MGMT_AGENT_INSPECT, MGMT_AGENT_READ} in compartment *MyCompartment*	(Optional) To enable monitoring of the managed instance. This is optional since the Management Agent will function correctly without this policy.	Stack Compartment
Network • **Create a new VCN** • **Create a New Subnet**	Allow dynamic-group *MyInstancesPrincipalGroup* to manage virtual-network-family in compartment *MyNetworkCompartment*	To read VCN information like CIDR, and so on, for VCN validation	Network Compartment

Table 1-7 (Cont.) Policies based on stack selection if the OCI Policies option is not selected

Selection	Policy	Description	Policy Location
Load Balancer • **Add a Load Balancer** • **Enable Authentication Using Identity Cloud Service**	Allow dynamic-group *MyInstancesPrincipalGroup* to manage load-balancers in compartment *MyNetworkCompartment*	To add load balancer backends	Network Compartment
Enable Secured Production Mode	Allow dynamic-group *<dynamic-group-name>* to manage leaf-certificates in compartment *MyCertificateCompartment*	To create certificates from the private key generated for the WebLogic Server custom identity store (used in SSL).	Certificate Compartment
	Allow dynamic-group <dynamic-group-name> to use certificate-authority-delegates in compartment *MyCertificateCompartment*	To use a certificate authority to sign a certificate using the certificate sign request (OCI CreateCertificate requires CERTIFICATE_CREATE and CERTIFICATE_AUTHORITY_APPLY permissions).	Certificate Compartment
	Allow dynamic-group <dynamic-group-name> to read leaf-certificate-bundles in compartment MyCertificateCompartment where target.leaf-certificate.bundle-type = 'CERTIFICATE_CONTENT_PUBLIC_ONLY'	To retrieve the public certificate chain for the created certificate so it can be imported into the WebLogic Server custom trust store (used for SSL connections).	Certificate Compartment
	Allow dynamic-group <dynamic-group-name> to read certificate-authorities in compartment MyCACompartment	To read the certificate authority rule for Maximum Validity Duration for Certificates so certificate requests do not exceed this duration.	Certificate Authority (CA) Compartment
Enable Authentication Using Identity Cloud Service	Allow dynamic-group *MyInstancesPrincipalGroup* to read secret-bundles in tenancy where target.secret.id = '<OCID_idcs_password_secret>'	To set the correct Client secret for the IDCS confidential application that will be created	Secret Compartment
Provision with JRF and **Autonomous Transaction Database Processing**	Allow dynamic-group *MyInstancesPrincipalGroup* to use autonomous-transaction-processing-family in compartment *MyDBCompartment*	To get the wallet from an Autonomous Database so RCU data sources can be set up	DB Compartment

Table 1-7 (Cont.) Policies based on stack selection if the `OCI Policies` option is not selected

Selection	Policy	Description	Policy Location
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to read secret-bundles in tenancy where target.secret.id = '<OCID_of_db_password_secret>'`	To get the ADMIN user required to run RCU	Root Compartment
Provision with JRF and **Database System**	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to inspect database-family in compartment` *`MyDBCompartment`*	To display the list of databases available	DB Compartment
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to read secret-bundles in tenancy where target.secret.id = '<OCID_of_db_password_secret>'`	To get the user with sysdba role required to run RCU	Root Compartment
Configure Observability and **Enable Exporting Logs to OCI Logging Service**	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to use logging-family in compartment` *`MyCompartment`*	To associate WLS logs with OCI Logging Service	Stack Compartment
Enable Autoscaling	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to use tag-namespaces in tenancy`	To use the tag the OCI functions	Root Compartment
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to read secret-bundles in tenancy where target.secret.id = '<ocid_of_auth_token_secret>'`	To read secret to access OCI repository	Root Compartment
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to use repos in tenancy`	To create repository for OCI functions	Root Compartment
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to manage functions-family in compartment` *`MyCompartment`*	To create OCI functions	Stack Compartment

Table 1-7 (Cont.) Policies based on stack selection if the `OCI Policies` option is not selected

Selection	Policy	Description	Policy Location
	`Allow dynamic-group` *`MyInstancesPrincipalGroup`* `to manage ons-subscriptions in compartment` *`MyCompartment`*	To create topic subscriptions	Stack Compartment

For information about removing secrets and policies, see About Deleting Secrets and Policies.

Dynamic Group Policies for Autoscaling

The dynamic group must include the compartment in which functions are created and the type of resource is `fnfunc`.

Create the dynamic group and required policies using a script

If you choose to create the policies prior to stack creation, then your OCI administrator must create the OCI dynamic group that includes the compartment that will contain the OCI Function application and functions.

To create a dynamic group for autoscaling, complete the following steps:

1. Create the script.
   1. Create a file with name: `create_autoscaling_policies.py`
   2. Go to Script File to Create Policies for Autoscaling Functions Post Provisioning.
   3. Copy and paste the script to the `create_autoscaling_policies.py` file.
   4. Save the file.
2. Run the following command from cloud shell:

   > **Note:**
   >
   > Ensure you have tenancy administrator privilege.

   ```
   python3 create_autoscaling_policies.py <OCID_of_the_stack>
   ```

   To run the script in a non-home region, run the following command:

   ```
   python3 create_autoscaling_policies.py <OCID_of_the_stack> --region <non-home_region>
   ```

Note:

The script uses the stack's OCID and creates function's dynamic group and policy with permissions for the function's dynamic group. It verifies if the stack has `create_policies` set to `false`, in which case it returns an error that policies would be automatically created via stack. However, you can use the `--force` option to override and create dynamic group and policies when `create_policies == true`.

Create the dynamic group and required policies through the OCI console

If you do not want to use the script, then to create the dynamic group for the function, add the following dynamic group entry:

```
ALL {resource.type = 'fnfunc',
resource.compartment.id='ocid_for_stack_compartment'}
```

If you choose to create the policies yourself prior to stack creation and select autoscaling for your stack, your OCI administrator must create the OCI autoscaling policies.

- Required policies
- Stack selection policies

Required policies

The following table lists the *required* policies (if the `OCI Policies` option is *not* selected):

Table 1-8 Required policies (if the `OCI Policies` option is *not* selected)

Policy	Description	Policy Location
`Allow dynamic-group` *`FunctionDynamicGroup`* `to inspect tenancies in tenancy`	For scaling functions to read `objectstorage` namespace for tenancy	Root Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to inspect limits in tenancy`	For scaling functions to determine if resources can be created in Network and Stack Compartment	Root Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage repos in tenancy`	For scaling functions to perform reapply during scaling	Root Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to use tag-namespaces in tenancy`	For scaling functions to use tags for the resource that might be created during reapply	Root Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage instance-family in compartment` *`MyCompartment`*	For scaling functions to create new instances during scale out	Root Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to inspect dynamic-groups in tenancy`	For scaling OCI functions to inspect dynamic groups during stack reapply	Root Compartment

Table 1-8 (Cont.) Required policies (if the `OCI Policies` option is *not* selected)

Policy	Description	Policy Location
`Allow dynamic-group` *`FunctionDynamicGroup`* `to use app-catalog-listing in compartment` *`MyCompartment`*	For scaling OCI functions to scale a stack created using Marketplace Listing	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage orm-family in compartment` *`MyCompartment`*	For scaling OCI functions to scale a stack by executing ORM stack apply	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage orm-family in compartment` *`MyNetworkCompartment`*	For scaling OCI functions to create resource manager private endpoint	Network Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage volume-family in compartment` *`MyCompartment`*	For scaling OCI functions to create volumes for new scaled out instances	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage virtual-network-family in compartment` *`MyCompartment`*	For scaling OCI functions for VCN CIDR validations	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage load-balancers in compartment` *`MyNetworkCompartment`*	For scaling OCI functions to add or remove load balancers and update backend set	Network Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to read metrics in compartment` *`MyAPMDomainCompartment`*	For scaling OCI functions to have permission to perform re-apply on the stack	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to use instance-agent-command-execution-family in compartment` *`MyCompartment`*	For scaling OCI functions to perform re-apply on the stack for scaling event	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage functions-family in compartment` *`MyCompartment`*	For scaling OCI functions to have permission to perform re-apply on the stack	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to read secret-bundles in tenancy where target.secret.id ='<ocir auth token>'`	For scaling OCI functions to have permission to read secret for repository	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage instance-agent-command-family in compartment` *`MyCompartment`*	To scaling OCI functions to perform re-apply on the stack for scaling event	Stack Compartment

Table 1-8 (Cont.) Required policies (if the `OCI Policies` option is *not* selected)

Policy	Description	Policy Location
`Allow dynamic-group` *`FunctionDynamicGroup`* `to use apm-domains in compartment` *`MyAPMDomainCompartment`*	To scaling OCI functions to perform re-apply on the stack for scaling event	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage alarms in compartment` *`MyCompartment`*	To scaling OCI functions to have permission to perform re-apply on the stack	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage cloudevents-rules in compartment` *`MyCompartment`*	For scaling OCI functions to have permission to perform re-apply on the stack	Stack Compartment
`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage ons-topics in compartment` *`MyCompartment`*	For scaling OCI functions to have permission to perform re-apply on the stack	Stack Compartment

Stack selection policies

The following table lists the autoscaling policies that are based on stack selection (if the `OCI Policies` option is *not* selected):

Table 1-9 Policies based on stack selection (if the `OCI Policies` option is *not* selected)

Selection	Policy	Description	Policy Location
Provision with JRF and **Autonomous Transaction Database Processing**	`Allow dynamic-group` *`FunctionDynamicGroup`* `to inspect autonomous-transaction-processing-family in compartment` *`MyDBCompartment`*	To get the wallet from an Autonomous Database, so that RCU data sources can be set up	DB Compartment
Enable Secured Production Mode	`Allow dynamic-group` *`FunctionDynamicGroup`* `to read certificate-authorities in compartment` *`MyCACompartment`*	To read the certificate authority rule for Maximum Validity Duration for Certificates so certificate requests do not exceed this duration.	Certificate Authority (CA) Compartment
Provision with JRF and **Database System**	`Allow dynamic-group` *`FunctionDynamicGroup`* `to inspect database-family in compartment` *`MyDBCompartment`*	To run some validations based on DB data sources	DB Compartment
Configure Observability and **Enable Exporting Logs to OCI Logging Service**	`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage log-groups in compartment` *`MyCompartment`*	To create log group	Stack Compartment
	`Allow dynamic-group` *`FunctionDynamicGroup`* `to use log-content in compartment` *`MyCompartment`*	To view the logs	Stack Compartment

Table 1-9 (Cont.) Policies based on stack selection (if the `OCI Policies` option is *not* selected)

Selection	Policy	Description	Policy Location
	`Allow dynamic-group` *`FunctionDynamicGroup`* `to manage unified-configuration in compartment`	To provision agent configurations	Stack Compartment

Create an Encryption Key

An encryption key allows you to encrypt the contents of secrets required for Oracle WebLogic Server for OCI.

Oracle WebLogic Server for OCI can use one or more keys in Oracle Cloud Infrastructure Vault to decrypt the secrets for a single domain.

Use Oracle Cloud Infrastructure Vault to create a vault and encryption key, or use an existing vault and key. Oracle WebLogic Server for OCI supports keys in standard vaults and virtual private vaults. See Managing Keys in the Oracle Cloud Infrastructure documentation.

Create Secrets for Passwords

Use secrets in Oracle Cloud Infrastructure Vault to store the passwords that you need to create a domain with Oracle WebLogic Server for OCI.

You must provide secrets for these passwords:

- Administrator password for the new domain
- Administrator password for an existing database, if you are creating a domain that includes the Java Required Files (JRF) components
- Client secret for an existing confidential application, if you are creating a domain that uses Oracle Identity Cloud Service for authentication

You must use Oracle Cloud Infrastructure Vault to create your secrets. When you create a domain using Oracle WebLogic Server for OCI, you'll be asked to provide the OCID values of the secrets.

To create a secret and copy the OCID:

1. Access the Oracle Cloud Infrastructure console.
2. Navigate to the vault that contains your encryption key.
3. Click **Secrets**, and then click **Create Secret**.
4. Enter a name to identify the secret.
5. Select your encryption key.

   The key is used to encrypt the secret contents while they are imported to the vault.
6. In **Secret Contents**, specify the password you want to store in this secret.

   Ensure the password meets the criteria for which it will be used (for example, the WebLogic Server administrator password requires that it starts with an alphabet, is between 8 and 30 characters long, contains at least one numeric, and optionally, any number of the special characters: '`$`', '`#`', and '`_`'. For example, `Ach1z0#d`.).

Passwords entered in plain-text are base64-encoded before they are sent to Oracle WebLogic Server for OCI.

7. Click **Create Secret**.
8. When the secret is created, click the name.
9. Copy the **OCID** for the secret.
10. Repeat steps 2 through 9 to create the remaining secrets you need.

See Managing Secrets in the Oracle Cloud Infrastructure documentation.

For information about removing secrets and policies, see About Deleting Secrets and Policies.

Create an SSH Key

Create a secure shell (SSH) key pair so that you can access the compute instances in your Oracle WebLogic Server domains.

A key pair consists of a public key and a corresponding private key. When you create a domain using Oracle WebLogic Server for OCI, you specify the public key. You then access the compute instances from an SSH client using the private key.

On a UNIX or UNIX-like platform, use the `ssh-keygen` utility. For example:

```
ssh-keygen -b 2048 -t rsa -f mykey
cat mykey.pub
```

On a Windows platform, you can use the PuTTY Key Generator utility. See Creating a Key Pair in the Oracle Cloud Infrastructure documentation.

Create a Virtual Cloud Network

Oracle WebLogic Server for OCI can create a Virtual Cloud Network (VCN) in Oracle Cloud Infrastructure for a new Oracle WebLogic Server domain, or you can create your own VCN before creating a domain.

A VCN includes one or more subnets, route tables, gateways, DHCP options, and network security groups or security lists.

By default subnets are public. Any compute instances assigned to a private subnet cannot be directly accessed from outside of Oracle Cloud.

If you create a VCN before creating a domain, then the VCN must meet the following requirements:

- The VCN must use DNS for hostnames.
- If you plan to use a public subnet for the bastion or load balancer, then the VCN must include an Internet Gateway.
- If you plan to create a public subnet in the VCN before creating a domain, then the VCN must include a route table that directs traffic to the Internet Gateway.
- When you create DHCP options, it is recommended to select the DNS type as **Internet and VCN Resolver**. If your network configuration demands you to select **Custom Resolver**, then add Oracle DNS Server IP `169.254.169.254` as the last DNS server entry.

If you plan to use a private subnet for the Oracle WebLogic Server compute instances, then the VCN must meet these additional requirements:

- The VCN for the bastion must have a route table that directs traffic to the Internet Gateway.
- The VCN must include a service gateway or a Network Address Translation (NAT) gateway, to provide access to other cloud services. For a service gateway, select the option **All <*Region*> Services In Oracle Services Network**.
- If you want to create the private subnet before creating a domain, then the VCN must also include a route table that directs traffic to the service gateway or the NAT gateway. For a service gateway route rule, select the option **All <*Region*> Services In Oracle Services Network**. If you intend to have your WebLogic Domain authenticate users from an enterprise application, formerly known as using Oracle Identity Cloud Service (IDCS) in WLS for OCI, you must use a NAT gateway.
- If you use a private subnet for a WebLogic domain, which requires access to the internet, then the VCN must also include a route table that directs traffic to access the NAT gateway.

If you use an existing VCN for a domain, and also choose for Oracle WebLogic Server for OCI to create new subnets for the domain, then Oracle WebLogic Server for OCI will also create the required route tables in the VCN.

If you use an existing VCN and existing subnets in Oracle WebLogic Server for OCI, you can certify the existing network setup using helper scripts. See Validate Existing Network.

See these topics in the Oracle Cloud Infrastructure documentation:

- VCNs and Subnets
- Access to Oracle Services: Service Gateway

Create a Private Endpoint

Oracle WebLogic Server for OCI can create private endpoints in Oracle Cloud Infrastructure that allows you to check the status of the resources in your domain. You can also create your own private endpoints before creating a domain.

See Managing Private Endpoints in Oracle Cloud Infrastructure documentation.

Create a Network Security Group

Oracle WebLogic Server for OCI can create network security groups (NSGs) in Oracle Cloud Infrastructure for a new Oracle WebLogic Server domain, or you can create your own NSGs before creating a domain.

In Oracle WebLogic Server for OCI, you can use NSGs for your compute instances, load balancer, bastion instance, and file storage.

For an NSG security rule's source (for ingress rules) or destination (for egress rules), you can specify an NSG ID instead of a CIDR. This means you can easily write security rules to control traffic between two NSGs in the same VCN, or traffic within a single NSG.

You can create one NSG for each type of resource; each resource will have it's own set of security rules.

If you want to use an existing NSG for your resources, the NSGs must include specific security rules. The following table lists the security rules for the NSGs:

Table 1-10 Network Security Group Rules

NSG	Rule Type	Protocol	Source	Target	Destination Ports	Description
Bastion	Stateful Ingress	TCP	0.0.0.0/0		22	SSH access
Bastion	Stateful Egress	All		0.0.0.0/0	All	Egress to everything
Load Balancer	Stateful Ingress	TCP	0.0.0.0/0		443	Application traffic using HTTPS
Load Balancer	Stateful Egress	All		0.0.0.0/0	All	Egress to everything
WebLogic Administration Server (secured production mode)	Stateful Ingress	TCP	Bastion NSG OCID		9002	Administration Server SSL port
WebLogic Managed Server (secured production mode)	Stateful Ingress	TCP	Admin network, or load balancer NSG OCID		7004	Managed Server SSL port
WebLogic Managed Server (secured production mode)	Stateful Ingress	TCP	WebLogic Managed Server NSG OCID		9072	Administration Server internal SSL port
WebLogic Managed Server (secured production mode)	Stateful Ingress	TCP	WebLogic Managed Server NSG OCID		9002	Administration Server SSL ports
WebLogic Managed Server (secured production mode)	Stateful Ingress	TCP	WebLogic Managed Server NSG OCID		9004	Managed Server Administration SSL port
WebLogic Managed Server (secured production mode)	Stateful Ingress	TCP	WebLogic Managed Server NSG OCID		5556	Node Manager port
WebLogic Administration Server (not secured production mode)	Stateful Ingress	TCP	Bastion NSG OCID		7001, 7002	Administration Server ports

Table 1-10 (Cont.) Network Security Group Rules

NSG	Rule Type	Protocol	Source	Target	Destination Ports	Description
WebLogic Administration Server (not secured production mode)	Stateful Ingress	TCP	Admin network, or load balancer NSG OCID		7003, 7004	Managed Server ports
WebLogic Managed Server	Stateful Ingress	TCP	Bastion NSG OCID		22	SSH access
WebLogic Managed Server	Stateful Ingress	TCP	Load balancer NSG OCID		9999 or custom redirect port	App Gateway in Oracle Identity Cloud Service
WebLogic Managed Server	Stateful Ingress	All	WebLogic Managed Server NSG OCID		All	All ports among server
WebLogic Managed Server	Stateful Egress	TCP		Database network CIDR	1521 or custom DB port	Database for JRF-enabled domain
WebLogic Managed Server	Stateful Egress	All		0.0.0.0/0	All	Egress to everything
Mount Target	Stateful Ingress	TCP	VCN CIDR		2048-2050	File system integration
Mount Target	Stateful Ingress	TCP	VCN CIDR		111	File system integration
Mount Target	Stateful Ingress	UDP	VCN CIDR		111	File system integration
Mount Target	Stateful Ingress	UDP	VCN CIDR		2048	File system integration
Mount Target	Stateful Egress	TCP	VCN CIDR		2048-2050	File system integration
Mount Target	Stateful Egress	TCP	VCN CIDR		111	File system integration
Mount Target	Stateful Egress	UDP	VCN CIDR		111	File system integration

If you use existing subnets and existing network security groups during stack creation, you can create the NSGs using a script.

To use the script, create a file named, `create_nsg`, and copy and paste the script in create_nsg.sh to this `create_nsg` file.

Then run the following commands:

```
#Set execute permission to the create_nsg.sh file.
chmod +x create_nsg.sh
./create_nsg.sh -w <WLS_Subnet_OCID> -l <Load_Balancer_Subnet_OCID> -b
```

```
<Bastion_Subnet_OCID> -f <File_System_Storage_Subnet_OCID> -d
<Database_Subnet_OCID>
```

This script creates NSGs and adds the security rules to the NSGs, for compute instances, load balancer, bastion instance, file system, and database subnet.

See Network Security Groups in the Oracle Cloud Infrastructure documentation.

Create a Private Subnet for the Oracle WebLogic Server Nodes

Oracle WebLogic Server for OCI can create a subnet in Oracle Cloud Infrastructure for a new Oracle WebLogic Server domain, or you can create your own subnet before creating a domain.

A subnet is a component of a Virtual Cloud Network (VCN). When you create a domain with Oracle WebLogic Server for OCI, the Oracle WebLogic Server compute instances are assigned to a subnet.

By default subnets span an entire region in Oracle Cloud Infrastructure.

Oracle WebLogic Server for OCI supports both regional and AD-scoped subnets.

If you assign a private subnet to the domain, the nodes cannot be directly accessed from outside of Oracle Cloud. Oracle WebLogic Server for OCI can create a bastion node on a public subnet, which you can use to administer the nodes that comprise your domain.

If you assign a private subnet that needs access to the public internet, you must include a route table that directs traffic to the NAT gateway.

If you want to use an existing subnet for the Oracle WebLogic Server nodes when creating a domain, the subnet must meet the following requirements:

- The subnet must use DNS for hostnames.
- The subnet must have a security list that enables inbound access to the SSH port (22) and to the administration server ports (by default, 7001 and 7002 or, if secured production mode, 7002 and 9002 by default).
  Alternatively, if you want to use network security groups (NSGs), the VCN of the subnet must have a network security group that enables inbound access to the SSH port (22) and to the administration server ports (by default, 7001 and 7002 or, if secured production mode, 7002 and 9002 by default).
- The subnet must have a security list that enables inbound access to the managed server ports (by default, 7003 and 7004 or if secured production mode, only 7004 by default).
  Alternatively, if you want to use network security groups, the VCN of the subnet must have a network security group that enables inbound access to the managed server ports (by default, 7003 and 7004 or if secured production mode, only 7004 by default) on the subnet you plan to use for WebLogic Server.

  If you are using a load balancer, the security list's source or the NSG's source should be restricted to the subnets that you plan to use for the load balancer.
- If you are creating a domain that uses Oracle Identity Cloud Service to authenticate application users, the subnet must have a security list that enables inbound access to the listen port for the App Gateway in Oracle Identity Cloud Service (by default, 9999).
- If you are creating a domain with the Java Required Files (JRF) components, the subnet must have a security list that enables outbound access to the database port (1521 by default) on the database subnet.
- If you are creating a domain with the JRF components, and the database is on a different VCN, then the subnet must have a security list that enables outbound access to port 53

(both TCP and UDP) on the subnet that you plan to create for DNS. The subnet must also be associated with the default DHCP option for the VCN (`Default DHCP Options for <vcn_name>`). The subnet cannot use a custom DNS resolver.

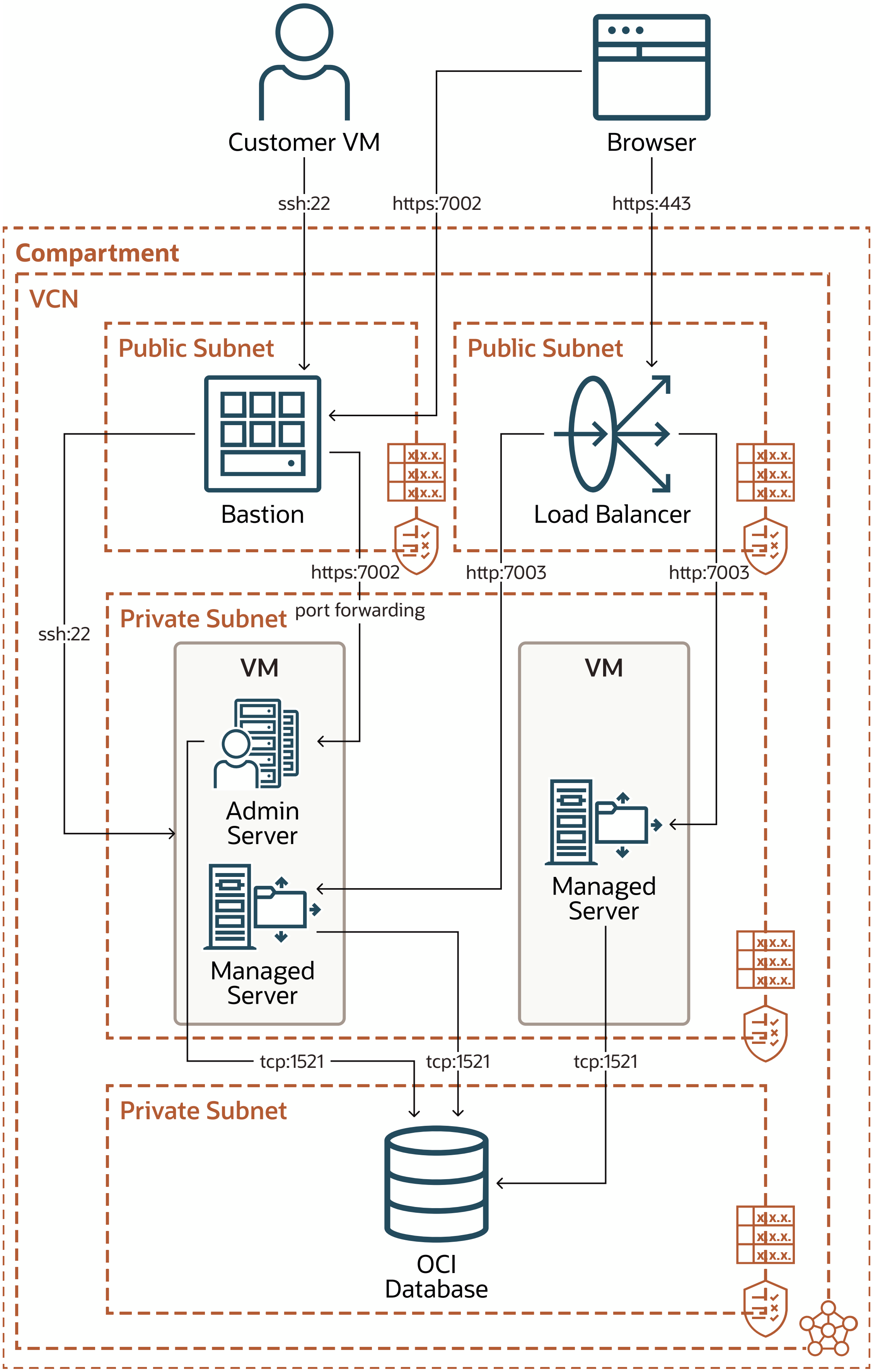
- The subnet must have a security list that enables inbound access to the node manager port (by default, 5556).
  Alternatively, if you want to use network security groups, the VCN of the subnet must have a network security group that enables inbound access to the node manager port (by default, 5556) on the subnet you plan to use for WebLogic Server.

If you use an existing subnet, you can specify the subnet compartment that is different than the VCN compartment.

If you use an existing subnet, you can certify the existing network setup using helper scripts. See Validate Existing Network.

The following diagram illustrations the default destination ports.

Figure 1-6   Default Destination Ports



The following table summarizes the security list requirements for an existing subnet. However, if you use network security groups, refer the *Network Security Rules* table in Create a Network Security Group.

> **Note:**
>
> If you change the default ports when creating a domain, then open the related ports accordingly.

Table 1-11 Security List Rules

Rule Type	Source CIDR and Protocol	Default Destination Ports	Description
Stateful Ingress	Bastion network, TCP	22	SSH access
Stateful Ingress	Bastion network, TCP	7001, 7002	Administration Server ports (not secured production mode)
Stateful Ingress	Bastion network, TCP	9002	Administration Server SSL port (secured production mode)
Stateful Ingress	Your admin network, or your load balancer subnet, TCP	7004	Managed Server SSL port (secured production mode)
Stateful Ingress	Weblogic subnet, CIDR	22	Administration Server ports (not secured production mode)
Stateful Ingress	Weblogic subnet, CIDR	9071	Used for provisioning and scaling (not secured production mode)
Stateful Ingress	Weblogic subnet, CIDR	9072	SSL internal port used for provisioning and scaling (secured production mode)
Stateful Ingress	Weblogic subnet, CIDR	9002	Administration Server SSL administration port (secured production mode)
Stateful Ingress	Weblogic subnet, CIDR	5556	Used for accessing node manager
Stateful Ingress	Weblogic subnet, CIDR	9004	Managed Server SSL administration port (secured production mode)
Stateful Ingress	Your admin network, or your load balancer subnet, TCP	7003, 7004	Managed Server ports (not secured production mode)
Stateful Ingress	Your load balancer subnet, TCP	9999 or custom redirect port	App Gateway in Oracle Identity Cloud Service
Stateful Ingress	Your database network, TCP	1521 or custom DB port	Database for JRF-enabled domain
Stateful Ingress	DNS subnet, TCP	53	JRF-enabled domain and database is on a different VCN

Table 1-11 (Cont.) Security List Rules

Rule Type	Source CIDR and Protocol	Default Destination Ports	Description
Stateful Ingress	DNS subnet, UDP	53	JRF-enabled domain and database is on a different VCN
Stateful Ingress	Mount target subnet, TCP	2048-2050	File system integration
Stateful Ingress	Mount target subnet, TCP	111	File system integration
Stateful Ingress	Mount target subnet, UDP	111	File system integration
Stateful Ingress	Mount target subnet, UDP	2048	File system integration
Stateful Egress	Mount target subnet, TCP	2048-2050	File system integration
Stateful Egress	Mount target subnet, TCP	111	File system integration
Stateful Egress	Mount target subnet, UDP	111	File system integration

See VCNs and Subnets in the Oracle Cloud Infrastructure documentation.

Configure the Load Balancer

Oracle WebLogic Server for OCI can create a load balancer in Oracle Cloud Infrastructure that is used to distribute traffic across the servers in your domain, or you can use your own load balancer for the domain.

Note:

- If you use your own load balancer, create it in the Network compartment, not the Stack compartment.
- The following instructions assume that you want to use TLS (SSL) on the load balancer and terminate TLS (SSL) at the load balancer.
  - If you are not using TLS (SSL), then you should not add the Rule Set and the Certificate Resource that are mentioned as part of the instructions.
  - If you are not terminating TLS (SSL) at the load balancer, you should ensure the following:
    - Do not add the Rule Set.
    - Change the port to the HTTPS port used by the managed servers to 7004 or the HTTPS port that matches with the managed server HTTPS port of the backend set.
    - Add the TLS (SSL) certificates required to access the managed server over HTTPS. See Create a Certificate Authority.

If you use an existing load balancer, then you must configure the load balancer as follows:

Note:

If you will be enabling secured production mode ensure that you have an OCI Certificate Authority available. If you do not yet have an OCI Certificate Authority see Create a Certificate Authority.

1. Create a backend set
2. Create a rule set
3. Create a listener
4. Create a routing policy

Create a backend set

Specify the load balancing and the health check policy for the backend set.

Note:

When you create a backend set, do not add backends for the backend set.

1. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Load Balancers**.
2. Click the name of the load balancer.
3. Under **Resources**, click **Backend Sets**.

4. Click **Create Backend Set**.
5. Enter a name for the backend set.
6. From **Traffic Distribution Policy**, select *Weighted Round Robin*.
7. From **Session Persistence**, select **Enable load balancer cookie persistence**.
8. Specify the following properties for **Health Check**:
    - Protocol: *HTTP*
    - Port: *7003* or *7004* if you will be enabling secured production mode
      The default port is *7003*. You must specify the port that matches with managed server port of the backend set.
    - Interval in MS: *30000*
    - Timeout in MS: *3000*
    - Number of retries: *3*
    - Status code: *404*
    - URL Path (URI): */*
    - Response body regex: *.**
9. Click **Create Backend Set**.

Create a rule set

Specify the rule sets for the load balancer.

> **Note:**
>
> This section is not required when secured production mode is selected since SSL is configured end to end. Without secured production mode SSL is terminated at the load balancer.

1. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Load Balancers**.
2. Select the **Compartment** in which the network resources for your domain were created.
3. Click the name of the load balancer.
4. Click **Rule sets**.
5. Click **Create rule set**.
6. For **Name**, enter `SSLHeaders`.
7. If you will be enabling secured production mode select "Use SSL" and specify the following properties:
    - Certificate resource: Certificate service managed certificate
    - Select **Certificate authority** option.
    - Certificate authority in CA Compartment: Select the certificate authority from section Create a Certificate Authority
    - Select **Verify peer certificate**

- Verify depth: 1

8. Click **Specify request header rules**.
9. Specify the following header parameters:
   - **Action**: Add Request Header
   - **Header**: `is_ssl`
   - **Value**: `ssl`
10. Click **Another request header rule**.
11. Specify the following header parameters:
    - **Action**: Add Request Header
    - **Header**: `WL-Proxy-SSL`
    - **Value**: `true`
12. Click **Create**, and then click **Close**.

Create a listener

Specify the listener properties for the load balancer.

1. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Load Balancers**.
2. Click the name of the load balancer.
3. Under **Resources**, click **Listeners**.
4. Click **Create Listener**.
5. Enter a Name for the listener.
6. Select the *HTTPS* Protocol.
7. For **Port**, enter *443*.
8. For **Certificate Resource**, select *Load Balancer Managed Certificate*, if not already selected, and then select the **Certificate Name** that you imported for the certificate resource.
9. Select the backend set that you created for the load balancer. See Create a backend set.
10. Click **Create Listener**.
11. After the listener is created, edit it to add the Rule Set you created earlier. See Create a rule set.
12. In the Edit listener page, in the Rule sets section, select the rule set **SSLHeaders** from the drop-down list.
13. Click **Save changes**, and then click **Close**.

Add the routing policy

Specify the routing policy for the load balancer.

1. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Load Balancers**.
2. Click the name of the load balancer.

3. Under **Resources**, click **Routing policies**.
4. Click **Routing Policy**.
5. Enter a name for the routing policy.
6. Specify the following conditions for the rule:
   - When the following conditions are met…: *If All Match*
   - Condition Type: *Path*
   - Operator: *Is*
   - URL String: */myPath*
7. Select the backend set that you created for the load balancer. See Create a backend set.
8. Click **Next**.
9. In the **Change Order** column, corresponding to the rule, click the down arrow to see a summary of the conditions and actions set in a rule.
10. Click **Reorder** to move a rule up or down in the policy order.
11. When the routing policy rules are created and in the right order, click **Create Routing Policy**.

See the following topics in Oracle Cloud Infrastructure documentation:

- Creating a Load Balancer
- Request Routing for Load Balancers

Create a Subnet for the Load Balancer

Oracle WebLogic Server for OCI can create subnets in Oracle Cloud Infrastructure for the load balancer that is used to access an Oracle WebLogic Server domain, or you can create your own subnets before creating a domain.

A subnet is a component of a Virtual Cloud Network (VCN). When you create a domain with a load balancer using Oracle WebLogic Server for OCI, the load balancer is assigned a subnet.

By default subnets span an entire region in Oracle Cloud Infrastructure.

To ensure high availability for a load balancer, you must assign it either one regional subnet, or two AD-scoped subnets.

If you want to use an existing subnet for the load balancer, the subnet must meet the following requirements:

- The subnet must use DNS for hostnames.
- The subnet must be public.
- The subnet must have a security list that enables inbound access to ports 80 and 443 or, if secured production mode, only 443.
  Alternatively, if you want to use network security groups (NSGs), the VCN of the subnet must have a network security group that enables inbound access to ports 80 and 443.
- The subnet must have a security list that enables outbound access to the managed server ports(by default, 7003 and 7004 or if secured production mode, only 7004 by default) on the subnet that you plan to use for Oracle WebLogic Server.
  Alternatively, if you want to use network security groups, the VCN of the subnet must have a network security group that enables outbound access to the managed server ports (by default, 7003 and 7004) on the subnet that you plan to use for Oracle WebLogic Server.

If you use an existing subnet, you can specify the subnet compartment that is different than the VCN compartment.

If you use an existing subnet, you can certify the existing network setup using helper scripts. See Validate Existing Network.

See VCNs and Subnets in the Oracle Cloud Infrastructure documentation.

Create a Subnet for the Bastion Node

Oracle WebLogic Server for OCI can create a public subnet in Oracle Cloud Infrastructure for the bastion node that is used to access a private Oracle WebLogic Server domain, or you can create your own subnet before creating a domain.

A subnet is a component of a Virtual Cloud Network (VCN). When you create a domain in Oracle WebLogic Server for OCI, you can assign the Oracle WebLogic Server compute instances to a public subnet or a private subnet. If you assign a private subnet, then the compute instances can not be directly accessed from outside of Oracle Cloud. Oracle WebLogic Server for OCI can create a bastion compute instance on a public subnet, and from this bastion you can administer the Oracle WebLogic Server compute instances.

> **Note:**
>
> Configuring a bastion is optional.
>
> If you do not configure a bastion, no status is returned for provisioning. You must check the status of provisioning by connecting to each compute instance and confirm that the `/u01/provStartMarker` file exists with details found in the file `/u01/logs/provisioning.log` file. See Configure a Bastion.

By default subnets span an entire region in Oracle Cloud Infrastructure.

Oracle WebLogic Server for OCI supports both regional and AD-scoped subnets.

If you want to use an existing subnet for the bastion node when creating a domain, then the subnet must meet the following requirements:

- The subnet must use DNS for hostnames.
- The subnet must be public.
- The subnet's route table must have an Internet Gateway rule
- The subnet must have a security list that enables inbound access to the SSH port (22).
  Alternatively, if you want to use network security groups (NSGs), the VCN of the subnet must have a network security group that enables inbound access to the SSH port (22).
- The subnet must have a security list that enables outbound access to the SSH port (22) on the subnet that you plan to use for Oracle WebLogic Server.
  Alternatively, if you want to use network security groups, the VCN of the subnet must have a network security group that enables outbound access to the SSH port (22) on the subnet that you plan to use for Oracle WebLogic Server.

If you use an existing subnet, the subnet is created in the subnet compartment that is different than the VCN compartment.

If you use an existing subnet, you can certify the existing network setup using helper scripts. See Validate Existing Network.

See VCNs and Subnets in the Oracle Cloud Infrastructure documentation.

Create a Subnet for the Mount Target

Oracle WebLogic Server for OCI can create subnets in Oracle Cloud Infrastructure for the mount target that is used to access an Oracle WebLogic Server domain, or you can create your own subnets before creating a domain.

A subnet is a component of a Virtual Cloud Network (VCN). When you create a domain with a mount target using Oracle WebLogic Server for OCI, the mount target is assigned a subnet.

By default subnets span an entire region in Oracle Cloud Infrastructure.

If you want to use an existing subnet for the mount target, the subnet must meet the following requirements:

- The subnet must use DNS for hostnames.
- The subnet must be public.
- The subnet must have a security list that enables inbound access to ports 111 and 2048.
  Alternatively, if you want to use network security group (NSGs), the VCN of the subnet must have a network security group that enables inbound access to ports 111 and 2048.
- The subnet must have a security list that enables outbound access to the SSH port 111 on the subnet that you plan to use for Oracle WebLogic Server.
  Alternatively, if you want to use network security group (NSGs), the VCN of the subnet must have a network security group that enables outbound access to the SSH port 111 on the subnet that you plan to use for Oracle WebLogic Server.

If you use an existing subnet, you can specify the subnet compartment that is different than the VCN compartment.

If you use an existing subnet, you can certify the existing network setup using helper scripts. See Validate Existing Network.

See VCNs and Subnets in the Oracle Cloud Infrastructure documentation.

Create a Database

Before creating an Oracle WebLogic Server domain that includes the Java Required Files (JRF) components, you must create a database in Oracle Cloud Infrastructure.

> **Note:**
>
> You cannot create an Oracle WebLogic Server domain that includes the Java Required Files (JRF) components for Oracle WebLogic Server 14.1.1.0 as this version does not support JRF.

A JRF-enabled domain supports the Oracle Application Development Framework (ADF). When you create a domain with Oracle WebLogic Server for OCI and associate it with an existing database, Oracle WebLogic Server for OCI does the following:

- Provisions the schemas to support the JRF components in the selected database
- Provisions data sources in the domain that provide connectivity to the selected database
- Deploys the JRF components and libraries to the domain

Oracle WebLogic Server for OCI also provides a tool to delete the JRF schemas for a specific domain from the database.

Choose one of these database options:

- Oracle Autonomous Database
  - Create an autonomous database using either the dedicated or shared infrastructure option. You can also create a Free-Tier autonomous database.
  - See Creating an Autonomous Database in the Oracle Cloud Infrastructure documentation.

  > **Note:**
  >
  > Oracle Autonomous Database provides a private endpoint configuration which can be created in subnet within a VCN, by using listen port `1522`.

- Oracle Cloud Infrastructure Database
  - Create bare metal, virtual machine (VM), and Exadata DB systems. For a 1-node VM DB system, note that you can use the fast provisioning option to create the database. Oracle WebLogic Server for OCI supports using Logical Volume Manager as the storage management software for a 1-node VM DB system.
  - For Oracle WebLogic Server 14.1.2.0.0 or 12.2.1.4.0, you can also specify a database connection string. This database connection string can be used only with existing VCN. To know the database connection string details, see Database Connect String for Database Version and Type in Configure Database Parameters and VCN Peering.
  - See Creating Bare Metal and Virtual Machine DB Systems or Managing Exadata DB Systems in the Oracle Cloud Infrastructure documentation.

The database must allow your domain to access its listen port (1521 by default):

- Oracle Autonomous Database - Update your access control list (ACL), if necessary.
- Oracle Cloud Infrastructure Database - by default, Oracle WebLogic Server for OCI creates a security list on the database's VCN that allows the WebLogic Server subnet to access the database. Alternatively, you can manually update the security lists in the database's VCN, or update the network security group that is assigned to the database. If you use database connection string, security list is not created to access the database. You must ensure that the ports are open to access the database.

To create a JRF-enabled domain with Oracle WebLogic Server for OCI, you need the following information about the database:

- Administrator credentials
- Pluggable database (PDB) name (only for Oracle Cloud Infrastructure Database running Oracle Database 12c or later)

If your database and domain are in different VCNs, then Oracle WebLogic Server for OCI configures local peering between the two VCNs. To support VCN peering, you must perform the following prerequisite tasks:

- Create an local peering Gateway (LPG) in the database VCN.
- Add a route to the current route table of the database subnet, to direct traffic to the CIDR of the WebLogic subnet to the LPG.
- Open the database port to the WebLogic subnet CIDR.

- If you use an existing VCN and existing subnet, or existing VCN and new subnets, you must add the default private view of the database VCN to the DNS resolver of the existing VCN. See Add a DNS view to the DNS Resolver.

You must also meet the following additional requirements to support VCN peering for the databases:

- The CIDRs for the VCNs must not overlap. For example, you cannot create a domain in VCN `10.0.0.0/16` that uses a database in VCN `10.0.0.1/24`.
- The database subnet must be associated with the default DHCP option for the VCN (`Default DHCP Options for` *`<vcn_name>`*). The subnet cannot use a custom DNS resolver.

If you use Oracle Cloud Infrastructure Database with connection string, you must peer the VCNs manually before creating the stack if your database VCN is on a different VCN than the WebLogic Server VCN. See Manual VCN Peering.

The following table summarizes the security list requirements for an existing subnet that will use local VCN peering to communicate with the domain.

Rule Type	CIDR and Protocol	Destination Ports	Description
Stateful Ingress	WebLogic Server subnet, TCP	1521 or custom database port	Database access
Stateful Ingress	DNS subnet, TCP	53	Access to custom DNS resolver
Stateful Ingress	DNS subnet, UDP	53	Access to custom DNS resolver

Oracle WebLogic Server for OCI supports the same database versions and drivers as those for on-premise WebLogic Server installations. Refer to the following excel files (`xls`) at Oracle Fusion Middleware Supported System Configurations:

- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.2.0.0)
- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.1.0.0)
- System Requirements and Supported Platforms for Oracle Fusion Middleware 12c (12.2.1.4.0)

> **Note:**
>
> From release 21.4.3 (December 9, 2021) onwards, you cannot provision a domain in Oracle Oracle WebLogic Server for OCI for Oracle WebLogic server versions 11g (10.3.6.0) and 12c (12.2.1.3) from the Marketplace.

Create a Confidential Application

Before creating an Oracle WebLogic Server domain that integrates with Oracle Identity Cloud Service, you must create a confidential application, and then identify its client ID and client secret.

This configuration is supported only for Oracle Cloud accounts that include Oracle Identity Cloud Service 19.2.1 or later.

When creating a new domain, Oracle WebLogic Server for OCI provisions an App Gateway and other security components in Oracle Identity Cloud Service. In order for Oracle WebLogic Server for OCI to perform these tasks, you must provide the following information:

- Your Oracle Identity Cloud Service instance ID, which is also referred to as your tenant name. This ID is typically found in the URL you use to access the Oracle Identity Cloud Service console, and has the format `idcs-<GUID>`.
- The client ID of a confidential application in Oracle Identity Cloud Service
- The client secret of the confidential application. You must use Oracle Cloud Infrastructure Vault to create a secret to store the client secret. You will asked to provide the OCID of the secret in the vault. See Create Secrets for Passwords.

Create a confidential application for Oracle WebLogic Server for OCI, or use an existing one. You can use a single confidential application in Oracle Identity Cloud Service to create multiple domains.

1. From the Oracle Identity Cloud Service Console, click the navigation menu, and then select **Applications**.
2. Click **Add**.
3. Select **Confidential Application**.
4. Enter a **Name**, and then click **Next**.
5. Click **Configure this application as a client now**.
6. For **Allowed Grant Types**, select **Client Credentials**.
7. Below **Grant the client access to Identity Cloud Service Admin APIs**, click **Add**.
8. Select **Identity Domain Administrator**, and then click **Add**.
9. (Optional) For a WebLogic Server 14.1.2.0 or 12.2.1.4 domain only, add **Cloud Gate App Role**. You can add this role after you create your WebLogic Server domain but you may need to restart the domain.

   > **Caution:**
   >
   > Add Cloud Gate App Role only if you need to open and log in to the Fusion Middleware Control Console from the Internet. While enabling this role means the Fusion Middleware Control Console is accessible from the Internet, it also means any application would be allowed to look up users.

10. Complete the Add Confidential Application wizard. Record the values of **Client ID** and **Client Secret**.
11. Select the check box for your application, click **Activate**, and then click **OK**.
12. In the Oracle Cloud Infrastructure console, create a secret in a vault to store the client secret of your confidential application.

See Add a Confidential Application in *Administering Oracle Identity Cloud Service*.

Create an Application Performance Monitoring Domain

Create an Application Performance Monitoring domain or use an existing Application Performance Monitoring domain.

When you create a domain with Oracle WebLogic Server for OCI, you can enable Application Performance Monitoring integration and provide the Application Performance Monitoring domain details. So, during provisioning, the Application Performance Monitoring Java agent is installed on each of the WebLogic compute instance. This agent records metrics and spans and sends them to the specified Application Performance Monitoring domain.

Metric-based autoscaling depends on the Application Performance Monitoring integration feature to be enabled during provisioning.

See Create an APM Domain in the Oracle Cloud Infrastructure documentation.

Note:

Application Performance Monitoring always free domain is not supported for autoscaling.

Create an Auth Token

In order for Oracle Cloud Infrastructure Registry to deploy autoscaling OCI Functions, you must provide an auth token.

Oracle WebLogic Server for OCI can access the registry as the same user that creates a stack, or as a different user. Every user in Oracle Cloud Infrastructure can be associated with up to two auth tokens. You can create a new auth token for a user with access to Oracle Cloud Infrastructure Registry, or use an existing auth token. When creating an auth token, be sure to copy the token string immediately. You can't retrieve it again later using the console.

See Managing User Credentials in the Oracle Cloud Infrastructure documentation.

Create a Certificate Authority

Create a Certificate Authority or use an existing Certificate Authority.

When you create a domain with Oracle WebLogic Server for OCI, you can enable Secured Production Mode and provide the Certificate Authority to be used for issuing SSL certificates. During provisioning a private key is generated for use in the PKCS12 Custom Identity SSL configuration of the WebLogic Administration Server, all managed servers, all node managers, and if Authentication Using Identity Domains (aka Identity Cloud Service) is enabled, for the AppGateway. A certificate signing request (CSR) for a wildcard certificate is presented to the OCI Certificate service with the generated private key using the Certificate Authority specified. The certificate chain for the certificate created is retrieved and added to a PKCS12 keystore for use in the Custom Trust SSL configuration of the WebLogic Administration Server, all managed servers, and if Authentication Using Identity Domains (aka Identity Cloud Service) is enabled, for the AppGateway.

See Creating a Certificate Authority.

> **Note:**
>
> Only Root Certificate Authority without any Subordinate Certificate Authorityis supported.

> **Note:**
>
> When creating an OCI Certificate Authority the Maximum Validity Duration for Certificates (Days) defaults to 90 days. WLS for OCI will set the certificate expiration based on this value or 365, depending on which is lower. Ensure that you are aware that the certificates will need to be renewed by the expiration date.

Create an OS Management Hub Profile

When you create a domain with Oracle WebLogic Server for OCI, you can enable OS Management Hub and provide an existing profile.

For more information see:

- Creating a Profile for how to create a OS Management Hub Profile.
- Listing Profiles in the Oracle Cloud Infrastructure documentation to find an existing profile and get its OCID.

> **Note:**
>
> To prevent a proliferation of new profiles Oracle strongly recommends that you use an existing profile. Once you have created one stack with a new profile you can reuse that profile in other stacks. See Listing Profiles in the Oracle Cloud Infrastructure documentation to find the profile created by a stack and get its OCID.

Validate Existing Network Setup

You can use helper scripts from the Oracle Cloud Infrastructure Cloud shell to certify the existing network setup (existing VCN and existing WebLogic Server subnet) in Oracle WebLogic Server for OCI. See Using Cloud Shell in Oracle Cloud Infrastructure documentation.

The helper scripts perform the following validations and functions:

- Validates if the service gateway or the NAT gateway is created for private subnets.
- Validates if the database port is accessible from WebLogic Server subnets.
- Validates if port number of the administration server is accessible to the SSH and T3 ports from managed servers.
- Validates if internet gateway is created for public bastion and WebLogic Server subnets. This validation is optional and can be ignored for private or FastConnect network.

- Checks if port 22 in WebLogic Server subnet is open for access to the CIDR of the bastion instance subnet or bastion host IP.
- Checks if port 7003, port 7004 if -z or --securemode argument set to true, in WebLogic Server subnet is open for access to the CIDR of the load balancer subnet.
- Checks if port 443 in load balancer subnet is open for access to `0.0.0.0/0` IP address.
- Checks if ports 111 (both TCP and UDP), 2048-2050 (TCP) and 2048 (UDP) in file system storage subnets are open for access.
- Checks if the private subnet for the Oracle WebLogic Server compute instances using the service gateway route rule has **All <*Region*> Services In Oracle Services Network** as the destination.

You must install the following tools before you run the validation scripts:

- OCI CLI 2.14.1 or later

  See Install CLI.
- jq 1.5 or later

  See Download jq.
- Bash 4.0 or later

  See Upgrade Bash Version.

As these tools are installed in the Cloud Shell by default, it is recommended to run the validation scripts from the Cloud Shell. The scripts can be copied to Cloud Shell or any Linux terminal running bash 4 and above.

Validation scripts can be run prior to provisioning or post-provisioning.

Validation can be done post-provisioning if the network configuration changes causes the following issues:

- Scale out fails on reapply. This occurs when ports 22 and 9071 or if -z or --securemode argument set to true ports 22, 9072, 9002, 9004, are not open to access Weblogic Server subnet CIDR in Weblogic Server subnet.
- Database connectivity fails. This occurs when the database port is not open to access Weblogic Server subnet CIDR in database subnet, or NAT gateway or service gateway for the WebLogic Server VCN is removed resulting in access failure to the Oracle Autonomous Database.

Using the Validation Script

You can run the helper scripts to perform validations for existing private subnets, existing public subnets, and network security groups.

You must run the commands on the validation script file to check the existing network setup. For example, in this case, let's create a file named, `network_validation.sh`, and then copy and paste the script in network_validation.sh to this `network_validation.sh` file.

Run the following commands on the validation script file, `network_validation.sh`:

1. Set execute permission to the `network_validation.sh` file.

   ```
   chmod +x network_validation.sh
   ```
2. Based on the configuration, run the following commands using `network_validation.sh` script to validate the subnets or the network security groups (NSG) prior to provisioning:

Note:

Make sure that you perform the validations for the configurations as described in Validate Existing Network Setup.

Note:

If you will be enabling secured production mode add the "-z" or "--securemode" argument with a value of "true" to each command below.

- WebLogic subnet:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID>
  ```

- WebLogic NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -a
  <OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server>
  ```

- Bastion subnet:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -b <Bastion_Subnet_OCID>
  ```

- Existing `Bastion Host IP`:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -i <Bastion_Host_IP>
  ```

- Bastion NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -b <Bastion_Subnet_OCID> -
  a <OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server> -n
  <OCID_of_NSG_for_Bastion_Host>
  ```

- Existing `Bastion Host IP` with NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -i <Bastion_Host_IP> -a
  <OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server>
  ```

- Load balancer subnet:

> **Note:**
>
> If you will enable Authentication Using Identity Domains (also referred to as Identity Cloud Service) replace the argument `-l <external_WLS_LB_Port>` with `-c <App Gateway Port>` in each example below.

```
./network_validation.sh -w <WLS_Subnet_OCID> -u
<Load_Balancer_Subnet1_OCID> -l <external_WLS_LB_Port>
```

If you use an availability domain-specific subnet, then you must provide the load balancer subnet 2.

```
./network_validation.sh -w <WLS_Subnet_OCID> -u
<Load_Balancer_Subnet1_OCID> -v <Load_Balancer_Subnet2_OCID> -l
<external_WLS_LB_Port>
```

If you use a public load balancer, then you must provide the load balancer source CIDR.

```
./network_validation.sh -w <WLS_Subnet_OCID> -u
<Load_Balancer_Subnet1_OCID> -l <external_WLS_LB_Port> -j 0.0.0.0/0
```

- Load balancer NSG:

> **Note:**
>
> If you will enable Authentication Using Identity Domains (also referred to as Identity Cloud Service) replace the argument `-l <external_WLS_LB_Port>` with `-c <App Gateway Port>` in each example below.

```
./network_validation.sh -w <WLS_Subnet_OCID> -u
<Load_Balancer_Subnet_OCID> -l <external_WLS_LB_Port> -a
<OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server> -e
<OCID_of_NSG_for_Load_Balancer>
```

If you use a public load balancer, then you must provide the load balancer source CIDR.

```
./network_validation.sh -w <WLS_Subnet_OCID> -u
<Load_Balancer_Subnet_OCID> -l <external_WLS_LB_Port> -j 0.0.0.0/0 -a
<OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server> -e
<OCID_of_NSG_for_Load_Balancer>
```

- File System Storage subnet:

```
./network_validation.sh -w <WLS_Subnet_OCID> -f
<File_System_Storage_Subnet_OCID>
```

- File System Storage NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -a
  <OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server> -e
  <OCID_of_NSG_for_File_System_Storage>
  ```

- JRF domain with OCI Database system:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -d <OCI_DB_OCID> -p
  <OCI_DB_Port>
  ```

- JRF domain with OCI Database system NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -d <OCI_DB_OCID> -p
  <OCI_DB_Port> -a <OCID_of_NSG_for_Admin_Server> -m
  <OCID_of_NSG_for_Managed_Server>
  ```

- JRF domain with an Autonomous Database:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -t <Autonomous_DB_OCID>
  ```

- JRF domain with an Autonomous Database NSG:

  ```
  ./network_validation.sh -w <WLS_Subnet_OCID> -t <Autonomous_DB_OCID> -a
  <OCID_of_NSG_for_Admin_Server> -m <OCID_of_NSG_for_Managed_Server>
  ```

```
network_validation.sh
```

```
example_user@cloudshell:~ (us-phoenix-1)$ ./network_validation.sh -w
<WLS_Subnet_OCID> -f <File_System_Storage_OCID>
ERROR: Port 22 is not open for access by WLS Subnet CIDR [10.0.62.0/24] in
WLS Subnet [<WLS_Subnet_OCID>]
ERROR: Port 9071 is not open for access by WLS Subnet CIDR [10.0.62.0/24] in
WLS Subnet [<WLS_Subnet_OCID>]
WARNING: Exposing the WebLogic administrator port [7001] in the subnet
[{<WLS_Subnet_OCID>}] to the internet [0.0.0.0/0] allows any user to access
the WebLogic console, which is not a recommended practice. Ensure that only a
specific CIDR range can access the WebLogic console.
WARNING: Exposing the WebLogic administrator port [7002] in the subnet
[{<WLS_Subnet_OCID>}] to the internet [0.0.0.0/0] allows any user to access
the WebLogic console, which is not a recommended practice. Ensure that only a
specific CIDR range can access the WebLogic console.
ERROR: TCP Port [111] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
ERROR: TCP Port [2048] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
ERROR: TCP Port [2049] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
ERROR: TCP Port [2050] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
ERROR: UDP Port [111] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
ERROR: UDP Port [2048] is not open in FSS Subnet [<WLS_Subnet_OCID>] for VCN
CIDR [10.0.0.0/16]
```

```
network_validation.sh

example_user@cloudshell:~ (us-phoenix-1)$ ./network_validation.sh -w
<WLS_Subnet_OCID> -e <OCID_of_NSG_for_File_System_Storage> ERROR: Port 22 is
not open for access by WLS Subnet CIDR [10.0.62.0/24] in WLS Subnet
[ocid1.subnet.oc1.phx.aaaaaaaatcel2ytjhgyws7rbfpobfb4vvce636ffepci36xxb33ohj3h
i6dq]
ERROR: Port 22 is not open for access by WLS Subnet CIDR [10.0.62.0/24] in
WLS Subnet [<WLS_Subnet_OCID>]
ERROR: Port 9071 is not open for access by WLS Subnet CIDR [10.0.62.0/24] in
WLS Subnet [<WLS_Subnet_OCID>]
WARNING: Exposing the WebLogic administrator port [7001] in the subnet
[{<WLS_Subnet_OCID>}] to the internet [0.0.0.0/0] allows any user to access
the WebLogic console, which is not a recommended practice. Ensure that only a
specific CIDR range can access the WebLogic console.
WARNING: Exposing the WebLogic administrator port [7002] in the subnet
[{<WLS_Subnet_OCID>}] to the internet [0.0.0.0/0] allows any user to access
the WebLogic console, which is not a recommended practice. Ensure that only a
specific CIDR range can access the WebLogic console.
ERROR: TCP Port [111] is not open in FSS NSG [<FSS_NSG_OCID>] for VCN CIDR
[10.0.0.0/16]
ERROR: TCP Port [2048] is not open in FSS NSG [<FSS_NSG_OCID>] for VCN CIDR
[10.0.0.0/16]
ERROR: TCP Port [2049] is not open in FSS NSG [<FSS_NSG_OCID>] for VCN CIDR
[10.0.0.0/16]
ERROR: TCP Port [2050] is not open in FSS NSG [<FSS_NSG_OCID>] for VCN CIDR
[10.0.0.0/16]
ERROR: UDP Port [111] is not open in FSS NSG [<FSS_NSG_OCID>] for VCN CIDR
[10.0.0.0/16]
```

2 Create a Stack

Learn how to create an Oracle WebLogic Server stack with Oracle WebLogic Server for OCI.

Topics:

- About Creating a Domain
- Create a Domain
- View the Cloud Resources for a Stack

About Creating a Domain

Learn about the options you have when creating a domain with Oracle WebLogic Server for OCI.

You have several options to choose from when you create a domain:

- Billing Type

  With Universal Credits (also called UCM), you are billed for the cost of the Oracle WebLogic Server license (based on OCPUs per hour) in addition to the cost of the compute resources. This option is not available for Oracle WebLogic Server Standard Edition.

  With Bring Your Own License (BYOL), you reuse your existing on-premises Oracle WebLogic Server licenses in Oracle Cloud. You are billed only for the cost of the compute resources.

- Domain Type and Database

  A basic domain does not require an existing database. See Create a Basic Domain.

  A JRF-enabled domain requires access to an existing Oracle Autonomous Database or Oracle Cloud Infrastructure Database (DB System). A JRF-enabled domain includes the Java Required Files (JRF) components, and the database is used to contain the JRF schema.

  See Create a Domain and Create a Database.

- OS Management Hub

  When you create a domain, Oracle WebLogic Server for OCI can create an OS Management Hub Profile and register that profile with the OCI Compute instances created for the domain. You can also select an existing OS Management Hub Profile and have that profile registered with the OCI Compute instances created for the domain. See Listing Profiles in the Oracle Cloud Infrastructure documentation to find an existing profile.

> **Note:**
>
> To prevent a proliferation of new profiles Oracle strongly recommends that you use an existing profile. Once you have created one stack with a new profile, then you can reuse that profile in other stacks. See Listing Profiles in the Oracle Cloud Infrastructure documentation to find the profile created by a stack and get its OCID.

- Secured Production Mode

  When you create a domain, Oracle WebLogic Server for OCI can also set up that domain according to the lockdown guide by selecting Enable Secured Production Mode. Oracle WebLogic Server for OCI creates a WebLogic domain that differs from a domain created without this feature in the following ways:

  - Secured Production Mode is enabled. In secured production mode, your production domain is highly secure because the security configuration defaults are more secure, insecure configuration items are logged as warnings, and default authorization and role mapping policies are more restrictive.
  - An administrative port is set. An administration port limits all administrative traffic between server instances in a WebLogic Server domain to a single port.
  - TLS (SSL) is configured end to end:
    - All t3 and http access points are disabled.
    - A custom identity keystore and a custom trust keystore are created on each node:
      - A new wildcard certificate is generated using the OCI Certificate Service and added to the custom identity keystore on each node.
      - A certificate chain is retrieved from the OCI Certificate Service for the wildcard certificate and populated in the custom trust keystore on each node.
    - The WebLogic Administration Server and all WebLogic Managed Servers have TLS (SSL) configured with Custom Identity and Custom Trust referencing the contents in these keystores.
    - Node Manager TLS (SSL) is configured to use the custom identity store with the wildcard certificate.
    - The load balancer uses the certificate chain to access the TLS external channel.
    - If Authentication Using Identity Domains (aka Identity Cloud Service) is also enabled it uses the same identity and trust keystore contents as the domain.
    - Hostname verification is enabled for all TLS communication.
    - TLS version is set to a minimum of version 1.2.
  - Obvious names such as "system, admin, administrator, or weblogic" for users with Admin role are disallowed.
  - Two administrative users are created.
  - Password requirements for new users are set as follows:
    - Reject if Password Contains the User Name
    - Reject if Password Contains the User Name Reversed
    - Minimum Length: 8 characters
    - Maximum Length: 30 characters

    - Maximum Instances of any Character: 4
    - Maximum Consecutive Characters: 3
    - Mimimum Number of Alphabetic Characters: 1
    - Minimum Number of Numeric Characters: 1
    - Minimum Number of Lower Case: 1
    - Minimum Number of Upper Case: 1
    - Mimimum Number of Non-Alphabetic Characters: 1
    - Mimimum Number of Non-Alphanumeric Characters: 1
  - An Auditing Provider is created for the Default Authentication Provider.
  - The thread pool limit in the Overload configuration is set.
  - By default, boot.properties with encrypted credentials is not created.

    > **Note:**
    >
    > This is an optional setting due to the implications of removing these files. Without a boot.properties:
    >
    > - node manager will not be able to restart servers that shutdown unexpectedly.
    > - Restarting a Compute instance will not automatically start WebLogic servers since node manager is used to revive them. Servers should be restarted by executing the /opt/scripts/restart_domain.sh script and then entering the administration user password.

  > **Note:**
  >
  > JRF domains are not supported in secured production mode on WebLogic Server release 12c (12.2.1.4).

- Virtual Cloud Network (VCN)

  Oracle WebLogic Server for OCI can create a VCN for you when you create a domain, or you can create a VCN before you create the domain. If you create a new VCN, you must specify a contiguous CIDR block of your choice.

  If you create a JRF-enabled domain and select an Oracle Cloud Infrastructure Database (DB System) in a different VCN, then Oracle WebLogic Server for OCI configures local peering between the two VCNs.

- Subnet

  Oracle WebLogic Server for OCI can create a new subnet for the WebLogic Server compute instances, or you can specify a subnet that you have already created. You must specify a CIDR if you create a new subnet.

  If you create a new VCN, you can only create a new regional subnet that spans the entire region.

- Network Access

The subnet for the WebLogic domain can be public or private. If the subnet is private, the nodes cannot be accessed directly from outside of Oracle Cloud. When you create a domain on a private subnet, you can specify a public subnet for the bastion host. Oracle WebLogic Server for OCI creates this compute instance to enable you to administer the WebLogic nodes.

If you already have an existing bastion to provide public access to the compute instances, or if you already have a VPN connection to your on-premise network, then you can delete the bastion created by Oracle WebLogic Server for OCI.

See Create a Basic Domain in a Private Subnet.

> **Note:**
>
> - Configuring a bastion is optional.
>
>   If you do not configure a bastion, no status is returned for provisioning. See Configure a Bastion.
> - It is recommended to not configure a bastion, that is deselect the **Provision Bastion Node on Public Subnet** option, only in network with fast connect setup

- Load Balancer

  When you create a domain, Oracle WebLogic Server for OCI can also create a load balancer to distribute application traffic to the WebLogic cluster. A load balancer consists of primary and standby nodes but it is accessible from a single IP address. If the primary node fails, traffic is automatically routed to the standby node.

  If you use a regional subnet for the WebLogic Server compute instances, use a regional subnet for the load balancer. The regional subnet is shared between both load balancer nodes.

  By default, the load balancer is public. You can also provision a public load balancer with a reserved public IP. If you create a domain in a private subnet, then you can provision a public or private load balancer. A private load balancer does not have a public IP address and cannot be accessed from outside of Oracle Cloud.

  Oracle WebLogic Server for OCI configures the load balancer to use Secure Socket Layer (SSL). A demonstration self-signed certificate is attached to the HTTPS listener. Oracle recommends you add your own SSL certificate to the load balancer after creating the domain. All traffic between the load balancer and compute instances uses HTTP.

- Oracle Cloud Infrastructure Root Policies and Dynamic Group

  When you create a domain, by default Oracle WebLogic Server for OCI creates a dynamic group and one or more root-level (tenancy) policies that allow the compute instances in the domain to access:

  - Keys and secrets in Oracle Cloud Infrastructure Vault
  - Load balancer resources
  - The database wallet if you're using Oracle Autonomous Database to contain the required infrastructure schemas for a JRF-enabled domain
  - The database network resources if you're using Oracle Cloud Infrastructure Database (DB System) to contain the required infrastructure schemas for a JRF-enabled domain

- Authentication

By default, the domain is configured to use the local WebLogic Server identity store to maintain users, groups, and roles. Alternatively, a domain running WebLogic Server can use Oracle Identity Cloud Service to authenticate users.

In order to use Oracle Identity Cloud Service, you must create a domain that includes a load balancer.

- Security List for DB System

  When you create a JRF-enabled domain and use Oracle Cloud Infrastructure Database (DB System) to contain the JRF components, by default Oracle WebLogic Server for OCI creates a security list on the database's VCN that allows the WebLogic Server subnet to access the database.

Create a Domain

Use Oracle WebLogic Server for OCI to create a stack that includes an Oracle WebLogic Server domain, one or more WebLogic Server compute instances, network resources, and an optional load balancer.

▶ Video

Launch a new stack from Marketplace by entering parameters that automatically create a domain.

You can specify a private subnet or a public subnet (either a regional or availability domain-specific) for the domain. In case of private subnet, the compute instances that are assigned to the private subnet are not accessible from the public internet. To access the virtual machines (VMs) created in the private subnet, you must configure a bastion host as described in Configure a Bastion.

If you plan to create a JRF-enabled domain, you specify an autonomous database or DB system database as described in Configure Database Parameters and VCN Peering.

> **Note:**
>
> Oracle WebLogic Server 12c (12.2.1.4.0) or 14c (14.1.2.0.0) must be specified as the version for a JRF-enabled domain if you intend to use an Oracle Autonomous Database.

Before you create a domain, you must first perform the following tasks:

- If you create a secured production domain you must have an existing OCI Certificate Authority (CA). If you do not yet have an OCI Certificate Authority see Create a Certificate Authority.
- Create a compartment for your domain resources, or use the same compartment in which you created the database. See Create a Compartment.
- Create an SSH key. See Create an SSH Key.
- Create an encryption key to use for secrets. See Create an Encryption Key.
- Create secrets for the passwords you want to use for the domain. You will need to select the compartment where you have the secret and the secret that contains the password. See Create Secrets for PasswordsSee Create Secrets for Passwords.

- If you use a private subnet, create a FastConnect or a VPN connection if you want to use your own bastion host to administer your Compute instances. See VPN Connect or FastConnect in the Oracle Cloud Infrastructure documentation.
- If you create a JRF-Enabled domain:
  - Create database in Oracle Autonomous Database or Oracle Cloud Infrastructure Database (DB System). See Create a Database.
  - Identify the pluggable database (PDB) name. This is required only for Oracle Cloud Infrastructure Database (DB System) running Oracle Database 12c or later.

Oracle WebLogic Server for OCI can create the virtual cloud network (VCN) and subnets for your new domain. If you want to use an existing VCN or existing subnets for the domain, then they must meet certain requirements. See:

- Create a Virtual Cloud Network
- Create a Private Subnet for the Oracle WebLogic Server Nodes
- Create a Subnet for the Bastion Node (if you use a private subnet for your existing network)
- Create a Subnet for the Load Balancer (if you want to create a load balancer)

Tutorial (Non-JRF)

Tutorial (using an autonomous database)

Tutorial (using a DB System database)

Topics:

- Launch a Stack
- Specify Stack Information
- Specify Stack Configuration
- Configure Network Parameters
- Configure WebLogic Domain Parameters
- Configure WebLogic Domain Parameters for Secured Production Mode
- Configure Database Parameters and VCN Peering
- Configure WebLogic Compute Instance Parameters
- Configure a Resource Manager Private Endpoint
- Configure a Bastion
- Configure a Load Balancer
- Configure WebLogic Authentication
- Configure Application Performance Monitoring
- Configure Autoscaling
- Configure File System
- Configure Tags
- Create the Domain Stack
- Use Your New Domain

Launch a Stack

Use Marketplace to specify initial stack information.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Marketplace**, and then click **All Applications**.
3. Set the filter type to **Stack**.
4. Select an application that matches the edition of Oracle WebLogic Server that you want to provision, and also uses the type of billing you want (Universal Credits or Bring Your Own License).
    - **Oracle WebLogic Server Standard Edition BYOL**
    - **Oracle WebLogic Server Enterprise Edition BYOL**
    - **Oracle WebLogic Server Enterprise Edition UCM**
    - **Oracle WebLogic Suite BYOL**
    - **Oracle WebLogic Suite UCM**

    > **Note:**
    >
    > If you set the filter type to **Image**, follow the steps in Create an Instance Using the Marketplace in *Images for Oracle WebLogic Server for OCI*.

5. Select a version of Oracle WebLogic Server 12c.

    The latest 14.1.1.0.0 patch level is the default version.
    If multiple builds are available for the same patch level (`.01`, `.02`, `.03`, and so on), choose the latest build.

    > **Note:**
    >
    > To use an Oracle Autonomous Database, you must select a 12c version.

6. Select the compartment in which to create the stack.

    By default the stack compartment is used to contain the domain compute instances and network resources. If later on you specify a network compartment on the Configure Variables page of the Create Stack wizard, then only the compute instances are created in the stack compartment that you select here.

7. Select the **Oracle Standard Terms and Restrictions** check box, and then click **Launch Stack**.

    The Create Stack wizard is displayed.

Specify Stack Information

Specify the name, description, and tags for the stack.

1. On the Stack Information page of the Create Stack wizard, enter a name for your stack.
2. Enter a description for the stack (optional).

3. Specify one or more tags for your stack (optional).
4. Click **Next**.

   The Configure Variables page opens.

Specify Stack Configuration

Specify the stack configuration options to create your domain.

1. Enter the resource name prefix.

   The maximum character length is 16.

   This prefix is used by all the created resources.
2. Enter the SSH public key, by either uploading the SSH key file or pasting the contents of your SSH public key file.
3. Specify the configuration options for the domain by selecting the check boxes, based on your requirements.

   Ensure that you have set up the user group polices and dynamic group polices that are applicable for the different configuration options. See User Group Policies and Dynamic Group Policies.

   The following table lists the configuration options and the prerequisites required for the configuration, wherever applicable.

   > **Note:**
   >
   > The **OCI Polices**, **Use Resource Manager Private Endpoint**, and **Provision Load Balancer** options are selected by default.

   > **Note:**
   >
   > As of release 14c (14.1.2.0.0), **Enable Secured Production Mode** is enabled by default.

Configuration Option	Description	Prerequisite
OCI Policies	When you create a domain, Oracle WebLogic Server for OCI creates a dynamic group and relevant root-level (tenancy) policies for you. If you are not an administrator, the necessary groups and policies must be in place before you can create a domain. Before you deselect the check box, ask your administrator to create the required dynamic group and relevant policies.	To create the required dynamic group and relevant policies, see Create Dynamic Groups and Policies.

Configuration Option	Description	Prerequisite
Create a Virtual Cloud Network	If you select this option, Oracle WebLogic Server for OCI creates a Virtual Cloud Network (VCN) in Oracle Cloud Infrastructure when you create a domain. You can also create your own VCN before creating a domain. In that case, you can either choose to select the existing VCN but create new subnets, or use an existing VCN and existing subnets. So, you need not select the **Create a Virtual Cloud Network** check box. If you create a new VCN, use existing VCN with new subnet, or existing VCN and exisitng subnet, see the following topics to configure the network, WebLogic compute instance, load balancer, and file system storage: • Configure Network Parameters • Configure WebLogic Compute Instance Parameters • Configure a Load Balancer • Configure File System	If you plan to create your own VCN, use existing VCN with new subnet, or existing VCN and exisitng subnet, see the following topics: • Create a Virtual Cloud Network • Create a Private Subnet for the Oracle WebLogic Server Nodes
Use Resource Manager Private Endpoint	You can create a private endpoint on a private subnet, or use an existing private endpoint to check the provisioning status of the resources. However, based on configuration option selected, you can check the provisioning status as follows: • If only **Use Resource Manager Private Endpoint** is selected, you can use the private endpoint to check the provisioning status of the resources. • If only **Provision Bastion Instance** is selected, you can access the WebLogic Server compute instances on a private subnet and check the status of provisioning. • If both **Use Resource Manager Private Endpoint** and **Provision Bastion Instance** are selected, the private endpoint is used to check the provisioning status of the resources. • If both **Use Resource Manager Private Endpoint** and **Provision Bastion Instance** are not selected, then you must check the status of provisioning from `/u01/logs/provisioning.log` on the compute instance. Also any domain creation failures are not reported. The **Use Resource Manager Private Endpoint** is not available when you create a new virtual cloud network, create a new subnet for an existing VCN, or use a public subnet for an existing VCN and existing subnet. To configure a resource manager endpoint, see Configure a Resource Manager Private Endpoint.	If you use an existing private subnet, you must create a private endpoint before creating the domain. See Create a Private Endpoint.

Configuration Option	Description	Prerequisite
Provision Bastion Instance	If you select this option, you can specify a private subnet for the domain and assign the Oracle WebLogic Server compute instances to a private subnet, so the instances are not accessible from the public Internet. To access the virtual machines (VMs) created in the private subnet, a bastion host is required. If you select this option, you can also access the compute instance and check the provisioning status of the resources. However, based on configuration option selected, you can check the provisioning status as follows: • If only **Use Resource Manager Private Endpoint** is selected, you can use the private endpoint to check the provisioning status of the resources. • If only **Provision Bastion Instance** is selected, you can access the WebLogic Server compute instances on a private subnet and check the status of provisioning. • If both **Use Resource Manager Private Endpoint** and **Provision Bastion Instance** are selected, the private endpoint is used to check the provisioning status of the resources. • If both **Use Resource Manager Private Endpoint** and **Provision Bastion Instance** are not selected, then you must check the status of provisioning from `/u01/logs/provisioning.log` on the compute instance. Also any domain creation failures are not reported. However, based on the configuration option that is selected, you can check the provisioning status as follows: • If only **Use Resource Manager Private Endpoint** is selected, you can create a private endpoint on a private subnet, or use an existing private endpoint to check the provisioning status of the resources. • If only **Provision Bastion Instance** is selected, you can specify a private subnet for the domain and assign the Oracle WebLogic Server compute instances to a private subnet, and then check the provisioning status of the resources. so the instances are not accessible from the public Internet, and • If both **Use Resource Manager Private Endpoint** and **Provision Bastion Instance** are selected, the private endpoint is used to check the provisioning status of the resources. • If **Use Resource Manager Private Endpoint** is deselected **Provision Bastion Instance** is selected, then you must check the status of provisioning from `/u01/logs/`	If you choose to create your own subnet for the bastion node before creating the domain, see Create a Subnet for the Bastion Node.

Configuration Option	Description	Prerequisite
	`provisioning.log` on the compute instance. Also any domain creation failures are not reported. You must create a bastion host with a FastConnect or a VPN connection before you create a domain or you must choose to have a bastion host created for you. The **Provision Bastion Instance** is not available when you create a new virtual cloud network, create a new subnet for an existing VCN, or use a public subnet for an existing VCN and existing subnet To configure a bastion, see Configure a Bastion.	
Provision Load Balancer	You can create a load balancer or use an existing load balancer to distribute application traffic to the WebLogic Managed Servers. If you deselect this option, a load balancer is not provisioned in Oracle Cloud Infrastructure when you create a domain. To configure a load balancer, see Configure a Load Balancer.	If you use existing subnets for the domain, you must create subnets for the load balancer before creating your domain. See Create a Subnet for the Load Balancer. If you use an existing load balancer for an existing VCN and an existing subnet, you must configure the load balancer. See Configure the Load Balancer in *Before you Begin*..
Enable Authentication Using Identity Cloud Service	You have the option to use Oracle Identity Cloud Service to authenticate application users for your domain. If you enable authentication using IDCS, the load balancer is automatically provisioned even if you deselect the **Provision Load Balancer** check box. To use Oracle Identity Cloud Service for authentication, see Configure WebLogic Authentication.	Before you enable authentication using Identity Cloud Service, you must create a confidential application, and then identify its client ID and client secret. See Create a Confidential Application.
Enable OS Management Hub	You have the option to register the OCI Compute instances with OS Management Hub. You can have Oracle WebLogic Server for OCI create an OS Management Hub Profile for you or use an existing profile.	If you choose supply an existing profile you must provide the OCID for the profile. See Listing Profiles in the Oracle Cloud Infrastructure documentation to find the profile created by a stack and get its OCID.
Enable Exporting Logs to OCI Logging Service	If you enable logging for the WebLogic instances, you can view the server logs such as errors or warnings in the Oracle Cloud Infrastructure console.	

Configuration Option	Description	Prerequisite
Enable Application Performance Monitoring	If you enable Application Performance Monitoring, you can export WebLogic metrics using Application Performance Monitoring (APM) Java Agent and create dashboards with WebLogic specific metrics. To monitor the application performance service, see Configure Application Performance Monitoring.	Before you enable Application Performance Monitoring, you must create an application performance monitoring domain. See Create an Application Performance Monitoring Domain.
Enable Autoscaling	If you enable autoscaling, you can scale out or scale in instances. When you enable autoscaling, Application Performance Monitoring is enabled for the WebLogic instance even if you do not select the **Enable Application Performance Monitoring** check box. To configure autoscaling, see Configure Autoscaling.	
Add File System	You have the option to create a file system and mount target in Oracle Cloud Infrastructure for an Oracle WebLogic Server domain. You can also use an existing file system and mount target for the domain. To configure file storage, see Configure File System.	If you use existing subnets for the domain, you must create subnets for the mount target before creating your domain. See Create a Subnet for the Mount Target.
Enable Secured Production Mode	If you enable Secured Production Mode, Oracle WebLogic Server for OCI sets up that domain according to the Lock Down WebLogic Server in the *Securing a Production Environment for Oracle WebLogic Server*. See About Creating a Domain for more details on what is configured for this option.	Before you enable Secured Production Mode, you must have an existing OCI Certificate Authority (CA). If you do not yet have an OCI Certificate Authority see Create a Certificate Authority.

Configure Network Parameters

Define the Virtual Cloud Network (VCN) and subnet configuration for the domain in the Virtual Cloud Networking section.

Based on your configuration in the Stack Configuration section, you can configure network parameters for an existing VCN or a new VCN.

- If you select **Create a Virtual Cloud Network** option, see Configure Network for New VCN.
- If you do not select **Create a Virtual Cloud Network** option, see Configure Network for Existing VCN.

> **Note:**
>
> During provisioning, by default, the existing network is validated using the validation script, `network_validation.sh`. See Validate Existing Network Setup.

> **Note:**
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies.

Configure Network for New VCN

Specify the parameters to configure a new VCN.

1. Select the Network Compartment in which to create the network resources for this domain. If you don't specify a network compartment, then all the network resources and the domain compute instances are created in the stack compartment that you selected earlier upon launching the stack. Select a network compartment if you want the network resources to be in a different compartment than the compute instances.
2. Enter a name and CIDR for the new VCN.

Configure Network for an Existing VCN

1. Select the Network Compartment in which to create the network resources for this domain. If you don't specify a network compartment, then all the network resources and the domain compute instances are created in the stack compartment that you selected earlier upon launching the stack. Select a network compartment if you want the network resources to be in a different compartment than the compute instances.
2. Select the name of the existing VCN.
3. Specify the subnet configuration as follows:
   - For existing VCN and new subnets:
     a. Select **Create New Subnets**.
   - For existing VCN and existing subnets:
     a. Do not select the **Create New Subnets** option.
     b. Select the **Subnet Compartment** to use for the existing subnet.
        The subnet compartment is different than the VCN compartment. The subnets for the WebLogic Server nodes, load balancer and the bastion node use this same subnet compartment.

        > **Note:**
        >
        > You can specify the subnet compartment only if you're using an existing subnet.

     c. Keep the default **Use Private Subnet** selection.
        Compute instances in a private subnet are not directly accessible from outside of Oracle Cloud. So, you must configure a bastion compute instance on a public subnet. See Configure a Bastion.

        You can also select **Use Public Subnet** for the domain.

        For a public subnet, ensure to limit the CIDR range to access WebLogic administration console ports, the default ports are `7001` and `7002` for `http` and `https` respectively.

> **Note:**
>
> The **Provision Bastion Instance** option is not available when you use a public subnet.

d. Select the regional subnet or availability domain-specific subnet.

e. Select **Existing Network Security Groups** to use existing Network Security Groups (NSGs) in the VCN, to secure your resources created by the stack.
You can specify network security groups for administration server node, managed server node, bastion instance, load balancer, and file storage.

Configure WebLogic Domain Parameters

Specify the parameters needed to configure the WebLogic domain.

> **Note:**
>
> If you set Secured Production Mode Enabled skip this section and refer to Configure WebLogic Domain Parameters for Secured Production Mode.

1. Enter a user name for the WebLogic Server administrator.
2. Select the compartment where you have the WebLogic Server administration secret and then select the secret that contains the administration password. To create secrets, see Create Secrets for Passwords.
3. Select the JDK version for Oracle WebLogic Server.

   > **Note:**
   >
   > Supported JDK versions and defaults vary by release.
   >
   > - 12.2.1.4 - only supports JDK8 (no option to select JDK)
   > - 14.1.1.0 - JDK8 or JDK11 (default is JDK8)
   > - 14.1.2.0 - JDK17 or JDK21 (default is JDK17)

4. Select **Provision with JRF** to configure the domain with JRF components.

   See Configure Database Parameters.
   If you are unsure of selecting this option, see JRF Domain.
5. Keep the default selection for **Deploy Sample Application**, if you want to install a sample web application to the WebLogic cluster, or Identity Cloud Service (IDCS) protected sample application, if you enabled IDCS.

   This option is not displayed if you selected WebLogic Server Standard Edition.
6. Specify the WebLogic startup arguments.

   You can use the server startup arguments to provide arguments to the Java Virtual Machine for WebLogic Server instances. When the servers are scaled out, any changes to the server startup arguments applies to the added nodes only.

For example, to configure memory settings, you can specify the arguments: *-Xms1024m -Xmx1024m*

7. Select **Configure Ports** to change the default port numbers.

   You can change the default ports for the WebLogic Server Node Manager, WebLogic Server Administration Console, WebLogic Server Administration Console SSL, WebLogic Cluster, WebLogic Managed Server External, and WebLogic Managed Server External SSL.

Configure WebLogic Domain Parameters for Secured Production Mode

Specify the parameters needed to configure the WebLogic domain in secured production mode.

1. Enter a user name for the WebLogic Server administrator. The name cannot be system, admin, administrator, or weblogic.
2. Select the compartment where you have the following secrets:
   a. The secret that contains the password for the primary WebLogic administration user.
   b. The secret that contains the password for the secondary WebLogic administration user.
   c. The secret to be used for accessing a PKCS12 keystore where TLS (SSL) keys and certificates will be added.
3. Select the secret that contains the password for the primary administrator user. To create secrets, see Create Secrets for Passwords.
4. Enter the name of the secondary WebLogic administration user. The name cannot be system, admin, administrator, or weblogic.
5. Select the secret that contains the password for the secondary administrator user. To create secrets, see Create Secrets for Passwords.
6. Select the secret that will be used for accessing a PKCS12 keystore.
7. Enter the OCID for a Certificate Authority. If you do not yet have an OCI Certificate Authority see Create a Certificate Authority.
8. Select whether a boot.properties file with encrypted passwords should be preserved. Note:

   If not selected then Oracle node manager will not be able to restart servers that shutdown unexpectedly and restarting a Compute instance will not automatically start WebLogic servers. The default is unselected since the Lock Down WebLogic Server in the *Securing a Production Environment for Oracle WebLogic Server* recommends not preserving boot.properties.
9. Select the JDK version for Oracle WebLogic Server.

> **Note:**
>
> Supported JDK versions and defaults vary by release.
>
> - 12.2.1.4 - only supports JDK8 (no option to select JDK)
> - 14.1.2.0 - JDK17 or JDK21 (default is JDK17)

10. Keep the default selection for Deploy Sample Application, if you want to install a sample web application to the WebLogic cluster, or Identity Cloud Service (IDCS) protected sample application, if you enabled IDCS.

    This option is not displayed if you selected WebLogic Server Standard Edition.

    For 14.1.2.0.0, select Provision with JRF to configure the domain with JRF components. See Configure Database Parameters.

11. Specify the WebLogic startup arguments.You can use the server startup arguments to provide arguments to the Java Virtual Machine for WebLogic Server instances. When the servers are scaled out, any changes to the server startup arguments applies to the added nodes only.
    For example, to configure memory settings, you can specify the arguments: `-Xms1024m -Xmx1024m`

12. Enter a value for the capacity of work managers (Throttle the Thread Pool). This is the total number of requests that can be present in a server. This includes requests that are enqueued and those under execution.

13. Select Configure Ports to change the default port numbers.

    You can change the default ports for the WebLogic Server Node Manager, WebLogic Server Administration (domain wide and managed servers), and WebLogic Managed Server External SSL.

Configure Database Parameters and VCN Peering

You must specify a database in Oracle Autonomous Database or Oracle Cloud Infrastructure Database (DB System) when you create an Oracle WebLogic Server for OCI domain that includes the Java Required Files (JRF) components.

The database you specify is used to create the required infrastructure schemas for the JRF-enabled domain.

> **Note:**
>
> For each schema that is created in the database, a data source is created in WebLogic Server. These data sources should not be used by applications deployed to the WebLogic domain after provisioning is complete. Instead, you must create independent data sources. See Manage Data Sources.

If you selected the option to create a new VCN, or selected the option to use an existing VCN with a new subnet for the WebLogic Server compute instances and the Oracle Cloud Infrastructure Database(DB System), or for the WebLogic Server compute instances and the Oracle Autonomous Database, Oracle WebLogic Server for OCI configures local peering in your VCNs, if your database is on a different VCN than the VCN you want to use for WebLogic Server.

> **Note:**
>
> In case of Autonomous Database, the database must use a private endpoint.

If you use an existing VCN and existing subnet, you must peer the VCNs manually before you create the stack. See Manual VCN Peering.

If your databases and domain are in different VCNs, then Oracle WebLogic Server for OCI configures local peering between the two VCNs. To support VCN peering, you must perform the following prerequisite tasks before creating a stack:

- Create an local peering Gateway (LPG) in the database VCN.
- Add a route to the current route table of the database subnet, to direct traffic to the CIDR of the WebLogic subnet to the LPG.
- Open the database port to the WebLogic subnet CIDR.
- If you use an existing VCN and new subnets, you must add the default private view of the database VCN to the DNS resolver of the existing VCN.
  See Add a DNS view to the DNS Resolver.

In the WebLogic Domain Configuration section, select the **Provision with JRF** check box to display the Database options.

- Configure Autonomous Database
- Configure Oracle Cloud Infrastructure Database

> **Note:**
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies.

Configure Autonomous Database

To configure the database parameters:

1. Select **Autonomous Transaction Processing Database** from **Database Strategy** list.
2. Select the compartment in which you've created the database.
3. Select the database where you want to create the JRF schemas for this WebLogic domain.
4. Select the compartment where you have the autonomous database secret and then select the secret that contains the administration user password in the autonomous database. To create secrets, see Create Secrets for Passwords.
5. Select the service level that the domain should use to connect to the selected autonomous database.
6. If your database uses private endpoint, then select the **Database uses private endpoint** check box and specify the following:
   a. The compartment in which the autonomous database's VCN is found.
   b. The VCN on which you've created the autonomous database. If this VCN is different than the WebLogic Server VCN, they cannot have overlapping CIDRs. For example, you cannot create a domain on VCN `10.0.0.0/16` that uses a database on VCN `10.0.0.1/24`.
   c. Specify the OCID of the local peering gateway in the database VCN to be peered with the WebLogic VCN.

   Oracle WebLogic Server for OCI creates a rule to the existing network security group for the autonomous database with private endpoints. This security list allows the WebLogic Server subnet to access the database port. If this step isn't required or you don't have the correct permissions to modify the database network, clear the **Create Database Security List** check box.

Configure Oracle Cloud Infrastructure Database

To configure the database parameters:

1. Select **Database System** from **Database Strategy** list.
2. Select the DB system to use for this WebLogic domain.
3. Select the compartment in which the database's VCN is found.
4. Select the VCN on which you've created the database. If this VCN is different than the WebLogic Server VCN, they cannot have overlapping CIDRs. For example, you cannot create a domain on VCN `10.0.0.0/16` that uses a database on VCN `10.0.0.1/24`.
5. Select the database home within the selected DB system.
6. Select the database home version.
7. Select the database within the selected DB system where you want to create the JRF schemas for this domain.
8. Specify the Pluggable database (PDB) name, only if the selected database is running Oracle Database 12c or later.
9. Specify the name of a database user with SYSDBA privileges.
10. Select the compartment where you have the database administration secret and then select the secret that contains the database administration password. To create secrets, see Create Secrets for Passwords.
11. Specify the database listen port (1521 by default)
12. Specify the OCID of the local peering gateway in the database VCN to be peered with the WebLogic VCN.

    > **Note:**
    >
    > This option is not applicable if you're using an existing VCN and existing subnet.

13. If you want to use a connection string, select the **Use Database Connection String** check box and enter the following:
    a. The connection string to connect to the database.

    > **WARNING:**
    >
    > Do not use the database connection string example provided in the **Oracle Database Connection String** field , instead use the format specified in the following table.

Table 2-1 Database Connect String for Database Version and Type

Database Version	Database Type	Database Connection String
12c and above	VM	`//<db_hostname>-scan.<db_domain>:<db_port>/<pdb_name>.<db_domain>`
12c and above	Bare Metal	`//<db_hostname>.<db_domain>:<db_port>/<pdb_name>.<db_domain>`

b. The pluggable database (PDB) name.
The PDB name ensures that you receive the notification about the Oracle Platform Security Services (OPSS) schema password expiry date. See Unable to Fetch the Password Expiry Date for the OPSS User.

14. Specify the name of a database user with SYSDBA privileges.

15. Select the compartment where you have the database secret and then select the secret that contains the password of the SYSDBA user. To create secrets, see Create Secrets for Passwords.

> **Note:**
>
> Oracle recommends you to use the connection string for Exadata Database systems.
>
> If you use database connection string, then Oracle WebLogic Server for OCI creates a single instance datasource. However, you can update the data source for Oracle WebLogic Suite with Active GridLink data source and data source for Oracle WebLogic Server Enterprise Edition with multi data source. See Configuring Active GridLink Connection Pool Features and Configuring JDBC Multi Data Sources.
>
> If using **Database System** with connection string, security list is not created to access the database. You must ensure that the ports are open to access the database.

Oracle WebLogic Server for OCI creates a security list in the VCN on which you've created the database. This security list allows the WebLogic Server subnet to access the database port. If this step isn't required or you don't have the correct permissions to modify the database network, clear the **Create Database Security List** check box.

Manual VCN Peering

To peer the VCNs manually, perform the following steps:

1. Create a local peering gateway (LPG) for the VCN that contains the WebLogic Server VM.
   a. Sign in to the Oracle Cloud Infrastructure Console.
   b. From the navigation menu ☰, click **Networking**, and then click **Virtual Cloud Networks**.
   c. From the list of Virtual Cloud Networks, click the name of the WebLogic Server VCN.
   d. Under **Resources**, click **Local Peering Gateways**.

   e. Click **Create Local Peering Gateway**.

   f. Enter a name for the LPG and select the compartment where you want to create the LPG, if you want the LPG to be created in a different compartment.

   g. Click **Show Advanced Options** and associate a route table and configure tags, if required.

   h. Click **Create Local Peering Gateway**.
      The LPG is then created and displayed on the **Local Peering Gateways** page in the compartment you chose.

2. Create a local peering gateway (LPG) for the VCN that contains your database.

   a. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Virtual Cloud Networks**.

   b. From the list of Virtual Cloud Networks, click the name of the database VCN.

   c. Under **Resources**, click **Local Peering Gateways**.

   d. Click **Create Local Peering Gateway**.

   e. Enter a name for the LPG and select the compartment where you want to create the LPG, if you want the LPG to be created in a different compartment.

   f. Click **Show Advanced Options** and associate a route table and configure tags, if required.

   g. Click **Create Local Peering Gateway**.
      The LPG is then created and displayed on the **Local Peering Gateway**s page in the compartment you chose.

3. On the **Local Peering Gateways** page, establish a connection between the LPGs created in Steps 1 and 2.

   a. For the LPG created for the VCN, click the Actions icon ⋮, and then click **Establish Peering Connection**.

   b. Select **Enter Local Peering Gateway OCID**, and enter the database LPG's OCID.

   c. Click **Establish Peering Connection**.
      The connection is established and the LPG's state changes to *PEERED*.

4. Configure the route table in the database subnet to enable traffic to flow to the WebLogic Server subnet.

   a. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Virtual Cloud Networks**.

   b. From the list of Virtual Cloud Networks, click the name of the database VCN.

   c. Under **Resources**, click **Route Tables**.

   d. Click the route table to which you want to add route rule.

   e. Click **Add Route Rules** and specify the following:

      - From the **Target Type** list, select *Local Peering Gateway*.
      - For **Destination CIDR Block**, enter the WebLogic Server VCN's CIDR block. If you want, you can specify a subnet or particular subset of the peered VCN's CIDR.
      - Select the compartment where the LPG is located, if not the current compartment.
      - (*Optional*): Enter a **Description** of the rule.

   f. Click **Add Route Rules**.

5. Configure the route table in the WebLogic Server subnet to enable traffic to flow to the WebLogic Server subnet.

   a. In the Oracle Cloud Infrastructure Console, from the navigation menu ☰, click **Networking**, and then click **Virtual Cloud Networks**.

   b. From the list of Virtual Cloud Networks, click the name of the WebLogic Server VCN.

   c. Under **Resources**, click **Route Tables**.

   d. Click the route table to which you want to add route rule.

   e. Click **Add Route Rules** and specify the following:

      - From the **Target Type** list, select *Local Peering Gateway*.
      - For **Destination CIDR Block**, enter the database VCN's CIDR block. If you want, you can specify a subnet or particular subset of the peered VCN's CIDR.
      - Select the compartment where the LPG is located, if not the current compartment.
      - (*Optional*): Enter a **Description** of the rule.

   f. Click **Add Route Rules**.

See Local VCN Peering in the Oracle Cloud Infrastructure documentation.

Add a DNS view to the DNS Resolver

If you use existing VCN with new subnets, or existing VCN and existing subnets, then for VCN peering, you must resolve the hosts in the database VCN using the WebLogic Server VM by adding the default private view of the database VCN to the DNS resolver of the WebLogic Server VCN.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu ☰, click **Networking**, and then click **Virtual Cloud Networks**.
3. From the list of Virtual Cloud Networks, click the name of the WebLogic Server VCN.
4. On the **VCN Information** tab, click the name of the **DNS Resolver** for the VCN.
5. From the **Associated Private Views** section, click **Manage Private Views**.
6. In the **Private Manage Views** window, select the **Private View** from the compartment where the private view of the private DNS zone is located.
   If a **Private View** is already associated with the resolver, click **Additional Private View** to select the private view of the private DNS zone.
7. Click **Save Changes**.

Configure WebLogic Compute Instance Parameters

Specify the parameters needed to configure the WebLogic instance domain.

1. Select the WebLogic Server shape for the compute instances.

> **Note:**
>
> In regional subnets, select the WebLogic Server shape that has sufficient service limits for an availability domain, else the provisioning fails.

2. For the flexible shapes, select the OCPU count for compute instances.

   > **Note:**
   >
   > You can specify the OCPU count only for the flexible shapes. The memory, network bandwidth, and number of Virtual Network Interface Cards (VNICs) scale proportionately with the number of OCPUs.

3. Select the number of managed servers you want to create. For 12c and 14c versions and all the editions, you can specify up to `8` nodes, which can be scaled out to 30 when you edit the domain.

   The managed servers will be members of a cluster, unless you selected WebLogic Server Standard Edition.

4. *Optional*: If you are using a regional subnet, then from the **WebLogic Administration Sever Availability Domain**, select the availability domain in which you want to create the WebLogic administration server compute instance. If you do not select an availability domain, then by default, the compute instance is created in availability domain 1.

5. For the WebLogic Server subnet, specify one of the following:

   - If you want to use an existing regional subnet, then choose the name of an existing regional subnet from the list of regional and availability domain-specific subnets.
   - If you are creating a new regional subnet, specify a CIDR for the new subnet.

     > **Note:**
     >
     > This is applicable only if you are creating a new VCN or using an existing VCN with new subnets.

6. If you are using existing network security groups (NSGs) for an existing subnet, specify the NSG that is assigned to the virtual machine of the administration server node and the managed server node.

Configure OS Management Hub Profile

Specify the parameters needed to register an OS Management Hub Profile with WebLogic Server Compute Instances.

1. Select whether an existing profile should be used.
2. If an existing profile is selected, enter the OCID for the profile.

> **Note:**
>
> See Listing Profiles in the Oracle Cloud Infrastructure documentation to find the profile created by a stack and get its OCID.

3. If an existing profile is not selected choose the compartment where you want to create a new profile.
4. Enter a name for the profile.

WebLogic Cloud Steps

This library groups steps that can be used for content reference (conref). Each element that you want to conref to should be in a separate note <element>.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Marketplace**, and then click **All Applications**.
3. Set the filter type to **Stack**.
4. Select an application that matches the edition of Oracle WebLogic Server that you want to provision, and also uses the type of billing you want (Universal Credits or Bring Your Own License).
   - **Oracle WebLogic Server Standard Edition BYOL**
   - **Oracle WebLogic Server Enterprise Edition BYOL**
   - **Oracle WebLogic Server Enterprise Edition UCM**
   - **Oracle WebLogic Suite BYOL**
   - **Oracle WebLogic Suite UCM**

   > **Note:**
   >
   > If you set the filter type to **Image**, follow the steps in Create an Instance Using the Marketplace in *Images for Oracle WebLogic Server for OCI*.

5. Select an Oracle WebLogic Server version.
   - **WLS 12.2.1.3.<patch_number>** (Oracle WebLogic Server 12c)
   - **WLS 10.3.6.0.<patch_number>(11.1.1.7)** (Oracle WebLogic Server 11g)
6. Select the compartment in which to create the stack.

   By default the stack compartment is used to contain the domain compute instances and network resources. If later on you specify a network compartment on the Configure Variables page of the Create Stack wizard, then only the compute instances are created in the stack compartment that you select here.
7. Select the **Oracle Standard Terms and Restrictions** check box, and then click **Launch Stack**.

   The Create Stack wizard is displayed.
8. On the Stack Information page of the Create Stack wizard, enter a name for your stack.
9. Enter a description for the stack (optional).

10. Specify one or more tags for your stack (optional).
11. Click **Next**.

    The Configure Variables page opens.
12. On the Configure Variables page of the Create Stack wizard, enter the region where you want to create your domain.
13. In the WebLogic Server Instance section, enter the resource name prefix.

    The maximum character length is 16.

    This prefix is used by all the created resources.
14. Select the WebLogic Server shape for the compute instances.

    > **Note:**
    >
    > In regional subnets, select the WebLogic Server shape that has sufficient service limits for an availability domain, else the provisioning fails.
15. For the flexible shapes, select the OCPU count for compute instances.

    > **Note:**
    >
    > You can specify the OCPU count only for the flexible shapes. The memory, network bandwidth, and number of Virtual Network Interface Cards (VNICs) scale proportionately with the number of OCPUs.
16. Enter the SSH public key, by either uploading the SSH key file or pasting the contents of your SSH public key file.
17. Select the number of managed servers you want to create. For 12c and 14c versions and all the editions, you can specify up to `8` nodes, which can be scaled out to 30 when you edit the domain.

    The managed servers will be members of a cluster, unless you selected WebLogic Server Standard Edition.
18. Enter a user name for the WebLogic Server administrator.
19. Select the compartment where you have the WebLogic Server administration secret and then select the secret that contains the administration password. To create secrets, see Create Secrets for Passwords.
20. Select the JDK version for Oracle WebLogic Server.

    > **Note:**
    >
    > Supported JDK versions and defaults vary by release.
    >
    > - 12.2.1.4 - only supports JDK8 (no option to select JDK)
    > - 14.1.1.0 - JDK8 or JDK11 (default is JDK8)
    > - 14.1.2.0 - JDK17 or JDK21 (default is JDK17)

21. Select **WLS Instance Advanced Configuration** if you want to change the default port numbers, or remove the sample application.

    Cluster-related parameters are not applicable if you selected WebLogic Server Standard Edition.

22. Select the **Network Compartment** in which to create the network resources for this domain.

    If you don't specify a network compartment, then all the network resources and the domain compute instances are created in the stack compartment that you selected earlier upon launching the stack. Select a network compartment if you want the network resources to be in a different compartment than the compute instances.

23. Enter a name and CIDR for the new VCN.

24. Select the name of the existing VCN.

25. Specify the subnet configuration as follows:

    - For existing VCN and new subnets:

      a. Select Create **New Subnet**.

    - For existing VCN and existing subnet:

      a. Do not select the **Create New Subnet** option.

      b. Enter *YES* to acknowledge if you validated the existing network. To validate a network, see Validate Existing Network Setup.

      c. Select the **Subnet Compartment** to use for the existing subnet.
         The subnet compartment is different than the VCN compartment. The subnets for the WebLogic Server nodes, load balancer and the bastion node use this same subnet compartment.

         > **Note:**
         >
         > You can specify the subnet compartment only if you're using an existing subnet.

      d. Select **Use Public Subnet**.

      e. Select the regional subnet or availability domain-specific subnet.

      f. Select **Existing Network Security Groups** to use existing Network Security Groups (NSGs) for the subnets.
         You can use existing network security groups for administration server node, managed server node, bastion instance, load balancer, and file system.

26. Select a Virtual Cloud Network (VCN) strategy:

    - Select **Use Existing VCN**, and then select the name of the existing VCN.
    - Select **Create New VCN**, and then enter a name and CIDR for the new VCN.

27. Select one of the following subnet strategies:

    - Select **Use Existing Subnet**.
    - Select **Create New Subnet**.

> **Note:**
>
> If you're creating a new VCN, you can only create a new regional subnet.

28. Select the **Subnet Compartment** to use for the existing subnet.

    The subnet compartment is different than the VCN compartment. The subnets for the WebLogic Server nodes, load balancer and the bastion node use this same subnet compartment.

    > **Note:**
    >
    > You can specify the subnet compartment only if you're using an existing subnet.

29. Keep the default **Use Public Subnet** selection.
30. For the WebLogic Server subnet, specify one of the following:
    - If you want to use an existing regional subnet, then choose the name of an existing regional subnet from the list of regional and availability domain-specific subnets.
    - If you are creating a new regional subnet, specify a CIDR for the new subnet.

      > **Note:**
      >
      > This is applicable only if you are creating a new VCN or using an existing VCN with new subnets.

31. *Optional*: If you are using a regional subnet, then from the **WebLogic Administration Sever Availability Domain**, select the availability domain in which you want to create the WebLogic administration server compute instance. If you do not select an availability domain, then by default, the compute instance is created in availability domain 1.
32. If your want to use a bastion compute instance with a reserved public IP, then select **Assign Reserved Public IP to Bastion Instance**.
33. Select **Add Load Balancer**, if not already selected.

    This option is selected by default.

34. Configure the load balancer network.
    - If you chose to use an existing regional subnet for WebLogic Server, then select an existing regional subnet from the list of regional and availability domain-specific subnets. A load balancer can have only one regional subnet, which is shared between both nodes.

      > **Note:**
      >
      > This option is not applicable if you use an existing load balancer.

    - If you chose to create a regional subnet for WebLogic Server, then specify a CIDR for the new load balancer subnet.
35. Configure the load balancer:

- If you chose to create a new Virtual Cloud Network (VCN) or use an existing VCN and a new subnet:
    a. Select **Create New Load Balancer**.
- If you chose to use an existing VCN and an existing subnet, you can do one of the following:
    a. Select **Create Load Balancer**.
    b. Select **Use Existing Load Balancer**.

    > **Note:**
    >
    > The existing load balancer should be in the same compartment as the stack compartment, and should be in the same VCN as the existing VCN chosen for the stack.

        i. Specify the OCID for the existing load balancer.
        ii. Enter the name of the backend set for the existing load balancer that has a routing policy associated with the backend set. The backend set should not have any backends.
        See Configure the Load Balancer.
- If you create a new load balancer, select **Load Balancer with Reserved Public IP**, if you want to use a public load balancer with a reserved public IP. Then, specify the OCID of the public IP for the load balancer
- If you create a new load balancer, select a minimum and maximum flexible load balancer shape.
  By default, the minimum bandwidth size is set to 10Mbps and maximum to 400Mbps.

  > **Note:**
  >
  > You can update the shape to a maximum of 8000Mbps. Before you select the maximum bandwidth, ensure to check the available service limit for the flexible load balancer bandwidth.

36. Configure load balancer:
    - If you chose to create a new Virtual Cloud Network (VCN), or use an existing VCN and a new subnet:
        a. Select **Create New Load Balancer**.
    - If you chose to use an existing VCN and an existing subnet, you can do one of the following:
        a. Select **Create Load Balancer**.
        b. Select **Use Existing Load Balancer**.

> **Note:**
>
> The existing load balancer should be in the same compartment as the stack compartment, and should be in the same VCN as the existing VCN chosen for the stack.

i. Specify the OCID for the existing load balancer.

ii. Enter the name of the backend set for the existing load balancer that has a routing policy associated with the backend set. The backend set should not have backends.
See Configure the Load Balancer.

- If you create a new load balancer, select **Private Load Balancer**, if you do not want to assign a public IP address to the load balancer.
This option is available only if you use a private subnet for WebLogic Server. You cannot create a load balancer in a private subnet.
- If you create a new load balancer, select **Load Balancer with Reserved Public IP**, if you want to use a public load balancer with a reserved public IP. Then, specify the OCID of the public IP for the load balancer.
- If you create a new load balancer, select a minimum and maximum flexible load balancer shape.
By default, the minimum bandwidth size is set to 10Mbps and maximum to 400Mbps.

> **Note:**
>
> You can update the shape to a maximum of 8000Mbps. Before you select the maximum bandwidth, ensure to check the available service limit for the flexible load balancer bandwidth.

37. If you create a new load balancer, select a minimum and maximum flexible load balancer shape.

    By default, the minimum bandwidth size is set to 10Mbps and maximum to 400Mbps.

    > **Note:**
    >
    > You can update the shape to a maximum of 8000Mbps. Before you select the maximum bandwidth, ensure to check the available service limit for the flexible load balancer bandwidth.

38. Select **Load balancer with reserved public IP**, if you want to use a public load balancer with a reserved public IP. Then, specify the OCID of the public IP for the load balancer.
39. Select **Prepare Load Balancer for HTTPS** if you want the load balancer to listen on the HTTPS port. You must add your SSL certificate to the load balancer after creating the domain.
40. Select **Enable Authentication Using Identity Cloud Service**.
41. Enter your Oracle Identity Cloud Service (IDCS) tenant name, which is also referred to as the instance ID.

This ID is typically found in the URL that you use to access Oracle Identity Cloud Service, and has the format `idcs-<GUID>`.

42. Enter the client ID of an existing confidential application in this Oracle Identity Cloud Service instance.
43. Select the compartment where you have the IDCS secret and then select the secret that contains the client secret of the existing confidential application. To create secrets, see Create Secrets for Passwords.
44. If necessary, you can override the default domain name and port that you use to access Oracle Identity Cloud Service, or the default port that is used for the App Gateway software appliance.
45. In the Database section of the Configure Variables page, for **Database Strategy**, select **No Database** if you are creating a basic domain.
46. In the **Key Management Service Key ID** field of the Key Management Service Configuration section of the Configure Variables page, enter the OCID of your encryption key.
47. In the **Key Management Service Cryptographic Endpoint** field, enter the endpoint URL for the vault that contains your encryption key.
48. Select **Create Tags**.
49. To assign a free-form tag, enter the **Tag Key** and **Value**.

    Free-form tag keys and values are case sensitive. For example, `costcenter` and `CostCenter` are treated as different tags.
50. To assign an existing tag, for **Tag Namespace**, select a defined tag, and then select the **Tag Key** and **Value**. If no value is displayed for the **Tag Key**, enter the **Value**.
51. Click **Additional Tag** to assign additional free-form or defined tags.
52. At the bottom of the Configure Variables page, click **Next**.
53. On the Review page of the Create Stack wizard, review the information you have provided, and then click **Create** to create your domain stack.
54. On the Stacks Details page, click **Terraform Actions**, then click **Plan**. When prompted for confirmation, click **Plan**.
55. Wait for the state of the plan job to change to `Succeeded`.
56. Click **Terraform Actions**, then click **Apply**. When prompted for confirmation, click **Apply**.

    When the state of the apply job changes to `Succeeded`, your domain instance and related resources are created.
57.
58. Clone both the middleware and data block volumes of the `Original instance`. See Cloning a Volume.

    > **Note:**
    >
    > Follow the next steps, to first attach the middleware volume and then attach the data block volume to the `Cloning instance`. This sequence ensures that the `mountVolume.sh` script works as desired, which you will be running later in this procedure.
59. **Attach Block Volume** steps

a. In the instance page, under Resources, click **Attached Block Volume** > **Attach Block Volume**.

b. In the **Attach Block Volume** page, select the compartment where you cloned the block volume earlier.

c. Click the drop-down and select the block volume you clone earlier.

d. Select the Attachment Type as **ISCSI**.

e. Click **Attach**.
An `Attach Block Volume` message appears.

f. Click **Close**.

60. Run the following script:

```
/opt/scripts/cloning/mountVolume.sh -m <cloned-middleware-volume-name> -d
<cloned-data-volume-name>
```

This script runs the iSCSI commands to attach both the middleware and data volumes. Also, the script updates the UUID entries of both the volumes in `/etc/fstab`, which ensures that the mount is persistent across reboot.

61. Update the metadata on each node by using `update_metadata.py` script, which is located at `/opt/scripts/utils/`:

We need to update the metadata for reboot to work correctly. This also starts the node manager and administration server automatically after every restart. There are 5 domain related values in metadata that need to be updated for reboot to work.

```
python3 /tmp/update_metadata.py -k wls_domain_name -v
<resource_prefix>_domain
python3 /tmp/update_metadata.py -k wls_machine_name -v
<resource_prefix>_machine_
python3 /tmp/update_metadata.py -k wls_cluster_name -v
<resource_prefix>_cluster
python3 /tmp/update_metadata.py -k wls_admin_server_name -v
<resource_prefix>_adminserver
python3 /tmp/update_metadata.py -k wls_ms_server_name -v
<resource_prefix>_server_
```

The script updates one value at time. Also, for every command, it creates the backup of previous setup under `/opt/scripts/utils/metadata_backup_<timestamp>.txt`.

> **Note:**
>
> After the restart works as desired, you can delete these backed up files.

62. Run the following `update_hostname.sh` script, which is located at `/opt/scripts/cloning`:

```
./update_hostname.sh <FQDN-of-source-instance>
```

For example: `./update_hostname.sh source12c-wls-0.subnet1fd5ed7.idcsvcn.oraclevcn.com`

This updates the hostname to the cloned hostname in the `nodemanager.properties` and `config.xml` files. It also updates the `startup.properties` file to reflect the correct admin url.

63. Reboot the instance.

    This will automatically first start the node manager and then start the administration server.

64. Edit the following files to change the host name to the cloned host name:

    - `/u01/data/cloudgate_config/origin_conf/weblogic.conf`
    - `/u01/data/cloudgate_config/cloudgate.config`

65. Restart the container.

```
sudo systemctl status appgateway.service
sudo systemctl start appgateway.service
sudo podman ps -a
sudo systemctl status appgateway.service
sudo /opt/scripts/idcs/run_cloudgate.sh
```

66. Verify if `nginx` is redirecting traffic to WebLogic server through port `9999`:

```
ip=$(hostname -i)
cloudgate_url=${ip}:9999
curl -v ${cloudgate_url}
curl -s -o /dev/null -w "%{http_code}\n" ${cloudgate_url}
```

67. Create the Backends of the load balanacer that the clone instance uses.

    a. From the navigation menu, click **Networking**, and then click **Load Balancers**.

    b. Click the name of the **Compartment** that contains the load balancer you want to modify, and then click the load balancer's name.

    c. In the **Resources** menu, click **Backend Sets**, and then click the name of the backend set you want to modify.

    d. In the **Resources** menu, click **Backends**.

    e. Click **Backends**.

    f. Select **IP Addresses**.

    g. In the **IP Address** field, enter the IP address of the VMs of the cloned instance

    h. In the Port field, enter `9999`.

    i. Click **Add** > **Close**.

68. After every reboot, manually start the `appgateway`:

```
sudo systemctl start appgateway
sudo podman ps -a
ip=$(hostname -i)
cloudgate_url=${ip}:9999
curl -v ${cloudgate_url}
curl -v ${cloudgate_url}/sample-app
```

69.

70. Select **Enable Access to Administration Console**.

71. Specify the CIDR to create a security list to allow access to the WebLogic administration console port to the source CIDR range.
72.
73. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
74. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
75. Keep the default **Use Public Subnet** selection.
76. If you selected to use an existing subnet, then for subnet type, select **Use Public Subnet** or **Use Private Subnet**. Compute instances in a private subnet are not directly accessible from outside of Oracle Cloud.
77. Select **Add File System Storage**.
78. Configure the file storage:
    - If you chose to create a Virtual Cloud Network (VCN):
      a. *Optional:* Select the availability domain in which you want to create the file system and mount target.
      b. Specify the CIDR of the new subnet.
    - If you chose to use an existing VCN and a new subnet:
      a. *Optional:* Select the availability domain where you want to create the file system and mount target.
      b. Specify the CIDR of the new subnet.
    - If you chose to use an existing VCN and an existing subnet:
      - If you *do not* want to use an existing mount target or an existing file system:
        a. *Optional:* Select the availability domain where you want to create the file system and mount target.
        b. Select an existing subnet to use for the mount target. This subnet must be available in the selected VCN.
      - If you select **Existing Mount Target**:
        a. Select the compartment where you have the existing mount target. The mount target must reside within the subnet in the selected VCN.
        b. Specify the OCID of the existing mount target ID.
      - If you select **Existing File System**:
        a. Select an existing subnet to use for the mount target. This subnet must be available in the selected VCN.
        b. Select the compartment where you have the existing file system.
        c. Specify the OCID of the existing file system.
      - If you select both **Existing Mount Target** and **Existing File System**:
        a. Select the compartment where you have the existing mount target.
        b. Specify the OCID of the existing mount target ID.
        c. Select the compartment where you have the existing file system.

   d. Specify the OCID of the existing file system. The existing file system must be in the same availability domain as the existing mount target.

79. Select **Configure Observability**.
80. Select **Enable exporting logs to OCI Logging Service** to integrate logging for the WebLogic instances.
81. If you use an existing VCN and subnet, validate the network and then enter `YES` in the **Validated Existing Network** field. To validate a network, see Validate Existing Network Setup.
82. Select **Provision with JRF** to configure the domain with JRF components.

    See Configure Database Parameters.
    If you are unsure of selecting this option, see JRF Domain.
83. Keep the default selection for **Deploy Sample Application**, if you want to install a sample web application to the WebLogic cluster, or Identity Cloud Service (IDCS) protected sample application, if you enabled IDCS.

    This option is not displayed if you selected WebLogic Server Standard Edition.
84. Specify the WebLogic startup arguments.

    You can use the server startup arguments to provide arguments to the Java Virtual Machine for WebLogic Server instances. When the servers are scaled out, any changes to the server startup arguments applies to the added nodes only.
    For example, to configure memory settings, you can specify the arguments: *-Xms1024m -Xmx1024m*
85. Select **Configure Ports** to change the default port numbers.

    You can change the default ports for the WebLogic Server Node Manager, WebLogic Server Administration Console, WebLogic Server Administration Console SSL, WebLogic Cluster, WebLogic Managed Server External, and WebLogic Managed Server External SSL.
86. Select the compartment where you have the Application Performance Monitoring domain of your WebLogic instance.
87. Select the Application Performance Monitoring domain of your WebLogic instance.
88. By default, the key automatically generated for your domain `auto_generated_private_datakey` is displayed. But, you can specify any additional private key that you generated for your APM domain.

    See Create an APM Domain in the Oracle Cloud Infrastructure documentation.
89. Connect to the compute instance or the bastion instance as the `opc` user.

    ```
    ssh -i path_to_private_key opc@node_public_ip
    ```

    Or,

    ```
    ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
    path_to_private_key opc@bastion_public_ip" opc@node_private_ip
    ```

For Autoscaling section

1. Select a performance metric for the WebLogic Monitoring Metrics.
2. Specify the threshold values as follows:

   - For *CPU Load* and *Used Heap Percent*, select the minimum and maximum threshold percentage.
   - For *Queue Length* and *Stuck threads*, select the minimum and maximum threshold counter values.

3. Enter a user name to access the image in the registry to deploy autoscaling OCI functions.

   The registry user name format is `tenancy_namespace/<username>`. If your tenancy is federated with Oracle Identity Cloud Service, then the registry user name format is `tenancy_namespace/oracleidentitycloudservice/<username>`.

   You can choose either to include the `tenancy_namespace` or remove the `tenancy_namespace` in the user name format. For example, you can either use `tenancy_namespace/<username>` or `<username>`.

   > **Note:**
   >
   > If you choose to include `tenancy_namespace` in the user format, ensure that you use the correct namespace for your tenancy.

4. Select the compartment where you have OCI secret that contains the registry authentication token, and then select the secret that contains the registry authentication token that you generated for the user to access the image registry.
5. *Optional*: Enter the email ID to receive scaling notifications.

   It is recommended to subscribe to email notifications.

Configure a Resource Manager Private Endpoint

For an existing network and an existing subnet, you can specify the resource manager endpoint to check the provisioning status of the resources on the private subnet.

To use private endpoint, do one of the following:

- Select **Use Existing Resource Manager Endpoint**, and specify the **Resource Manager Private Endpoint**.
- Select **Create New Resource Manager Endpoint**.

Configure a Bastion

You can configure a bastion compute instance on a public subnet to provide access to the WebLogic Server compute instances on a private subnet.

> **Note:**
>
> If you do not select this option, no status is returned for provisioning, then you must check the status of provisioning by connecting to each compute instance and confirm that the `/u01/provStartMarker` file exists with details found in the file `/u01/logs/provisioning.log` file.
>
> It is recommended to deselect the **Provision Bastion Instance** option only in network with fast connect setup.
>
> When you use a public subnet for an existing VCN and existing subnet, bastion is not required. However, when you create a new VCN, or create a new subnet for an existing VCN, you must create a bastion during stack creation.

To configure a bastion:

1. If your want to use a bastion compute instance with a reserved public IP, then select **Assign Reserved Public IP to Bastion Instance**.
2. For the bastion host subnet, specify one of the following:
   - If you want to use an existing regional subnet, then choose the name of an existing regional subnet from the list of regional and availability domain-specific subnets.
   - If you are creating a new regional subnet, specify a CIDR for the new subnet.

   > **Note:**
   >
   > This is applicable if you are using an existing VCN with new subnets or creating a new VCN.
3. Select a shape for the bastion compute instance.
4. If you are using existing network security groups (NSGs) for an existing subnet, specify the NSG that is assigned to the bastion instance.

Configure a Load Balancer

You have the option to create a load balancer to distribute application traffic to the WebLogic Managed Servers. You can also use an existing load balancer for an existing VCN and an existing subnet, to distribute application traffic to the WebLogic Managed Servers.

> **Note:**
>
> If you enable autoscaling for WebLogic instances, you must configure a load balancer in Oracle Cloud Infrastructure when you create a stack, else the stack provisioning fails with a validation error.

- Configure New Load Balancer
- Configure Existing Load Balancer

> **Note:**
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies.

Configure New Load Balancer

Specify the parameters to configure a new load balancer.

1. Select **Create New Load Balancer**.
2. Select **Private Load Balancer**, if you do not want to assign a public IP address to the load balancer.
   This option is available only if you use a private subnet for WebLogic Server. You cannot create a private load balancer in a public subnet.
3. Select **Load Balancer with Reserved Public IP**, if you want to use a public load balancer with a reserved public IP. Then, specify the OCID of the public IP for the load balancer.
4. Select a minimum and maximum flexible load balancer shape. By default, the minimum bandwidth size is set to 10Mbps and maximum to 400Mbps.

   > **Note:**
   >
   > You can update the shape to a maximum of 8000Mbps. Before you select the maximum bandwidth, ensure to check the available service limit for the flexible load balancer bandwidth.

5. Configure the load balancer network.
   - If you want to use an existing regional subnet for WebLogic Server, then select an existing regional subnet from the list of regional and availability domain-specific subnets. A load balancer can have only one regional subnet, which is shared between both nodes.

- If you are creating a regional subnet for WebLogic Server, then specify a CIDR for the new load balancer subnet.

6. If you are using existing network security groups (NSGs) for an existing subnet, specify the NSG that is assigned to the load balancer.

Configure an Existing Load Balancer

Specify the parameters to configure an existing load balancer.

You can use an existing load balancer only if you are using an existing VCN and existing subnet. If you use an existing load balancer, you must create a backend set, listener, and routing policy. See Configure the Load Balancer.

> **Note:**
>
> The existing load balancer must be in the network compartment, which can be same as the stack compartment, and must be in the same VCN as the existing VCN chosen for the stack.

1. Select **Use Existing Load Balancer**.
2. Specify the OCID for the existing load balancer.
3. Enter the name of the backend set for the existing load balancer that has a routing policy associated with the backend set. The backend set should not have backends. See Configure the Load Balancer.
4. If you are using existing network security groups (NSGs), specify the NSG that is assigned to the load balancer.

Configure WebLogic Authentication

You have the option to use Oracle Identity Cloud Service to authenticate application users for your domain.

This configuration is only available if the domain includes a load balancer.

> **Note:**
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies.

To use Oracle Identity Cloud Service for authentication:

1. Enter your Oracle Identity Cloud Service (IDCS) tenant name, which is also referred to as the instance ID.

   This ID is typically found in the URL that you use to access Oracle Identity Cloud Service, and has the format `idcs-<GUID>`.

2. Enter the client ID of an existing confidential application in this Oracle Identity Cloud Service instance.

3. Select the compartment where you have the IDCS secret and then select the secret that contains the client secret of the existing confidential application. To create secrets, see Create Secrets for Passwords.
4. If necessary, you can override the default domain name and port that you use to access Oracle Identity Cloud Service, or the default port that is used for the App Gateway software appliance.

Configure Application Performance Monitoring

You can configure Application Performance Monitoring (APM) to gain visibility into the performance of applications.

This configuration is required for metric-based autoscaling of instances.

> **Note:**
>
> If you enable autoscaling, you cannot use the always free Application Performance Monitoring domain.

1. Select the compartment where you have the Application Performance Monitoring domain of your WebLogic instance.
2. Select the Application Performance Monitoring domain of your WebLogic instance.
3. By default, the key automatically generated for your domain `auto_generated_private_datakey` is displayed. But, you can specify any additional private key that you generated for your APM domain.

   See Create an APM Domain in the Oracle Cloud Infrastructure documentation.

Configure Autoscaling

You can enable autoscaling for WebLogic instances based on WebLogic Monitoring Metrics.

You must configure Application Performance Monitoring before you configure autoscaling. However, Application Performance Monitoring always free domain is not supported for autoscaling.
See Configure Application Performance Monitoring.

> **Note:**
>
> You cannot enable autoscaling after you have created the Oracle WebLogic Server for OCI domain.
> For autoscaling, ensure that you configure either public load balancer, private load balancer, or load balancer with reserved IP.
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies for Autoscaling.

To use autoscaling for your WebLogic instances:

1. Select a performance metric for the WebLogic Monitoring Metrics.

2. Specify the threshold values as follows:
   - For *CPU Load* and *Used Heap Percent*, select the minimum and maximum threshold percentage.
   - For *Queue Length* and *Stuck threads*, select the minimum and maximum threshold counter values.
3. Enter a user name to access the image in the registry to deploy autoscaling OCI functions.

   The registry user name format is `tenancy_namespace/<username>`. If your tenancy is federated with Oracle Identity Cloud Service, then the registry user name format is `tenancy_namespace/oracleidentitycloudservice/<username>`.

   You can choose either to include the `tenancy_namespace` or remove the `tenancy_namespace` in the user name format. For example, you can either use `tenancy_namespace/<username>` or `<username>`.

   > **Note:**
   >
   > If you choose to include `tenancy_namespace` in the user format, ensure that you use the correct namespace for your tenancy.

4. Select the compartment where you have OCI secret that contains the registry authentication token, and then select the secret that contains the registry authentication token that you generated for the user to access the image registry.
5. *Optional*: Enter the email ID to receive scaling notifications.

   It is recommended to subscribe to email notifications.

Configure File System

Specify the parameters to configure file storage.

If you are not an administrator, the necessary groups and policies must be in place before you can create a domain.

You can configure a file storage as follows:

> **Note:**
>
> You can configure an existing file system, configure an existing mount target, and create a file system and mount target, if you only chose to use an existing VCN with new subnets, or existing VCN and existing subnets.

- Configure Existing File System
- Configure Existing Mount Target
- Configure File System and Mount Target
- Configure File Storage for a New VCN

> **Note:**
>
> Ensure that you have set up the user group polices and dynamic group polices that are applicable for the configuration. See User Group Policies and Dynamic Group Policies.

Configure Existing File System

Specify the parameters to configure an existing file system.

1. Select **Existing File System**.
2. Select the name of the availability domain for your file system and mount target.
3. Select the compartment where you have the existing file system.
4. Specify the OCID of the existing file system.
5. Select the compartment where you have the existing mount target.
6. Select the existing mount target. The mount target must reside within the subnet in the selected VCN.

Configure Existing Mount Target

Specify the parameters to configure an existing mount target.

1. Select the name of the availability domain for your file system and mount target.
2. Select **Existing Mount Target**.
3. Select the compartment where you have the existing mount target.
4. Select the existing mount target. The mount target must reside within the subnet in the selected VCN.

Configure File System and Mount Target

Specify the parameters to configure file system and mount target, if you do not want to use an existing mount target or an existing file system.

1. Select the name of the availability domain for your file system and mount target.

   > **Note:**
   >
   > You can select the availability domain for regional subnets only.

2. For the subnets, specify one of the following:
   - If you want to use an existing regional subnet, then choose the name of an existing regional subnet from the list of regional and availability domain-specific subnets.
   - If you are creating a new regional subnet, specify a CIDR for the new subnet.
3. If you are using existing network security groups (NSGs) for an existing subnet, specify the NSG that is assigned to the mount target.

> **Note:**
>
> You can specify NSGs only for an existing VCN and existing subnet.

Configure File System for a New VCN

Specify the parameters to configure file storage if you chose to create a new VCN.

1. Select the availability domain in which you want to create the file system and mount target.
2. Specify a CIDR for the new mount target subnet.

Configure Tags

Oracle WebLogic Server for OCI can optionally assign tags to the resources (compute, network, and so on) that it creates for your domain.

Tagging allows you to define keys and values and associate them with resources. You can then use the tags to help you organize and find resources based on your business needs. There are separate fields to tag the stack and to tag the resources created within the stack.

1. Select **Create Tags**.
2. To assign a free-form tag, enter the **Tag Key** and **Value**.

   Free-form tag keys and values are case sensitive. For example, `costcenter` and `CostCenter` are treated as different tags.
3. To assign an existing tag, for **Tag Namespace**, select a defined tag, and then select the **Tag Key** and **Value**. If no value is displayed for the **Tag Key**, enter the **Value**.
4. Click **Additional Tag** to assign additional free-form or defined tags.

Create the Domain Stack

After you have specified the WebLogic instance variables, finish creating the domain stack.

On the Review page of the Create Stack wizard, review the information you have provided, and then click **Create**.

The Job Details page of the stack in Resource Manager is displayed. A stack creation job name has the format `ormjob`*`yyyymmddnnnnnn`*. (for example, `ormjob20190919165004`). Periodically monitor the progress of the job until it is finished. If an email address is associated with your user profile, you will receive an email notification. In the **Application Information** tab, you can directly access the OCI resources using the WebLogic instance IP and the bastion instance IP.

> **Note:**
>
> If there is an error during the creation of the stack, the compute, network, and other resources in the stack are not automatically deleted. If you want to delete the failed stack, see Delete a Stack.

Use Your New Domain

Access and manage your new domain after creating a stack with Oracle WebLogic Server for OCI.

Typical tasks that you might perform after creating a domain:

- View and manage the cloud resources that were created to support your domain. See View the Cloud Resources for a Stack.
- Use the WebLogic Server administration console to configure your domain. Create data sources, JMS modules, Coherence clusters, and so on, or deploy applications. See Access the WebLogic Server Administration Console.

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Oracle WebLogic Remote Console.

- Access the sample application that's deployed to your domain. See Access the Sample Application.
- Secure access to your applications using Oracle Identity Cloud Service. See Secure a Domain Using Identity Cloud Service.
- Add your own SSL certificate to the load balancer. See Add a Certificate to the Load Balancer.
- Troubleshoot a problem with your new stack. See Stack Creation Failed.

You can also use the Fusion Middleware Control Console to monitor, configure, and manage a JRF-enabled domain. See Access the Fusion Middleware Control Console.

View the Cloud Resources for a Stack

Use Resource Manager to view the Oracle Cloud Infrastructure compute instances, networks, and other resources that were provisioned by Oracle WebLogic Server for OCI for your Oracle WebLogic Server stack.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. From the Jobs section, click the latest job whose type is Apply.
6. Click **Associated Resources**.

   A list of resources in this stack displays. The list might include compute instances, virtual cloud networks (VCN), subnets, gateways, block storage volumes, and network security groups or security lists.
7. Click the name of a resource to manage it and to view its details.

For example, click a compute instance to view its IP address or to reboot it.

Alternatively, you can find your domain's resources by using the search field at the top of the console. For example, if you assigned tags to the resources in the stack, you can enter these tags in the search field.

3 Manage a Domain

Learn how to access the administration console for an Oracle WebLogic Server for OCI domain, access the sample application, secure the domain, and delete the domain when you no longer need it.

Topics:

- About Managing a Domain
- About the Default Ports
- About the Security Checkup Tool
- Access the Compute Instance
- Access the WebLogic Server Administration Console
- Access the WebLogic Remote Console
- Access the Fusion Middleware Control Console
- Access the Sample Application
- Start and Stop a Domain
- Add IDCS After Creating a Domain
- Back Up and Restore a Domain
- Add a Certificate to the Load Balancer
- Secure a Domain Using Identity Cloud Service
- Secure Web Services Using Identity Cloud Service
- Integrate OPSS User and Group APIs with Identity Cloud Service
- Upgrade the Oracle Identity Cloud Service App Gateway Version
- Configure Session Persistence
- Update the Password Secret OCID, Policy, and User Name
- Update the Infrastructure Schema Password
- Manage Data Sources
- Connect to a Domain Using Oracle JDeveloper
- Upgrade a Domain

About Managing a Domain

Learn about managing an Oracle WebLogic Server domain after creating it with Oracle WebLogic Server for OCI.

In general, you configure, manage, and maintain an Oracle WebLogic Server for OCI domain just like an on-premise domain. For example, to deploy an application:

- Roadmap for Deploying Applications in WebLogic Server (14.1.2.0)

- Roadmap for Deploying Applications in WebLogic Server (14.1.1.0)
- Roadmap for Deploying Applications in WebLogic Server (12.2.1.4)

This chapter provides additional help, best practices, and tools for some WebLogic Server tasks:

- Access the WebLogic Server Administration Console
- Access the Fusion Middleware Control Console
- Access the Sample Application
- Start and Stop a Domain
- Scale a Stack
- Add IDCS After Creating a Domain
- Back Up and Restore a Domain
- Add a Certificate to the Load Balancer
- Secure a Domain Using Identity Cloud Service
- Secure Web Services Using Identity Cloud Service
- Integrate OPSS User and Group APIs with Identity Cloud Service
- Configure Session Persistence
- Manage Data Sources
- Connect to a Domain Using Oracle JDeveloper
- Delete a Stack

About the Default Ports

Oracle WebLogic Server for OCI configures listen ports and network channels in a new domain.

The default ports will vary depending on the domain type selected: Production Mode (default) or Secured Production Mode.

Production Mode (default)

Each server has both internal and external ports. Internal ports are not accessible from outside of Oracle Cloud. External ports support HTTP and HTTPS only. They do not support the T3 and T3S protocols, and they do not support HTTP tunneling. Oracle does not recommend enabling the T3 protocol or HTTP tunneling on network channels that are accessible from outside of Oracle Cloud.

If you chose to use a public subnet for the domain's compute instances, then the instances are assigned public IP addresses, and the external server ports are accessible from the Internet. If you used a private subnet, the external server ports are accessible only from the Oracle Cloud network, or from your on-premises data center over a VPN network.

Administration Server

Channel	Port, Protocol	Purpose
Default listen port	9071, T3	• Internal domain communication • Use administration clients like the WebLogic Scripting Tool (WLST) within this virtual cloud network (VCN) • Access T3 applications within this VCN
Default SSL listen port	9072, T3S	• Internal domain communication • Use administration tools within this VCN • Access T3S applications within this VCN
`ExternAdmin`	7001, HTTP	• Access the administration console • Access web applications
`SecuredExternAdmin`	7002, HTTPS	• Access the administration console • Access web applications

Managed Servers

Channel	Port, Protocol	Purpose
Default listen port	9073, T3	• Internal domain communication • Use administration tools within this VCN • Access T3 applications within this VCN
Default SSL listen port	9074, T3S	• Internal domain communication • Use administration tools within this VCN • Access T3S applications within this VCN
`ExternAdmin`	7003, HTTP	Access web applications
`SecuredExternAdmin`	7004, HTTPS	Access web applications

Secured Production Mode
When the Secured Production Mode Enabled option is selected Oracle WebLogic Server for OCI configures TLS (SSL) listen ports, network channels, and administration ports in a new domain.

Each server has both internal and external ports. Internal ports are not accessible from outside of Oracle Cloud. External ports support HTTPS only. They do not support the T3 and T3S protocols, and they do not support HTTP tunneling. Oracle does not recommend enabling the T3 protocol or HTTP tunneling on network channels that are accessible from outside of Oracle Cloud.

If you chose to use a public subnet for the domain's compute instances, then the instances are assigned public IP addresses, and the external server ports are accessible from the Internet. If you used a private subnet, the external server ports are accessible only from the Oracle Cloud network, or from your on-premises data center over a VPN network.

Administration Server

Channel	Port, Protocol	Purpose
Administration port	9002	• Access the administration console • Internal domain communication • Use administration clients like the WebLogic Scripting Tool (WLST) within this virtual cloud network (VCN)
Default TLS listen port	9072, T3S	Access T3S applications within this VCN
`SecuredExternAdmin`	7002, HTTPS	Access web applications

Managed Servers

Channel	Port, Protocol	Purpose
Administration port	9004	• Internal domain communication • Use administration clients like the WebLogic Scripting Tool (WLST) within this virtual cloud network (VCN)
Default TLS listen port	9074, T3S	Access T3S applications within this VCN
`SecuredExternAdmin`	7004, HTTPS	Access web applications

About the Security Checkup Tool

Oracle WebLogic Server Administration console includes a security checkup tool that displays security check warnings. These security check warnings are displayed for Oracle WebLogic Server for OCI instances that are created using WebLogic Server versions 12.2.1.4 and 14.1.1.0.

> **Note:**
>
> When the Secured Production Mode Enabled option is selected, the security checkup tool will report no issues. This section should be skipped.

In case of Oracle WebLogic Server for OCI instances created after July 20, 2021, or the instances on which the July 2021 PSUs are applied, the message `Security warnings detected. Click here to view the report and recommended remedies` is displayed at the top of the Oracle WebLogic Server Administration console. When you click the message, a list of security warnings are displayed as listed in the following table.

The warning messages listed in the table are examples.

Security Warnings

Warning Message	Resolution
`The configuration for key stores for this server are set to Demo Identity and Demo Trust. Trust Demo certificates are not supported in production mode domains.`	Configure the identity and trust keystores for each server and the name of the certificate in the identity keystore that the server uses for SSL communication. See Configure Keystore Attributes for Identity and Trust. **Note:** This warning is displayed for Oracle WebLogic Server for OCI instances created after October 20, 2021, or the instances on which the October PSUs are applied.
`Remote Anonymous RMI T3 or IIOP requests are enabled. Set the RemoteAnonymousRMIT3Enabled and RemoteAnonymousRMIIIOPEnabled attributes to false.`	Disable the anonymous RMI T3 and IIOP requests in the WebLogic Server Administration Console as soon as possible unless your deployment requires anonymous T3 or IIOP (not typical). See Disable Remote Anonymous RMI T3 and IIOP Requests.

> **Note:**
>
> For existing Oracle WebLogic Server for OCI instances created before release 21.3.2 (August 17, 2021), you see the SSL host name verification warnings. See Security Checkup Tool Warnings.

After you address the warnings, you must click **Refresh Warnings** to see the warnings removed in the console.

For Oracle WebLogic Server for OCI instances created after July 20, 2021, though the java properties to disable anonymous requests for preventing anonymous RMI access are configured, the warnings still appear. This is a known issue in Oracle WebLogic Server.

If you want to perform anonymous RMI requests, you must disable the java properties. Go to the `nodemanager.properties` file located under `DOMAIN_HOME/nodemanager` and remove the `weblogic.startup.Arguments` property.

Disable Remote Anonymous RMI T3 and IIOP Requests

To disable the remote anonymous RMI T3 and IIOP requests in the WebLogic Server Administration console:

1. Locate the **Change Center** and click **Lock & Edit** to lock the editable configuration hierarchy for the domain.
2. Under **Domain structure**, select the domain name, and then select the **Security** tab.
3. Expand **Advanced** and deselect **Remote anonymous RMI access via IIOP** and **Remote anonymous RMI access via T3**.

After saving the changes, return to **Change Center** and click **Activate Changes**.

Configure Keystore Attributes for Identity and Trust

To configure the identity and trust keystore files and the name of the certificate in the identity keystore in the WebLogic Server Administration console:

1. Locate the **Change Center** and click **Lock & Edit** to lock the editable configuration hierarchy for the domain.
2. Under **Domain structure**, select **Environment** and then select **Servers**.
3. In the Servers table, select the server you want to configure.
4. On the **Configuration** tab, click **Keystores**, and then click **Change**.
5. Select *Custom Identity and Custom Trust*, and then click **Save**.
6. Under **Identity**, provide the following details:
   a. Enter the full path of your identity keystore.

      For example: `/u01/data/keystores/identity.jks`
   b. For **Custom Identity Keystore Type**, enter *JKS*.
   c. For **Custom Identity Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Identity Keystore Passphrase**.
7. Under **Trust**, provide the following details:
   a. Enter the full path of your identity keystore.

      For example, `/u01/data/keystores/trust.jks`
   b. For **Custom Trust Keystore Type**, enter *JKS*.
   c. For **Custom Trust Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Trust Keystore Passphrase**.
8. Click **Save**.
9. Click the **SSL** tab.
10. Under **Identity**, provide the following details:
    a. For **Private Key Alias**, enter the name of the certificate (private key) in the identitykeystore, *server_cert*.
    b. For **Private Key Passphrase**, enter the password for this certificate in the keystore. Enter the same value for **Confirm Private Key Passphrase**.

       By default, the password for the certificate is the same as the identity keystore password.
11. Click **Save**.

    After saving the changes, return to **Change Center** and click **Activate Changes**.
12. Repeat steps 3 to 9 to configure each server in the domain.

Access the Compute Instance

Use SSH and set up the proxy to access the compute instance.

1. Identify the IP address of the node in your domain.

The name of the node is *servicename*-wls-*n*, where servicename is the resource name prefix you provided during stack creation. The Administration Server runs on the first node, *servicename*-wls-0

- If your domain is on a public subnet, then use the public IP address of the compute instance.
- If your domain is on a private subnet, then use the public IP address of the bastion and the private IP address of the compute instance.

2. Open an SSH connection to the node as the opc user.

```
ssh -i <path_to_private_key> opc@<node_public_ip>
```

Or,

```
ssh -i <path_to_private_key> -o ProxyCommand="ssh -W %h:%p -i
<path_to_private_key> opc@<bastion_public_ip>" opc@<node_private_ip>
```

3. Change to the oracle user.

```
sudo su - oracle
```

Access the WebLogic Server Administration Console

Use the WebLogic Server Administration Console to access a domain in Oracle WebLogic Server for OCI releases 12.2.1.4 and 14.1.1.0.

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Access the WebLogic Remote Console.

- Access the WebLogic Console in a Public Subnet
- Access the WebLogic Console in a Private Subnet
- Access the WebLogic Console Through a Load Balancer

> **Note:**
>
> Security check warnings are displayed at the top of the console. See About the Security Checkup Tool for the warnings and how to handle them.

Access the WebLogic Console in a Public Subnet

Oracle WebLogic Server compute instances assigned to a public subnet are accessible from the public Internet.

1. Sign in to the Oracle Cloud Infrastructure Console.

2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Copy the public IP address value.
6. In a browser, specify the URL of the WebLogic Server Administration Console in the following format, using the public IP address and appropriate port:

   ```
   https://IP-address:port/console
   ```

   The default SSL port is 7002, unless it was changed during stack creation. For domains created in secured production mode in 12.2.1.4 or14.1.2.0 then the SSL administration port is 9002 as shown in the following examples:

   Production Mode:

   ```
   https://192.0.2.1:7002/console
   ```

   Secured Production Mode (default for 12.2.1.4 or 14.1.2.0):

   ```
   https://192.0.2.1:9002/console
   ```
7. When prompted, enter the WebLogic Server administrator user name and password for this domain.

Access the WebLogic Console in a Private Subnet

Oracle WebLogic Server compute instances assigned to a private subnet are not accessible from the public Internet.

To access the WebLogic Server Administration Console to administer such instances, you can use:

- Bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility. See Access by Using the Bastion Instance.
- Bastion service and dynamic port forwarding with a secure shell (SSH) utility. See Access by Using the Bastion Service.

By opening an SSH tunnel with dynamic port forwarding, the SSH client becomes a Socket Secure (SOCKS) proxy listening on the port you specify. All traffic that routes to the proxy port is forwarded to its destination through the proxy server. Then when you configure your browser to use a SOCKS proxy, you can access a private domain's administration console.

Access by Using the Bastion Instance

To access the WebLogic console in a private subnet by using the bastion instance, complete the following steps:

1. Sign in to the Oracle Cloud Infrastructure Console.

2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** drop-down list, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Copy the private IP address value.
6. Return to the Compute Instances page.
7. Click the name of the bastion instance that's associated with the domain.

   The domain's bastion instance is identified by `*servicename*-bastion-instance`. For example: `abcde7xy-bastion-instance`
8. Copy the public IP address value.
9. From your computer, open an SSH tunnel. There are two options:

   a. To open an SSH tunnel by running the `ssh` command, follow these instructions:

      i. From your computer, open an SSH tunnel to an unused port on the bastion node as the `opc` user. For example, you can use port `1088` for SOCKS proxy. It can be any other unused port.
      Specify the `-D` option to use dynamic port forwarding. Provide the path to the private key that corresponds to the public key that you specified when you created the domain.

      The SSH command format is:

      ```
      ssh -C -D port_for_socks_proxy -i path_to_private_key 
      opc@bastion_public_ip
      ```

      For example:

      ```
      ssh -C -D 1088 -i ~/.ssh/mykey.openssh opc@198.51.100.1
      ```

      ii. In your browser settings, set up the SOCKS (version 5) proxy configuration. Specify your local computer and the same SOCKS port that you used in your SSH command.

      iii. Specify the URL of the WebLogic Server Administration Console in the following format, using the private IP address:

      ```
      http://private_ip_address:port/console
      ```

      The default SSL port is 7002, unless it was changed during stack creation. If secured production mode was enabled then the SSL administration port is 9002 as shown in the following examples:

      Production Mode (default):

      ```
      https://192.0.2.1:7002/console
      ```

Secured Production Mode:

```
https://192.0.2.1:9002/console
```

iv. When prompted, enter the WebLogic Server administrator user name and password.

On a Windows platform, you can use Windows PowerShell to run the SSH command or use PuTTY.

b. To open an SSH Tunnel using PuTTY, follow these instructions:

i. Start PuTTY on your Windows computer.
The PuTTY Configuration page is displayed, showing the **Session** panel.

ii. In the **Host Name (or IP address**) field, enter the public IP address of the bastion node.

iii. In the Category navigation tree, expand **Connection**, and then click **Data**.

iv. In the **Auto-login username** field, enter `opc`.

v. In the **When username is not specified** field, select **Prompt**.

vi. In the Category tree, expand **Connection**, and then click **SSH**.

vii. Under **Protocol options**, select the **Don't start a shell command at all** check box.

viii. In the Category tree, expand **SSH**, and then click **Auth**.

ix. Under **Private key file for authentication**, click **Browse**.

x. Navigate to the location of your private key file, and select it. Click **Open.**This private key corresponds to the public key that you specified when you created this service instance.

> **Note:**
>
> The `.ppk` file extension indicates that the private key is in PuTTY's proprietary format. You must use a key of this format when using PuTTY. If Oracle Cloud generated this key for your service instance, see the PuTTY documentation for information about converting the key format

xi. In the Category tree, expand **SSH**, and then click **Tunnels**.

xii. In the **Destination** field, enter `IP:port` where IP is the private IP address of the WebLogic Server node and port is the port number on the node to which you want to connect.

xiii. In the **Source Port** field, enter the same port number.

xiv. Click the **Add** button.

xv. **Optional:** To save this session configuration, click **Session** in the Category tree, and then click **Save**.
To load a saved configuration, select the configuration name, and then click **Load**.

xvi. Click **Open**.

xvii.If prompted, enter the passphrase for the private key.

Applications that are running on your local computer can now communicate with the node by using `localhost:`*`port`*, where *port* is the local port number.

For example:

```
http://localhost:7001/console
```

The default port is 7001, unless it was changed during stack creation. When prompted, enter the WebLogic Server administrator user name and password.

After your work with the SSH tunnel is completed, press Ctrl+C to close the SSH tunnel.

Access by Using the Bastion Service

To access the WebLogic console in a private subnet by using the bastion service, complete the following steps:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. If you have already created a Bastion service and a session in the required VCN, navigate to the Bastion and the Session, and then continue from step 18.
3. Click the navigation menu ☰, select **Identity & Security**. Under the **Identity & Security** group, click **Bastion**.
4. From the **Compartment** drop-down list, select the compartment in which your domain is created.
5. Click **Create bastion**.
6. Enter a name for the bastion.
7. Under **Configure networking**, select the **Target virtual cloud network** of the target resource that you intend to connect to by using sessions hosted on this bastion.
8. Select the **Target subnet**.

   The subnet must either be the same as the target resource's subnet or it must be a subnet from which the target resource's subnet accepts network traffic.
9. In **CIDR block allowlist**, add one or more address ranges in CIDR notation that you want to allow to connect to sessions hosted by this bastion.

   Enter a CIDR block, and then either click the value or press **Enter** to add the value to the list. The maximum allowed number of CIDR blocks is 20.
10. From the list of Bastion, click the name of the Bastion you created.
11. Click **Create session**.
12. From the **Session type** drop-down list, select **SSH port forwarding session**.
13. Under **Connect to the target host** option, select **Instance name**.
14. From the Compute instance, select the domain instance that has the Administration Server node.

    The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
15. In the **Port** field, enter the WebLogic administration server's listener port number, where the administration console is accessible. By default, the port number `7002`.
16. Under **Add SSH Key**, provide the public key file of the SSH key pair that you want to use for the session.
17. Click **Create Session**.

18. Click the **Actions** icon for the session you created, and select **View SSH command**.
19. In the **View SSH command** window, click **Copy** to copy the SSH command.
20. Paste the SSH command in a text editor.
21. In the SSH command, replace <*privateKey*> with the path to the private key that corresponds to the public key that you specified when you created the domain.
22. In the SSH command, replace the <*localPort*> with the preferred local port number.
23. Copy the updated SSH command.
24. From your computer, open a SSH utility.
25. Paste the SSH command and then press **Enter**.

    If prompted to continue connecting, type `Yes` and press **Enter**.
26. Specify the URL of the WebLogic Server Administration Console in the following format:

    ```
    https://localhost:<local-port>/console
    ```

    The default port is `7002`, unless it was changed during stack creation.

    For example:

    ```
    https://localhost:7002/console
    ```
27. When prompted, enter the WebLogic Server administrator user name and password.

Access the WebLogic Console Through a Load Balancer

Oracle WebLogic Server compute instances assigned to a private subnet are not accessible from the public Internet. So, if you have created the load balancer manually, you can access the administration console via load balancer's public IP address.

However, it is recommended to access the WebLogic Server Administration Console using the bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility. See Access the WebLogic Console in a Private Subnet.

> **Note:**
>
> If you created the load balancer in a different network than the Oracle WebLogic Server domain network with nonoverlapping CIDRs, you must create a Local Peering Gateway on both the domain network and load balancer network and add a route table for the subnet for WebLogic domain and the load balancer subnet to use the respective Local Peering Gateway.

For SSL connections that terminate at the load balancer, you must configure the load balancer to include the `WL-Proxy-SSL` header in the rule set and enable the WebLogic Proxy Plugin on the cluster and the administration server.

To access the WebLogic administration console through the load balancer:

1. Configure the load balancer to include the `WL-Proxy-SSL` header in the rule set. See Create an HTTPS Listener for the Load Balancer.

2. Enable WebLogic Proxy Plugin for the administration server by adding the following line in domain `config.xml` under `AdminServer` config:

   ```
   <weblogic-plugin-enabled>true</weblogic-plugin-enabled>
   ```

3. Restart the domain using the following script:

   ```
   /opt/scripts/restart_domain.sh -o restart
   ```

4. Create a backend set for the load balancer.
   a. Sign in to the Oracle Cloud Infrastructure Console.
   b. From the navigation menu, click **Networking**, and then click **Load Balancers**.
   c. Click the name of the Compartment that contains the load balancer you want to modify, and then click the load balancer's name.
   d. Under **Resources** menu, click **Backend Sets**,

      The list of Backend Sets is displayed.
   e. Click **Create Backend Set**.
   f. Specify the backend set details.
      i. Enter a **Name** for the backend set.
      ii. Select the load balancer policy for the backend set.
      For policies, see Load Balancing Policies.
      iii. Under **Health Check**, specify the parameters to confirm the health of backend servers.
         - **Protocol**: *HTTPS*
         - **Port**: *7001* (default) or *9002* (secured production mode)
         - **URL Path**: Private IP address of the WebLogic Server admin VM

         Optionally, you can specify the other test parameters to confirm the health of backend servers, as required.
   g. Click **Create Backend Set**.
5. Specify the backend server for the backend set.
   a. Click the name of the backend set that you created in step 4.
   b. Under **Resources** menu, click **Backends**.
   c. Click **Add Backends**.
   d. Select **IP Addresses**.
   e. Enter the private IP address of a backend server you want to add to the backend set.
   f. For **Port**, enter *7001* (default) or *9002* (secured production mode).
   g. Click **Add**.
6. In the Oracle Cloud Infrastructure Console, from the navigation menu, click **Networking**, and then select **Load Balancers**.
7. Select the Compartment where your load balancer is located.
8. Click the name of the load balancer that is associated with your domain.
9. Copy the public IP address value.

10. In a browser, specify the URL of the WebLogic Server Administration Console in the following format, using the public IP address and port:

    The default SSL port is 7002, unless it was changed during stack creation. If secured production mode was enabled then the SSL administration port is 9002 as shown in the following examples:

    Production Mode (default):

    ```
    https://192.0.2.1:7002/console
    ```

    Secured Production Mode:

    ```
    https://192.0.2.1:9002/console
    ```

11. When prompted, enter the WebLogic Server administrator user name and password.

Access the WebLogic Remote Console

Use the WebLogic Server Remote Console to access a domain in Oracle WebLogic Server for OCI release 14c (14.1.2.0.0).

> ✎ **Note:**
>
> As of 14c (14.1.2.0.0) the WebLogic Server Administration Console has been removed. For comparable functionality, you will use the WebLogic Remote Console.
>
> If you are running 12c (12.2.1.4.0) or 14c (14.1.1), then you will continue to use the WebLogic Console as described in Access the WebLogic Server Administration Console.

In 14*c* (14.1.2.0.0) you will use the new Oracle WebLogic Remote console to manage domains installed with WebLogic Server for OCI Stacks. The following sections describe how to perform some basic management functions. For more information about using the Remote Console, see the Remote Console online help.

After you have created a domain using WebLogic Server for OCI stack version 14.1.2, you must download and install the Remote Console from the following GitHub repository:https://github.com/oracle/weblogic-remote-console .

Access the WebLogic Remote Console in a Public Subnet

Oracle WebLogic Servers running on compute instances assigned to a public subnet are accessible from the public internet using the Remote Console.

1. Sign into the Oracle Cloud Infrastructure Console.
2. Click the Navigation Menu icon, select **Compute**. Under the Compute group, click **Instances**.
3. From the Compartment dropdown, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

5. The instance with the Administration Server node has wls-0 appended to the name. For example: `abcde7xy-wls-0`
6. Copy the public IP address value.
7. Open the WebLogic Remote Console and create a new Admin Server Connection Provider.
8. In the provider information provide the following values:
   a. **Name**: Use a name that can easily correlate to this WebLogic domain e.g. abcde7xy
   b. **Provide Credentials**: There are two methods to provide credentials in WebLogic 14.1.2:
      - specify the Username and Password fields in the provider
      - ask the remote console to open a web browser to authenticate using Single Sign on which will require you to re-authenticate when your session is expired
   c. **URL**: https://*IP-address:wls_administration_port*
      The default WebLogic Server domain-wide Administration Port is 9002

      For example:

      ```
      https://192.0.2.1:9002
      ```
   d. **Proxy Override:** Leave this field empty.
   e. **Make Insecure Connection**: Selected
      The certificates used in WebLogic Server for OCI are issued to OCI DNS which is not resolvable outside. If you want to verify for Domain name information you can create a local host alias to pointing the Admin Server Internal FQDN to the Admin Server IP address and use the FQDN instead of the IP in URL.

      > **Note:**
      >
      > You may also need to trust the CA used to sign your certificate.

Access the WebLogic Remote Console in a Private Subnet

Oracle WebLogic Server compute instances assigned to a private subnet are not accessible from the public Internet.Therefore, in order to use WebLogic Remote console to manage WebLogic domains deployed in such instances, you must create SSH tunneling.

1. Create SSH tunneling.
   By opening an SSH tunnel with dynamic port forwarding, the SSH client becomes a Socket Secure (SOCKS) proxy listening on the port you specify. All traffic that routes to the proxy port is forwarded to its destination through the proxy server. Then when you configure your local installation of the WebLogic Remote Console to use a SOCKS proxy.

   You can open an SSH tunnel using any of the following options:
   - Bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility. See Create SSH Tunneling Using the Bastion Instance.
   - Bastion service and dynamic port forwarding with a secure shell (SSH) utility. See Create SSH Tunneling Using the Bastion Service.
2. Configure a new provider in WebLogic Remote Console. See Configure Admin Server Provider using Proxy

Create SSH Tunneling Using the Bastion Instance

To use the WebLogic Remote console to manage WebLogic domains in a private subnet, you must use the bastion instance.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the Navigation Menu icon, select **Compute**. Under the Compute group, click **Instances**.
3. From the Compartment drop-down list, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.
   The instance with the Administration Server node has wls-0 appended to the name. For example: `abcde7xy-wls-0`
5. Copy the private IP address value.
6. Return to the Compute Instances page.
7. Click the name of the bastion instance that's associated with the domain.
   The domain's bastion instance is identified by servicename-bastion-instance. For example: `abcde7xy-bastion-instance`
8. Copy the public IP address value.
9. From your computer, open an SSH tunnel. There are two options:
   - running the ssh command
   - using PuTTY

   To open an SSH tunnel by running the `ssh` command, follow these instructions:

   a. From your computer, open an SSH tunnel to an unused port on the bastion node as the opc user.
      For example, you can use port 1088 for SOCKS proxy, but it can be any other unused port.

   b. Specify the `-D` option to use dynamic port forwarding.
      Provide the path to the private key that corresponds to the public key that you specified when you created the domain.

      The SSH command format is:

      ```
      ssh -i path_to_private_key -N -L localPort:AdminServerIP:wls_admin_port
      -p ssh_port opc@bastion_public_ip
      ```

      The default WebLogic Server domain-wide Administration Port is 9002

      For example:

      ```
      ssh -i ~/.ssh/mykey.openssh -N -L 9002:10.0.2.1:9002 -p 22
      opc@198.51.100.1
      ```

   To open an SSH Tunnel using PuTTY, follow these instructions:

   a. Start PuTTY on your Windows computer.
      The PuTTY Configuration page is displayed, showing the Session panel.

   b. In the Host Name (or IP address) field, enter the public IP address of the bastion node.

c. In the Category navigation tree, expand Connection, and then click **Data**.

d. In the Auto-login username field, enter `opc`.

e. In the When username is not specified field, select **Prompt**.

f. In the Category tree, expand Connection, and then click **SSH**.

g. Under Protocol options, select the **Don't start a shell command at all** check box.

h. In the Category tree, expand SSH, and then click **Auth**.

i. Under Private key file for authentication, click **Browse**.

j. Navigate to the location of your private key file, and select it. Click **Open**.This private key corresponds to the public key that you specified when you created this service instance.

> **Note:**
>
> The `.ppk` file extension indicates that the private key is in PuTTY's proprietary format. You must use a key of this format when using PuTTY. If Oracle Cloud generated this key for your service instance, see the PuTTY documentation for information about converting the key format

k. In the Category tree, expand SSH, and then click **Tunnels**.

l. In the Destination field, enter *IP:port* where IP is the private IP address of the WebLogic Server node and port is the port number on the node to which you want to connect.

m. In the Source Port field, enter the same port number.

n. Click the **Add** button.

o. Optional: To save this session configuration, click **Session** in the Category tree, and then click **Save**. To load a saved configuration, select the configuration name, and then click **Load**.

p. Click **Open.**

q. If prompted, enter the passphrase for the private key

Create SSH Tunneling Using the Bastion Service

Create a Bastion service and dynamic port forwarding with a secure shell (SSH) utility.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. If you have already created a Bastion service and a session in the required VCN, navigate to the Bastion and the Session, and then continue from step 18.
3. Click the navigation menu Navigation Menu icon, select Identity & Security. Under the Identity & Security group, click **Bastion**.
4. From the Compartment drop-down list, select the compartment in which your domain is created.
5. Click **Create bastion**.
6. Enter a name for the bastion.
7. Under Configure networking, select the Target virtual cloud network of the target resource that you intend to connect to by using sessions hosted on this bastion.

8. Select the Target subnet.
   The subnet must either be the same as the target resource's subnet or it must be a subnet from which the target resource's subnet accepts network traffic.
9. In CIDR block allowlist, add one or more address ranges in CIDR notation that you want to allow to connect to sessions hosted by this bastion.
   Enter a CIDR block, and then either click the value or press Enter to add the value to the list. The maximum allowed number of CIDR blocks is 20.
10. From the list of Bastion, click the name of the Bastion you created.
11. Click **Create session**.
12. From the Session type drop-down list, select SSH port forwarding session.
13. Under Connect to the target host option, select Instance name.
14. From the Compute instance, select the domain instance that has the Administration Server node.
    The instance with the Administration Server node has wls-0 appended to the name. For example: `abcde7xy-wls-0`
15. In the Port field, enter the WebLogic administration server's listener port number, where the administration console is accessible. By default, the port number is 9002.
16. Under Add SSH Key, provide the public key file of the SSH key pair that you want to use for the session.
17. Click Create Session.
18. Click **Actions** for the session you created, and select View SSH command.
19. In the View SSH command window, click **Copy** to copy the SSH command.
20. Paste the SSH command in a text editor.
21. In the SSH command, replace <*privateKey*> with the path to the private key that corresponds to the public key that you specified when you created the domain.
22. In the SSH command, replace the <*localPort*> with the preferred local port number.
23. Copy the updated SSH command.
24. From your computer, open a SSH utility.
25. Paste the SSH command and then press Enter.
    If prompted to continue connecting, type `Yes` and press Enter.

Configure Admin Server Provider using Proxy

After you have set up the SSH tunneling, you must configure a new provider in WebLogic Remote Console.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu Navigation Menu icon, select Compute. Under the Compute group, click **Instances**.
3. From the Compartment drop-down list, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.
   The instance with the Administration Server node has wls-0 appended to the name. For example: `abcde7xy-wls-0`

5. Copy the private IP address value that will be referred to as the *admin_server_private_ip_address*.
6. Open the WebLogic Remote Console and create a new Admin Server Connection Provider.
7. In the provider information panel, provide the following values:
   a. **Name**: Use a name that can easily correlate to this WebLogic domain e.g. abcde7xy
   b. **Provide Credentials**: There are two methods to provide credentials in WebLogic 14.1.2:
      - specify the Username and Password fields in the provider
      - ask the Remote Console to open a web browser to authenticate using Single Sign on. This option requires you to re-authenticate when your session is expired and configure your browser with Socks4.
   c. **URL**: https://*localHost:localPort*
      For example:

      ```
      https://127.0.0.1:9002
      ```

   d. **Proxy Override**: Leave this field empty.
   e. **Make Insecure Connection**: Selected
      The certificates used in WebLogic Server for OCI are issued to OCI DNS which is not resolvable outside. If you want to verify for Domain name information, then you can create a local host alias to point to the Admin Server Internal FQDN to the Admin Server IP address and use the FQDN instead of the IP in URL.

      > **Note:**
      >
      > You may also need to trust the CA used to sign your certificate.

   f. When you click **OK,** the Remote Console should load your WebLogic domain home.

      > **Note:**
      >
      > If you selected option “Use Web Authentication” you will first be redirected to a browser to provide credentials.

Access the Fusion Middleware Control Console

If your Oracle WebLogic Server for OCI domain includes the Java Required Files (JRF) components, you can use the Fusion Middleware Control to access and manage the domain and its Fusion Middleware components.

Topics:

- Access the Fusion Middleware Control Console in a Public Subnet
- Access the Fusion Middleware Control Console in a Private Subnet

Access the Fusion Middleware Control Console in a Public Subnet

Oracle WebLogic Server compute instances assigned to a public subnet are accessible from the public Internet.

To access the Fusion Middleware Control:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Adminstration Server node has `wls-0` appended to the name. For example: `abcde1xy-wls-0`
5. Copy the public IP address value.
6. In a browser, specify the URL of the Fusion Middleware Control in the following format, using the public IP address and appropriate port:

   ```
   https://IP-address:port/em
   ```

   The default SSL port is 7002, unless it was changed during stack creation. For domains created in secured production mode in 12.2.1.4 or14.1.2.0 then the SSL administration port is 9002 as shown in the following examples:

   Production Mode:

   ```
   https://192.0.2.1:7002/em
   ```

   Secured Production Mode (default for 12.2.1.4 or 14.1.2.0):

   ```
   https://192.0.2.1:9002/em
   ```
7. When prompted, enter the WebLogic Server administrator user name and password for this domain.

Access the Fusion Middleware Control Console in a Private Subnet

Oracle WebLogic Server compute instances assigned to a private subnet are not accessible from the public Internet.

To access the Fusion Middleware Control Console for a private domain, you can use the bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility.

By opening an SSH tunnel with dynamic port forwarding, the SSH client becomes a Socket Secure (SOCKS) proxy listening on the port you specify. All traffic that routes to the proxy port is forwarded to its destination through the proxy server. Then when you configure your browser to use a SOCKS proxy, you can access a private domain and its Fusion Middleware components.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Copy the private IP address value.
6. Return to the Compute Instances page.
7. Click the name of the bastion instance that's associated with the domain.

   The domain's bastion instance is identified by *`servicename`*`-bastion-instance`. For example: `abcde7xy-bastion-instance`
8. Copy the public IP address value.
9. From your computer, open an SSH tunnel to an unused port on the bastion node as the `opc` user. For example, you can use port `1088` for SOCKS proxy. It can be any other unused port.

   Specify the `-D` option to use dynamic port forwarding. Provide the path to the private key that corresponds to the public key that you specified when you created the domain.

   The SSH command format is:

   ```
   ssh -C -D port_for_socks_proxy -i path_to_private_key opc@bastion_public_ip
   ```

   For example:

   ```
   ssh -C -D 1088 -i ~/.ssh/mykey.openssh opc@198.51.100.1
   ```

   On a Windows platform, you can use Windows PowerShell to run the SSH command.
10. In your browser settings, set up the SOCKS (version 5) proxy configuration. Specify your local computer and the same SOCKS port that you used in your SSH command.
11. Specify the URL of the Fusion Middleware Control Console in the following format, using the private IP address:

    ```
    https://private_ip_address:port/em
    ```

    The default port is `7001`, unless it was changed during stack creation.

    For example:

    ```
    https://192.0.2.3:7001/em
    ```
12. When prompted, enter the WebLogic Server administrator user name and password for this domain.

Access the Sample Application

By default when you create an Oracle WebLogic Server for OCI domain, sample applications are deployed automatically.

How to Access the Application

To access the `sample-app` application in a browser, enter a URL that's similar to the following:

```
https://IP_address:port/sample-app
```

The IP address and port you'll use depends on whether your domain is in a public or private subnet, and whether a load balancer is configured. See:

- Access the Sample Application Through a Public Load Balancer
- Access the Sample Application Through a Private Load Balancer
- Access the Sample Application Without a Load Balancer
- Access the Sample Application in a Private Subnet

If your domain is configured to use Oracle Identity Cloud Service, then a second application is deployed named `idcs-sample-app`. See Access the Sample Application Using Identity Cloud Service.

What the Application Does

From the `sample-app` application, you can find documentation, tutorials, and other resources for Oracle WebLogic Server for OCI in the Oracle Help Center.

Use the `idcs-sample-app` application to test the integration with Oracle Identity Cloud Service. After signing in, the application displays information about the current user.

How to Manage the Application

You can verify that the sample applications are deployed and running by viewing the Deployments table in the WebLogic Server Administration Console. From the Deployments table, you can stop, start, and undeploy the applications.

Access the Sample Application Through a Public Load Balancer

If your Oracle WebLogic Server for OCI domain is associated with a public load balancer, use the public IP address of the load balancer to access the sample application.

This procedure applies to a domain created in a public or private subnet.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, and then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack is initially created, this is the compartment that contains the compute instances and network resources for the domain, or this is the network compartment that has only the network resources for the domain.
4. Click the name of the load balancer instance that's associated with your domain instance.

The load balancer instance has `-lb` appended to the domain name. For example: `abcde1xy-lb`

5. Copy the public IP address value.
6. In a browser, specify the URL of the sample application. By default, the load balancer listens for HTTPS requests on port `443`.

```
https://IP-address/sample-app/
```

Access the Sample Application Through a Private Load Balancer

If your Oracle WebLogic Server for OCI domain is in a private subnet and is associated with a private load balancer, it is not accessible from the public Internet.

To access the sample application, you can use the bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility.

By opening an SSH tunnel with dynamic port forwarding, the SSH client becomes a Socket Secure (SOCKS) proxy listening on the port you specify. All traffic that routes to the proxy port is forwarded to its destination through the proxy server. Then when you configure your browser to use a SOCKS proxy, you can access the load balancer's private IP address.

Alternatively, you can configure a virtual private network (VPN) between your VCN and your on-premise data center. See VPN Connect in the Oracle Cloud Infrastructure documentation.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, and then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack is initially created, this is the compartment that contains the compute instances and network resources for the domain, or this is the network compartment that has only the network resources for the domain.
4. Click the name of the load balancer instance that's associated with your domain instance.

   The load balancer instance has `-lb` appended to the domain name. For example: `abcde1xy-lb`
5. Copy the IP address value.
6. From the navigation menu, click **Compute**, and then click **Instances**.
7. Select the **Compartment** in which the compute instances for your domain were created.
8. Click the name of the bastion instance that's associated with the domain.

   The domain's bastion instance is identified by *`servicename`*`-bastion-instance`. For example: `abcde1xy-bastion-instance`
9. Copy the public IP address value.
10. From your computer, open an SSH tunnel to an unused port on the bastion node as the `opc` user. For SOCKS proxy, you can use port `1088`.

    Specify the `-D` option to use dynamic port forwarding. Provide the path to the private key that corresponds to the public key that you specified when you created the domain.

    The SSH command format is:

    ```
    ssh -C -D port_for_socks_proxy -i path_to_private_key opc@bastion_public_ip
    ```

For example:

```
ssh -C -D 1088 -i ~/.ssh/mykey.openssh opc@198.51.100.1
```

On a Windows platform, you can use Windows PowerShell to run the SSH command.

11. In your browser settings, set up the SOCKS (version 5) proxy configuration. Specify your local computer and the same SOCKS port that you used in your SSH command.
12. Specify the URL of the sample application in the following format, using the private IP address of the load balancer:

    ```
    https://private_ip_address/sample-app
    ```

    By default, the load balancer listens for HTTPS requests on port `443`.

    For example:

    ```
    https://192.0.2.254/sample-app
    ```

Access the Sample Application Without a Load Balancer

If your Oracle WebLogic Server for OCI domain is not associated with a load balancer, use the IP address of a Managed Server node to access the sample application.

> **Note:**
>
> This procedure does not apply to a domain created in a private subnet. See Access the Sample Application in a Private Subnet.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the compute instance that has the Administration Server and the first Managed Server.

   The instance has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Copy the public IP address value.
6. In a browser, specify the URL of the sample application using the public IP address and the Managed Server port, in one of the following formats:

   ```
   https://IP-address:7004/sample-app/
   ```

   ```
   http://IP-address:7003/sample-app/
   ```

   Unless the ports were changed during stack creation, the default Managed Server ports are `7004` (SSL) and `7003`.

For example:

```
https://198.51.100.1:7004/sample-app/
```

```
http://198.51.100.1:7003/sample-app/
```

Access the Sample Application in a Private Subnet

Oracle WebLogic Server compute instances assigned to a private subnet are not accessible from the public Internet.

If your private domain is configured with a load balancer, see:

- Access the Sample Application Through a Public Load Balancer
- Access the Sample Application Through a Private Load Balancer

To access the sample application deployed to a private domain without a load balancer, you can use the bastion instance that's created on a public subnet and dynamic port forwarding with a secure shell (SSH) utility.

By opening an SSH tunnel with dynamic port forwarding, the SSH client becomes a Socket Secure (SOCKS) proxy listening on the port you specify. All traffic that routes to the proxy port is forwarded to its destination through the proxy server. Then when you configure your browser to use a SOCKS proxy, you can access the domain's private IP address.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the compute instance that has the Administration Server and the first Managed Server.

   The instance has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Copy the private IP address value.
6. Return to the Compute Instances page.
7. Click the name of the bastion instance that's associated with the domain.

   The domain's bastion instance is identified by *`servicename`*`-bastion-instance`. For example: `abcde7xy-bastion-instance`
8. Copy the public IP address value.
9. From your computer, open an SSH tunnel to an unused port on the bastion node as the `opc` user. For SOCKS proxy, you can use port `1088`.

   Specify the `-D` option to use dynamic port forwarding. Provide the path to the private key that corresponds to the public key that you specified when you created the domain.

   The SSH command format is:

   ```
   ssh -C -D port_for_socks_proxy -i path_to_private_key opc@bastion_public_ip
   ```

For example:

```
ssh -C -D 1088 -i ~/.ssh/mykey.openssh opc@198.51.100.1
```

On a Windows platform, you can use Windows PowerShell to run the SSH command.

10. In your browser settings, set up the SOCKS (version 5) proxy configuration. Specify your local computer and the same SOCKS port that you used in your SSH command.
11. Specify the URL of the sample application in the following format, using the private IP address:

    ```
    http://private_ip_address:port/sample-app
    ```

    The default Managed Server port is `7003`, unless it was changed during stack creation.

    For example:

    ```
    http://192.0.2.254:7003/sample-app
    ```

Access the Sample Application Using Identity Cloud Service

If your Oracle WebLogic Server for OCI domain is integrated with Oracle Identity Cloud Service, then a second sample application is automatically deployed so that you can test this integration.

The application is protected by Oracle Identity Cloud Service, so you must sign in as a valid user in Oracle Identity Cloud Service.

A domain that uses Oracle Identity Cloud Service for authentication always includes a load balancer. This procedure applies to a domain created in a public or private subnet.

Part of the application requires the user to be a member of a group named `SampleAppAdmin`. Create this group in Oracle Identity Cloud Service, and add at least one user to the group. See Create Groups in Oracle Cloud Infrastructure documentation.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack is initially created, this is the compartment that contains the compute instances and network resources for the domain, or this is the network compartment that has only the network resources for the domain.
4. Click the name of the load balancer instance that's associated with your domain instance.

   The load balancer instance has `-lb` appended to the domain name. For example: `abcde1xy-lb`
5. Copy the public IP address value.
6. Sign out from the Oracle Cloud Infrastructure Console.
7. In a browser, specify the URL of the sample application. By default, the load balancer listens for HTTPS requests on port `443`.

   ```
   https://IP-address/__protected/idcs-sample-app
   ```

The URL includes two underscore characters.

For example:

```
https://192.0.2.254/__protected/idcs-sample-app
```

8. Sign in to the Oracle Identity Cloud Service as a user who is a member of the `SampleAppAdmin` group.
9. Click **Protected page**.

To protect other applications in the domain, see Secure a Domain Using Identity Cloud Service.

Start and Stop a Domain

Oracle WebLogic Server for OCI provides utilities to manage the server processes in your domain.

With these utilities you can stop, start, or restart (stop and then start) the WebLogic Server and Node Manager processes in your domain.

1. Identify the IP address of the node in your domain.

   The name of the node is *`servicename`*`-wls-`*`n`*, where `servicename` is the resource name prefix you provided during stack creation. The Administration Server runs on the first node, *`servicename`*`-wls-0`

   - If your domain is on a public subnet, then use the public IP address of the compute instance.
   - If your domain is on a private subnet, then use the public IP address of the bastion and the private IP address of the compute instance.

2. Open an SSH connection to the node as the `opc` user.

   ```
   ssh -i <path_to_private_key> opc@<node_public_ip>
   ```

   Or,

   ```
   ssh -i <path_to_private_key> -o ProxyCommand="ssh -W %h:%p -i
   <path_to_private_key> opc@<bastion_public_ip>" opc@<node_private_ip>
   ```

3. Change to the `oracle` user.

   ```
   sudo su - oracle
   ```

4. Execute the `restart_domain.sh` script.

   ```
   /opt/scripts/restart_domain.sh -o [start|stop|stopMS|restart]
   ```

   - `start` - start the Node Manager and all servers on this node
   - `stop` - stop the Node Manager and all servers on this node
   - `stopMS` - stop only the Managed Server process on this node
   - `restart` - stop and then start the Node Manager and all servers on this node

For the `stop` operation, the Administration Server and Node Manager must be running.

To start, stop, or restart all servers in the domain, run the following command:

```
/opt/scripts/restart_domain.sh -o [start|stop|restart] -servers all
```

> **Note:**
>
> To stop all servers in your domain, the Administration Server must be running, and to start all servers in your domain, the Node Manager in the virtual machine of the Administration server must be running.

If you modified certain settings after creating the domain, like the administrator password or port number, then you must provide additional parameters to `restart_domain.sh`:

- `-u` - User name for the domain administrator
- `-p` - Password for the domain administrator
- `--adminhost` - Hostname of the administration server
- `--port` - Port number of the administration server
- `--domain-name` - Name of the domain
- `--domain-home` - Home directory for the domain
- `--admin-servername` - Name of the administration server

Add IDCS After Creating a Domain

After you create a domain, you can add Oracle Identity Cloud Service (IDCS) to your Oracle WebLogic Server for OCI instance.

> **Note:**
>
> This procedure applies to domains that are created from November 2021 (Release 21.4.2) onwards. For previous releases, contact *Support*.

Prerequisites:

- Create a confidential application in IDCS to use IDCS for authentication in the domain. You will need the client ID and client secret for this confidential application. See Create a Confidential Application.
- An OCI secret with the IDCS client secret value in the tenancy. Create Secrets for Passwords. Copy the Secret OCID.
- At the `root` compartment level, create an OCI policy with the following policy statement:

  ```
  Allow dynamic-group <service-prefix>-wlsc-principal-group to read secret-
  bundles in tenancy where target.secret.id ='<secret-ocid>'
  ```

  Where, <*secret-ocid*> is the OCI secret that you obtained in the previous step.

- Add a Load Balancer, if not already configured. See Add a Load Balancer.

Complete the following steps to add IDCS to your domain:

1. Create a JSON file that contains the following information:

   ```
   {
     "is_idcs_selected" : "true",
     "idcs_host" : "<Domain name to access IDCS> (typically,
   identity.oraclecloud.com)",
     "idcs_port" : "443",
     "idcs_tenant" : "<IDCS Instance ID> (format is idcs-<GUID>)",
     "idcs_client_id" : "<Client ID of the confidential application in IDCS>",
     "idcs_client_secret_ocid" : "<Client secret of the confidential
   application>",
     "idcs_cloudgate_port" : "9999",
     "idcs_cloudgate_docker_image_tar" : "/u01/zips/APP-GATEWAY/21.2.2/
   appgateway-21.2.2-2105050509.tar.gz",
     "load_balancer_id" : "<OCID of the Load Balancer>",
     "lbip" : "<IP address of the Load Balancer>"
   }
   ```

   > **Note:**
   >
   > Add the `backend_set_name` parameter to the JSON payload if you configured a Load Balancer prior to Stack creation per the instructions in Configure the Load Balancer. This will allow the script to locate the backend set to update backends for.
   >
   > For example:
   >
   > `"backend_set_name" : "bs_lb_2024-0706-1459"`

   In the JSON file, use the client ID and client secret that you created for the confidential application. See Prerequisites.

2. Log in as a `root` user to the Administration server.

3. Save the JSON file to an accessible location.

4. Run the following command to verify if a Podman is available and displays the status as `loaded`:

   ```
   systemctl status podman
   ```

5. Run the following command:

   ```
   python3 /opt/scripts/idcs/configure_idcs.py <json-file-location>
   ```

6. Log in to each of the nodes and complete step 3 through step 5:

7. If the domain is a JRF domain, then add the OPSS SCIM template to the domain.

   a. Change to the `oracle` user.

      ```
      sudo su - oracle
      ```

b. Run the following commands only on the VM where the Administrator server is running:

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
readDomain("/u01/data/domains/<domain_name>")
addTemplate("/u01/app/oracle/middleware/oracle_common/common/
templates/wls/oracle.opss_scim_template.jar")
updateDomain()
closeDomain()
exit()
```

8. Restart the Administration server

9. If the domain is a JRF domain and you added the OPSS SCIM template, then restart the managed servers.

Back Up and Restore a Domain

Use the backup and restore capabilities of Oracle Cloud Infrastructure Block Volumes to return your Oracle WebLogic Server for OCI domain to a specific state, or to recover from a failure.

Topics:

- About Volume Backups
- About Database Backups
- Create a Backup
- Restore a Backup
- Repair a Boot Volume

About Volume Backups

The backup feature of Oracle Cloud Infrastructure Block Volumes lets you make a point-in-time backup of data on a volume. You can create a new block volume from a backup, and then attach the new volume to a domain's compute instance.

Each Oracle WebLogic Server for OCI compute instance is comprised of three volumes:

- *Boot* volume - This volume contains the Linux operating system. To restore a boot volume backup, you must create a new compute instance.

  > **Note:**
  >
  > Restoring from a boot volume is not recommended. It must be performed only as a last option.

- *Block* volume (*Domain* volume) - This volume has the contents of the `/u01/data` folder, including the domain configuration.
- *Block* volume (*Middleware* volume) - This volume has the contents of the `/u01/app` folder. The *Middleware* volume is introduced in instances created after release 20.3.3 (September 29, 2020).

Oracle recommends that you create regular backups of all boot volumes and block volumes for your domain. You can create backups manually, or you can configure policies that automatically create backups based on a specific schedule, and retain the backups for a specific period of time.

The first backup of a volume is a *full* backup, which includes all changes since the volume was created. After the first backup, you can choose to create additional full backups, or to create *incremental* backups, which include only the changes since the last backup. Both backup types enable you to restore the full volume contents to the point-in-time snapshot of the volume when the backup was taken.

Backups are encrypted and stored in Oracle Cloud Infrastructure Object Storage. Therefore, backups can be restored as new volumes to any availability domain within the same region. You can also copy volume backups across regions for disaster recovery purposes.

You must restart WebLogic Server (or restart the entire compute instance) when you restore a block volume backup. To avoid downtime and ensure your applications remain available to users, Oracle recommends that you create a cluster with multiple compute instances, and that you restore the block volume for one compute instance at a time. This approach is also called a *rolling recovery*.

When you restore a block volume backup for the first compute instance, the domain's administration server will be temporarily unavailable. However, your applications do not depend on the administration server and will not be affected.

See Overview of Block Volume Backups in the Oracle Cloud Infrastructure documentation.

About Database Backups

If your Oracle WebLogic Server for OCI domain includes the Java Required Files (JRF) components, then Oracle recommends that you back up the database containing the JRF schemas.

When you restore the block storage volumes for a domain from a backup, then you can also restore the database from a backup that was taken at the same time.

The backup and restoration procedures vary depending on type of database: Oracle Autonomous Database or Oracle Cloud Infrastructure Database.

Database Type	Details
Oracle Autonomous Database	• Oracle automatically backs up your autonomous databases and retains these backups for 60 days. Automatic backups are weekly full backups and daily incremental backups. • Backing Up an Autonomous Database Manually • Restoring an Autonomous Database
Oracle Cloud Infrastructure Database	• Backing Up a Database • Recovering a Database

Create a Backup

Create a backup of all block volumes for your Oracle WebLogic Server for OCI domain.

You can either take an immediate backup of the volumes, or you can configure a backup policy that takes a future backup according to the policy's schedule.

> **Note:**
>
> Creating a new instance by restoring from a boot volume is not recommended. It must be preformed only as a last option. To backup a boot volume, see Backing Up a Volume.

To backup a block volume, complete the following steps:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. Select the **Compartment** in which your domain was created.
4. Click the name of the first compute instance for your domain.

   The instance has `wls-0` appended to the name. For example: `mydomain-wls-0`
5. Scroll down and under **Resources**, select **Attached Block Volumes**.
6. Create a manual backup of the block volume, or assign it a backup policy.
   - Create a manual backup. See Backing Up a Volume.
   - Assign a backup policy. See Policy-Based Backups.
7. Repeat from step 2 through step 6 for the remaining compute instances in your domain.

Restore a Backup

Restore the volumes for your Oracle WebLogic Server for OCI domain from a backup.

There are two main scenarios where you would restore a backup:

- Restore to a previous working domain
- Recover from an Operating System failure

Restore to a Previous Domain

If your domain configuration is corrupted, accidentally destroyed, or the domain is not behaving as expected after a change to the binaries, then you can restore the block volumes to a previous working backup.

> **Tip:**
>
> If you have a corrupted or missing domain, follow the restoration steps in this section using the data volume. Format: *`<my_domain_name>`*`-data-block-`*`node`*.
> If you have patched or updated the WebLogic binaries and had unexpected results, follow the restoration steps in this section using the `mw` block volume. Format: *`<my_domain_name>`*`-mw-block-`*`node`* .

To avoid downtime, Oracle recommends that you restore the block volume for one compute instance in your domain at a time. The remaining compute instances can continue to process client requests.

When you attach a block volume to a virtual machine (VM) compute instance, you have two options for attachment type: iSCSI or paravirtualized. Paravirtualized attachments greatly simplify the process of configuring your block storage but IOPS performance is greater for iSCSI attachments. Bare metal compute instances must use iSCSI attachments. See Overview of Block Volume.

To restore a block volume, complete the following steps:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. Select the **Compartment** in which your domain was created.
4. Click the name of the first compute instance for your domain.

   The instance has `wls-0` appended to the name. For example: `mydomain-wls-0`
5. Click **Attached Block Volumes**.
6. Identify the name of the block volume for this compute instance, and the **Attachment Type**: `paravirtualized` or `iscsi`.
7. Create a new block volume from the backup. See Restoring a Backup to a New Volume.

   For example, `mydomain-data-block-0-new` for a domain volume or `mydomain-mw-block-0-new` for the binaries volume.
8. Shut down the WebLogic Server processes on the compute instance if any are running.
   a. Connect to the compute instance as the `opc` user with a secure shell (SSH).
   b. Switch to the `oracle` user.

      ```
      sudo su - oracle
      ```
   c. Identify the process IDs of all server and Node Manager processes.

      ```
      ps -ef | grep weblogic.
      oracle <processID> ... weblogic.NodeManager
      oracle <processID> ... weblogic.Server
      oracle <processID> ... weblogic.Server
      ```
   d. Kill the Node Manager process ID, and then kill the server process IDs.

      ```
      kill -9 <processID>
      ```
9. Detach the current block volume from the compute instance's file system.
   a. Connect to the compute instance as the `opc` user with SSH.
   b. Unmount `/u01/data`.

      ```
      sudo umount /u01/data
      ```
10. Detach the current block volume from the compute instance. See Detaching a Volume.

    If the volume attachment type is iSCSI, then connect to the compute instance as the `opc` user with SSH, and follow the instructions in the **Detach Block Volume** dialog.
11. Attach the new block volume to the compute instance. See Attaching a Volume.

Select the same attachment type as the original block volume: paravirtualized or iSCSI.

12. If the attachment type is iSCSI, then connect the new block volume to the compute instance's file system. See Connecting to a Volume.

    You can copy the required commands from the **iSCSI Commands and Information** dialog.

13. From the Oracle Cloud Infrastructure Console, click **Reboot** to reboot the compute instance.

    The WebLogic Server processes start automatically.

14. Repeat all of these steps for each of the remaining compute instances in your domain.

After you restore the domain, verify that you can access the WebLogic Server administration console and your applications. See:

- Access the WebLogic Server Administration Console
- Access the Fusion Middleware Control Console
- Access the Sample Application

Recover from an Operating System Failure

If you encounter an Operating System (OS) failure, you have different options to recover your domain.

To recover your domain, complete the following:

1. Repair the boot volume. See Repair a Boot Volume.

   Did this recover your domain?

   - **Yes**: Skip the next steps.
   - **No**: Go to next step.

2. Move the WebLogic Server instance. See Move a WebLogic Server Instance.

   Did this recover your domain?

   - **Yes**: Skip the next step.
   - **No**: Go to next step.

3. If step 1 or step 2 does not recover your domain, then as a last option complete this step.

   Delete the instance and then create an instance from a boot volume backup. See Restoring a Boot Volume.

   > **Note:**
   >
   > If you restore a boot volume:
   >
   > - All the stack metadata is lost. So, no post-provisioning actions, like restart script is available.
   > - If there are no other stack instances in the same compartment, then the new instance will not inherit the UCM subscription pricing.

Repair a Boot Volume

If the boot volume for one of your Oracle WebLogic Server for OCI compute instances becomes corrupted and will not boot, you can attach it to another compute instance to troubleshoot and repair it.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. Select the **Compartment** in which your domain was created.
4. Click the compute instance that you need to repair.
5. Identify the region and availability domain for the compute instance.
6. If the compute instance is running, click **Stop**.
7. Click **Boot Volume**.
8. Click the **Actions** icon for the boot volume, and then select **Detach**.
9. Identify a working Linux compute instance in the same region and availability domain, or create a new Linux compute instance.

   See Creating an Instance.
10. Access the new compute instance with SSH.
11. Use the `lsblk` command to identify the devices and partitions that are currently configured on the compute instance.

    Example:

    ```
    sudo lsblk
    NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
    sda      8:0    0 46.6G  0 disk
    ...
    ```
12. Return to the Oracle Cloud Infrastructure Console.
13. From the Instances page, click the new compute instance.
14. Click **Attached Block Volumes**.
15. Click **Attach Block Volume**.
16. For **Block Volume**, select the boot volume that you need to repair.
17. Click **Attach**.
18. Return to the SSH session.
19. Run the `lsblk` command again, and identify the new device and partitions.

    Example:

    ```
    sudo lsblk
    NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
    sdb      8:16   0 46.6G  0 disk
    ¿¿sdb2   8:18   0    8G  0 part
    ¿¿sdb3   8:19   0 38.4G  0 part
    ```

```
¿¿sdb1    8:17    0  200M  0 part
sda       8:0     0 46.6G  0 disk
...
```

20. Use the `fsck` command to perform a consistency check of each partition on the new device.

    ```
    sudo fsck -V /dev/<partition>
    ```

    Example:

    ```
    sudo fsck -V /dev/sdb1
    sudo fsck -V /dev/sdb2
    sudo fsck -V /dev/sdb3
    ```

21. When prompted, correct any errors found in the partitions.
22. Return to the Oracle Cloud Infrastructure Console.
23. Detach the repaired boot volume from the compute instance's block volumes. See Detaching a Volume.

    If the volume attachment type is iSCSI, then connect to the compute instance with SSH, and follow the instructions in the **Detach Block Volume** dialog.

24. Navigate to the original compute instance.
25. Attach the repaired boot volume to original compute instance. See Attaching a Volume.
26. Click **Start** to start the compute instance.

Add a Certificate to the Load Balancer

When you create a domain with a load balancer, Oracle WebLogic Server for OCI configures the load balancer to use Secure Socket Layer (SSL) and also adds a demonstration self-signed certificate. Oracle recommends you upload your own SSL certificate, and then associate the certificate with the HTTPS listener.

> **Note:**
>
> This procedure applies only to domains that were created after June 2020. For domains created before June 2020, see Configure SSL for a Domain.

You can use a custom, self-signed SSL certificate, or a certificate that you've obtained from a Certificate Authority (CA). For production WebLogic Server environments, Oracle recommends that you use a CA-issued SSL certificate, which reduces the chances of experiencing a man-in-the-middle attack.

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Networking**, and then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack was initially created, this might be the same compartment that contains the compute instances for the domain.

4. Click the load balancer that was provisioned as part of your stack, *prefix*-lb.

5. Click **Certificates**.
6. Click **Add Certificate**.
7. Enter a name for your certificate.
8. Either upload the certificate file, or paste its contents into the text area.
9. If applicable, specify a CA certificate or a private key file.

   For example, if you are using a self-signed certificate, upload the corresponding private key file. See Managing SSL Certificates in the Oracle Cloud Infrastructure documentation.
10. Click **Add Certificate**, and then click **Close**.
11. After the certificate was successfully added, click **Listeners**.
12. Edit the `https` listener.
13. Select your new certificate.
14. Click **Save Changes**, and then click **Close**.

You cannot modify an existing load balancer certificate. You must add a new certificate, and then associate the listener with the new certificate.

Enable OS Management Hub After Creating a Domain

You can enable OS Management Hub after creating a domain.

If you created a domain without "Enable OS Management Hub" selected you can enable OS Management Hub at any time. See Getting Started with OS Management Hub for more details on using OS Management Hub.

Stacks created before May 2025 (Release 25.2.1) will not have OS Management Hub enabled, but will have its predecessor, OS Management Service, enabled. OS Management Service is deprecated and Compute instances with the OS Management Service Agent enabled should be migrated to OS Management Hub. See Migrating from OS Management to OS Management Hub for details on how to migrate your instances to OS Management Hub.

Renew a Certificate When Secured Production Mode Is Enabled

When certificates created by the OCI Certificate Service are close to expiration or have expired, you can renew the certificate created by Oracle WebLogic Server for OCI or specify a different certificate OCID.

When Secured Production Mode is enabled certificates are created through the OCI Certificate Service with an expiration according to the OCI Certificate Authority's Maximum Validity Duration for Certificates (Days) or 365 days, whichever is lower. When these certificates are close to expiration or have expired you can renew the certificate created by Oracle WebLogic Server for OCI or specify a different certificate OCID.

1. From your computer, run the `ssh` command to connect to the domain's Administration Server node as the `opc` user.

```
ssh -i path_to_private_key opc
@node_IP_address
```

For example:

```
ssh -i /home/myuser/mykey opc
@203
.0.
113.13
```

2. If prompted, enter the passphrase for the private key.

3. Run the following commands to renew the certificates on the domain's Administration Server node as the `oracle` user.

```
sudo su - oracle
python3 /opt/scripts/secure_mode_renew_certificate.py <optional: certificate
OCID>
exit
```

For example, if you are renewing the certificate created for you by Oracle WebLogic Server for OCI:

```
sudo su - oracle
python3 /opt/scripts/secure_mode_renew_certificate.py
exit
```

4. If IDCS is enabled use the following commands to restart the appgateway container on each Compute instance as the root user:

```
sudo bash
sudo podman container stop appgateway
sudo podman container rm appgateway
sudo /opt/scripts/idcs/run_cloudgate.sh
exit
```

5. Sign in to the correct Administration Console for your domain.

> **Note:**
>
> As of 14c (14.1.2.0.0), the WebLogic Server Administration Console has been replaced by the WebLogic Remote Console.

- For 12.2.1.4.0 domains see, Access the WebLogic Server Administration ConsoleAccess the WebLogic Console.
- For 14.1.2.0.0 domains, see Access the WebLogic Remote Console

6. From the WebLogic Server Administration Console, click **Environment|Servers** from the Domain Structure tree view on the left side of the console.

7. Select the **Control** tab.

8. Check the checkbox by the server(s) associated with the Compute instance on which you ran the `secure_mode_renew_certificate.py` script.

9. Click **Restart SSL**.

> **Note:**
>
> If you experience any issues accessing the WebLogic Console after restarting SSL on the Administration Server you can execute the `restart_domain.sh` script. See Start and Stop a Domain.

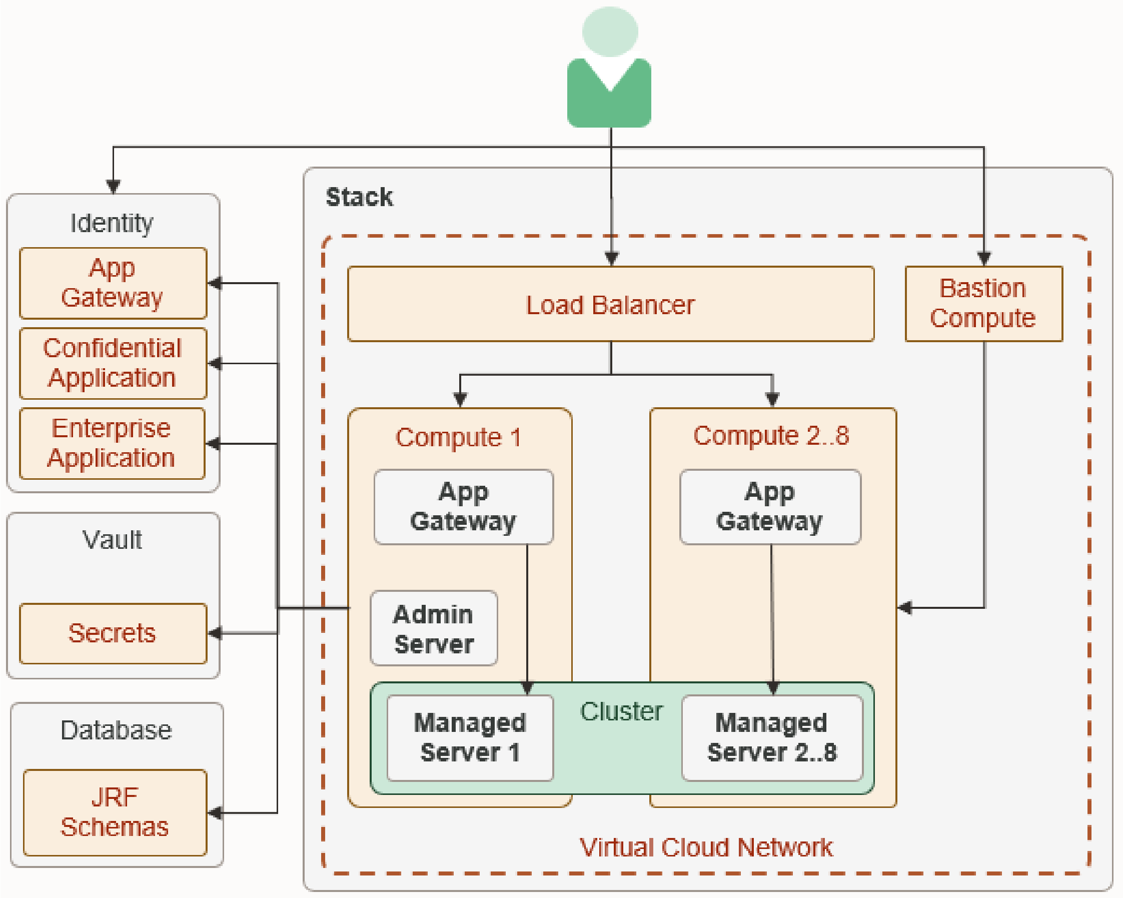
10. Repeat all the steps on each Managed Server node.

Secure a Domain Using Identity Cloud Service

Use Oracle Identity Cloud Service to protect applications and restrict administrative access for an Oracle WebLogic Server domain that you created with Oracle WebLogic Server for OCI.

By default, a domain is configured to use the local WebLogic Server identity store to maintain administrators, application users, groups, and roles. These security elements are used to authenticate users, and to also authorize access to your applications and to tools like the WebLogic Server Administration Console.

When you create a domain running WebLogic Server, you can also choose to enable Oracle Identity Cloud Service for authentication. The following diagram illustrates this configuration.



See About Oracle Identity Cloud Service Concepts in *Administering Oracle Identity Cloud Service*.

Topics:

- Create WebLogic Administrator Groups
- Update WebLogic Administrator Roles
- Update Protected Application Resources
- Update Application Deployment Descriptors

Create WebLogic Administrator Groups

Create groups in Oracle Identity Cloud Service to grant users administrative access to your domain.

Global roles in WebLogic Server control the administrative operations that a user can perform in the domain. For example, users with the `Deployer` role can deploy Java applications to the domain. By default, these roles are assigned to group names like `Deployers`. After creating these groups in Oracle Identity Cloud Service, you can add users to them.

When you create a domain, you specify a default administrative user. This user is configured in the default WebLogic Server identity store. You can use standard WebLogic Server tools like the Administration Console in order to modify this user or to change its password.

1. From the Identity Cloud Service console, expand the Navigation Drawer, and then click **Groups**.
2. Create the following groups.
   - `Administrators`
   - `Deployers`
   - `Operators`
   - `Monitors`

   By default, members of these groups will have administrative access to all domains that you create with Oracle WebLogic Server for OCI and that you configure to use Oracle Identity Cloud Service.
3. Optional: Create groups to control administrative access to a specific domain.

   For example:
   - `MyDomain_Administrators`
   - `MyDomain_Deployers`
   - `MyDomain_Operators`
   - `MyDomain_Monitors`

See these topics in *Administering Oracle Identity Cloud Service*:

- Create Groups
- Assign User Accounts to the Group

Update WebLogic Administrator Roles

Map groups in Oracle Identity Cloud Service to the administrator roles in your domain.

By default, the global administrator roles in a domain are mapped to these groups.

- `Administrators`
- `Deployers`
- `Operators`
- `Monitors`

If you do not modify the administrator roles in a domain, members of these groups in Oracle Identity Cloud Service will have access to all domains that you create with Oracle WebLogic Server for OCI and that you configure to use Oracle Identity Cloud Service.

Sign in to the WebLogic Server Administration Console for your domain. See Access the WebLogic Server Administration Console.

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Oracle WebLogic Remote Console.

1. From the WebLogic Server Administration Console, click **Security Realms**.
2. Click the default realm.
3. Click the **Roles and Policies** tab.
4. From the Roles table, expand **Global Roles**, and then expand **Roles**.
5. Click **View Role Conditions** for the `Admin` role.
6. Click the group name assigned to this role. The default is **Administrators**.
7. Enter the name of the Oracle Identity Cloud Service group to which you want to map to this role.
8. Click **OK**, and then click **Save**.
9. From the breadcrumb links at the top of the page, click **Realm Roles**.
10. Repeat from step 4 for each administrator role that you want to update.

Update Protected Application Resources

Configure the URL patterns that Oracle Identity Cloud Service uses to determine which application requests require authentication for your domain.

Oracle WebLogic Server for OCI provisions each compute instance in the domain with the App Gateway software appliance. The App Gateway acts as a reverse proxy, intercepts HTTP requests to the domain, and ensures that the users are authenticated with Oracle Identity Cloud Service.

Oracle WebLogic Server for OCI also creates an enterprise application in Oracle Identity Cloud Service for the domain. The enterprise application defines which resources require the user to be authenticated, and which resources don't require authentication.

By default, all requests whose URI begins with `/__protected` (two underscore characters followed by the word "protected") are protected. For example, a client request to the URL `https://<lb_host>/`**`__protected`**`/myapp/doaction` requires authentication, while a request to `https://<lb_host>/myapp/doaction` does not.

> **Note:**
>
> Any changes to the Oracle Identity Cloud Service applications configuration take an hour to be propagated to the App Gateway running on the compute instances.

1. From the Identity Cloud Service console, expand the Navigation Drawer, and then click **Applications**.
2. Click the enterprise application associated with your domain.

   The name of the application is `<`*`stack`*`>_enterprise_idcs_app_<`*`timestamp`*`>`. For example, `myweblogic_enterprise_idcs_app_2019-08-01T01:02:01.123456`.
3. Click the **SSO Configuration** tab.
4. Expand **Resources**.
5. Add, remove, or update the resources in this application.

   You can use regular expressions (regex) to define the protected URL patterns. If you use regular expressions, you must select the **Regex** check box.

   For example, suppose the Java applications deployed to this domain are configured to use the context roots `store` and `marketplace`. To protect all requests to the marketplace application, and also requests to the path `/store/cart`, add the following resources:

   - `/marketplace/.*`
   - `/store/cart/.*`
6. Expand **Authentication Policy**.
7. Under Managed Resources, add, remove, or update the managed resource policy for each resource.

   Set the policy's **Authentication Method** to `Form or Access Token`. You can use other authentication methods. See Developing Secure Web Applications in *Developing Applications with the WebLogic Security Service*.

   You can also change the order in which the managed resources are evaluated. If you are using the default configuration set up by Oracle WebLogic Server for OCI, you must move any added managed resources higher up the order than the `Make all paths public` resource.

   To move the order of the managed resources:

   - For Oracle Identity Cloud Service
     a. Go to your enterprise application page, and click **SSO Configuration** and then expand **Authentication Policy**.
     b. Under **Managed Resources**, right-click the resource that you created and select **Cut**.
     c. Click the `Make all paths public` resource and select **Paste Before**.
   - For Identity Domains:
     a. Go to your enterprise application page, and click **SSO Configuration**, and then click **Edit SSO configuration**.
     b. Under **Managed Resources**, click **Edit Priority** for the resource that you created to move the resource in the order of priority than the `Make all paths public` resource.

See About Enterprise Applications in *Administering Oracle Identity Cloud Service*.

Update Application Deployment Descriptors

Secure a Java application that's deployed to your domain by updating the application’s context path, security constraints, and role assignments.

Oracle WebLogic Server supports the Java Enterprise Edition declarative model for securing web applications with XML deployment descriptors.

Let's assume that you have created a managed resource with `Forms or Access Token`. Then, to ensure that the identity is propagated to the web application, complete the following steps:

1. Update the value of `context-root` in the application’s `weblogic.xml` file. Prefix the current value with one of the protected resource URLs that are defined in the Oracle Identity Cloud Service enterprise application for this domain.

   By default, the only protected context root is `/__protected` (two underscore characters followed by the word "protected"). For example:

   ```
   <context-root>/__protected/store</context-root>
   ```

2. Create one or more `security-role` elements in the application’s `web.xml` file.

   Simply list the user roles for your application. For example:

   ```
   <security-role>
     <role-name>HRAdmin</role-name>
   </security-role>
   ```

3. Create one or more `security-constraint` elements in the application’s `web.xml` file.

   Each security constraint grants access to one or more URL patterns in your application, and to specific roles. For example:

   ```
   <security-constraint>
     <web-resource-collection>
       <web-resource-name>AdminPages</web-resource-name>
       <url-pattern>/admin/*</url-pattern>
     </web-resource-collection>
     <auth-constraint>
       <role-name>HRAdmin</role-name>
     </auth-constraint>
   </security-constraint>
   ```

   Do not include the context root path in the URL patterns.

4. Create one or more `security-role-assignment` elements in the application’s `weblogic.xml` file.

   Map your application roles to specific users and/or groups found in Oracle Identity Cloud Service. For example:

   ```
   <security-role-assigment>
     <role-name>HRAdmin</role-name>
   ```

```
  <principal-name>HRManagersGroup</principal-name>
</security-role-assigment>
```

5. Set the login configuration of the application to `CLIENT-CERT`.

   ```
   <login-config>
       <auth-method>CLIENT-CERT</auth-method>
       <realm-name>default</realm-name>
   </login-config>
   ```

6. Redeploy your application for these changes to take effect.

   For example, use the WebLogic Server Administration Console.

Secure Web Services Using Identity Cloud Service

Use Oracle Identity Cloud Service and Oracle Web Services Manager to protect web service applications and clients that you deploy to Oracle WebLogic Server for OCI domains.

This configuration is applicable only for domains that you created with Oracle WebLogic Server for OCI, and that meet all of these requirements:

- Is JRF-enabled
- Includes a load balancer that is configured for HTTPS. For domains that were created before June 2020, see Configure SSL for a Domain.
- Uses Oracle Identity Cloud Service for authentication. See Access the Sample Application Using Identity Cloud Service.

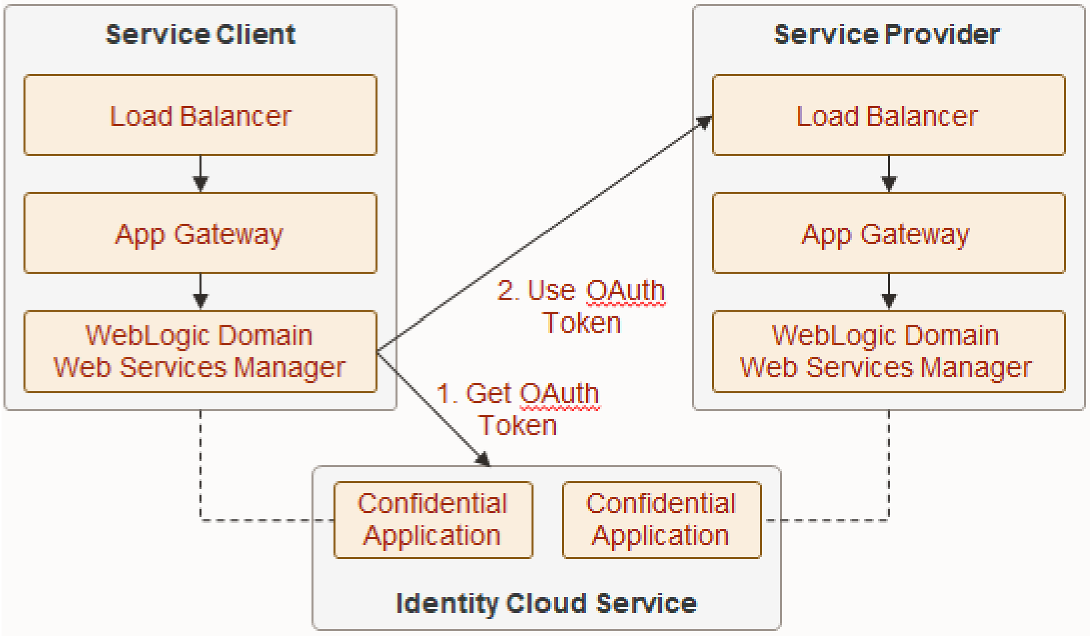
All JRF-enabled domains include Oracle Web Services Manager (OWSM), which provides a policy framework to manage and secure web services consistently across your organization. Both Oracle Web Services Manager and Oracle Identity Cloud Service support the OAuth protocol. The web service client requests an access token by authenticating with the authorization server (Oracle Identity Cloud Service) and presenting the authorization grant. The Oracle Web Services Manager server-side agent validates the access token and then accepts the client request if valid.

See Using OAuth2 with Oracle Web Services Manager in *Securing Web Services and Managing Policies with Oracle Web Services Manager*.

When you secure web service communication, the following terms are used:

- *Provider* - The Oracle WebLogic Server for OCI stack (WebLogic Server domain, load balancer, and so on) that hosts your web service application.
- *Client* - The Oracle WebLogic Server for OCI stack that hosts your web service client application.

The following diagram illustrates this security configuration.



Topics:

- Deploy a Sample Web Service Client Application
- Get Information About Identity Cloud Service
- Configure OAuth for the Web Services Provider
- Update the Confidential Application for the Web Services Provider
- Configure OAuth for the Web Services Client
- Update the Confidential Application for the Web Services Client
- Test the Sample Web Service Client

Deploy a Sample Web Service Client Application

To quickly verify the OAuth integration between Oracle Web Services Manager and Oracle Identity Cloud Service, you can build and deploy a sample client application.

This sample web application consists of a single page with an HTML form, and a single Servlet that invokes the specified web service URL. The client uses the OAuth policy `oracle/http_oauth2_token_over_ssl_idcs_client_policy`.

Alternatively, you can deploy and test your own web service client application.

1. Connect to the Administration Server node in your client domain as the `opc` user.

   ```
   ssh -i <path_to_private_key> opc@<node_public_ip>
   ```

2. Change to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. Create the web application directory structure.

   ```
   mkdir -p /home/oracle/oauth-client/WEB-INF/classes
   ```

4. Copy and paste the following text into a new file named `/home/oracle/oauth-client/index.jsp`.

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html"/>
    <title>Test web services call using wss-oauth</title>
</head>
<body>
<form name="oauth-test" method="post" action="helloworldclientservlet">
    <table>
        <tr>
            <th>Address</th>
            <td>
                <input name="address" type="text" size="110" value=""/>
            </td>
        </tr>
        <tr>
            <th>Scope</th>
            <td>
                <input name="scope" type="text" size="110" value=""/>
            </td>
        </tr>
        <tr>
            <th>Test</th>
            <td>
                <input name="submit" type="submit" size="20" value="Test"/>
            </td>
        </tr>
    </table>
</form>
</body>
</html>
```

5. Copy and paste the following text into a new file named `/home/oracle/HelloWorldClientServlet.java`.

```
import java.util.*;
import java.io.*;
import java.lang.*;
import java.security.AccessController;
import javax.security.auth.Subject;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import weblogic.jaxrs.api.client.Client;
import com.sun.jersey.api.client.WebResource;
import com.sun.jersey.api.client.config.DefaultClientConfig;
import com.sun.jersey.api.client.filter.LoggingFilter;
import com.sun.jersey.api.client.ClientResponse;
import oracle.wsm.metadata.feature.AbstractPolicyFeature;
import oracle.wsm.metadata.feature.PolicyReferenceFeature;
import oracle.wsm.metadata.feature.PolicySetFeature;
import oracle.wsm.metadata.feature.PropertyFeature;
import oracle.wsm.security.util.SecurityConstants;

public class HelloWorldClientServlet extends HttpServlet {
    @Override
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response) throws
ServletException, IOException {
        response.setContentType("text/plain");
        PrintWriter out = response.getWriter();
        String BASE_URI = request.getParameter("address");
        String SCOPE_URI = request.getParameter("scope");
        PropertyFeature scope = new
PropertyFeature(SecurityConstants.ConfigOverride.CO_SCOPE, SCOPE_URI);
        PolicyReferenceFeature clientPRF = new
PolicyReferenceFeature("oracle/
http_oauth2_token_over_ssl_idcs_client_policy", scope);
        DefaultClientConfig cc = new DefaultClientConfig();
        Map<String, Object> properties = cc.getProperties();
        properties.put(AbstractPolicyFeature.ABSTRACT_POLICY_FEATURE, new
PolicySetFeature(clientPRF));
        Client client = Client.create(cc);
        OutputStream baos = new ByteArrayOutputStream();
        PrintStream ps = new PrintStream(baos);
        client.addFilter(new LoggingFilter(ps));
        try {
            Subject subject =
Subject.getSubject(AccessController.getContext());
            Set<java.security.Principal> principals =
subject.getPrincipals();
            for (java.security.Principal principal : principals) {
                out.println("principal : " + principal.toString());
            }
            WebResource webResource = client.resource(BASE_URI);
            ClientResponse res = webResource.get(ClientResponse.class);
            ps.flush();
            out.println(baos.toString());
        } catch (Exception e) {
            String[] messages = getAllInnerErrorMessages(e);
            int count = 0;
            String allMsg = null;
            for (String mes : messages) {
                if (count != 0) {
                    allMsg += "\n|";
                    allMsg = allMsg + "\n+" + (getString("-", count * 3) +
"-> Caused By : " + mes);
                } else {
                    allMsg = mes;
                }
                count++;
            }
            out.println("The client was not able to call the service using
OAuth. The exception causes are: \n" + allMsg);
        }
        out.close();
    }
    private String[] getAllInnerErrorMessages(Throwable th) {
        List<String> all = new ArrayList<String>();
        boolean msgFound = false;
        while (th != null) {
            if (th.getMessage() != null && !
th.getMessage().trim().equals("")) {
                if (!msgFound && all.size() > 0) {
                    all.add(0, th.getMessage());
                }
                all.add(th.getMessage());
                msgFound = true;
            } else {
                all.add(th.getClass().getName());
            }
            th = th.getCause();
        }
        return all.toArray(new String[0]);
    }

    private String getString(String sp, int count) {
        StringBuffer ret = new StringBuffer();
        for (int i = 0; i < count; i++) {
            ret.append(sp);
        }
        return ret.toString();
    }

}
```

6. Copy and paste the following text into a new file named `/home/oracle/oauth-client/WEB-INF/web.xml`.

```
<?xml version = '1.0' encoding = 'windows-1252'?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://
java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         version="2.5" xmlns="http://java.sun.com/xml/ns/javaee">
    <servlet>
        <servlet-name>HelloWorldClientServlet</servlet-name>
        <servlet-class>HelloWorldClientServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloWorldClientServlet</servlet-name>
        <url-pattern>/helloworldclientservlet</url-pattern>
    </servlet-mapping>
    <security-constraint>
        <web-resource-collection>
            <web-resource-name>Success</web-resource-name>
            <url-pattern>/index.jsp</url-pattern>
        </web-resource-collection>
    </security-constraint>
    <login-config>
        <auth-method>CLIENT-CERT</auth-method>
    </login-config>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

7. Run `setWLSEnv.sh`.

   ```
   source /u01/app/oracle/middleware/wlserver/server/bin/setWLSEnv.sh
   ```

8. Compile the Servlet and put the class file in `/home/oracle/oauth-client/WEB-INF/classes`.

   ```
   cd /home/oracle
   javac -cp $CLASSPATH:/u01/app/oracle/middleware/oracle_common/modules/
   clients/com.oracle.webservices.wls.jaxws-owsm-client.jar \
   -d /home/oracle/oauth-client/WEB-INF/classes \
   HelloWorldClientServlet.java
   ```

9. Package the application.

   ```
   cd oauth-client
   zip -r ../oauth-client.war *
   cd ..
   ```

10. Use the WebLogic Scripting Tool (WLST) to deploy `oauth-client.war` to your domain.

    ```
    /u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
    connect('<admin_user>','<admin_password>','t3://
    <admin_server_IP>:<admin_server_port>')
    deploy(appName='oauth-client',path='/home/oracle/oauth-
    client.war',targets='<server_or_cluster>',upload='true')
    ```

    Example:

    ```
    /u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
    connect('weblogic','<admin_password>','t3://203.0.113.20:7001')
    deploy(appName='oauth-client',path='/home/oracle/oauth-
    client.war',targets='wlsoci_cluster',upload='true')
    ```

Get Information About Identity Cloud Service

Record configuration details about your Oracle Identity Cloud Service instance, and also download its certificates.

Before you begin, identify the confidential application in Oracle Identity Cloud Service that was created for your web services client domain. See Identity Resources for Oracle Identity Cloud Service.

1. Access the Oracle Identity Cloud Service console.
2. From the navigation menu, click **Applications**.
3. Click the confidential application that was created for your client domain.
4. Click the **Configuration** tab.
5. Under General Information, copy the **Client ID**.
6. Click **Show Secret**, and then copy the secret value.

7. From the navigation menu, click **Settings**, and then click **Default Settings**.
8. If it's not already enabled, select **Access Signing Certificate**, click **Save**, and then click **Yes**.
9. Access the following URL: `https://idcs-GUID.identity.oraclecloud.com/admin/v1/SigningCert/jwk`

   You can obtain the `GUID` for your Oracle Identity Cloud Service instance from the console URL.

   The response contains two certificates, separated by a comma.

   ```
   {"keys":[{"kty":"RSA",...,"kid":"SIGNING_KEY",...:
   ["<idcs_cert>","<idcs_root_ca_cert>"],"key_ops":
   ["verify","encrypt"],"alg":"RS256",...}]}
   ```

10. Copy and paste the first certificate, `idcs_cert`, into a new file named `idcs.cert`.

    Add the standard certificate header and footer lines.

    ```
    -----BEGIN CERTIFICATE-----
    <idcs-cert>
    -----END CERTIFICATE-----
    ```

11. Copy and paste the second certificate, `idcs_root_ca_cert`, into a new file named `idcs_ca.cert`.

    Add the standard certificate header and footer lines.

    ```
    -----BEGIN CERTIFICATE-----
    <idcs_root_ca_cert>
    -----END CERTIFICATE-----
    ```

12. Use `openssl` to view `idcs.cert` in text format.

    ```
    openssl x509 -in idcs.cert -text
    ```

13. Copy the contents of the `Subject` field, which contains a list of distinguished names (DN).

    ```
    Certificate:
        Data:
            ...
            Validity
                Not Before: Aug 27 09:20:19 2019 GMT
                Not After : Aug 27 09:20:19 2029 GMT
            Subject: <dn_list>
            ...
    ```

14. Use `openssl` to view `idcs_ca.cert` in text format, and to verify that it has no validation errors.

    ```
    openssl x509 -in idcs_ca.cert -text
    ```

15. Access the following URL: `https://idcs-GUID.identity.oraclecloud.com/.well-known/idcs-configuration`
16. From the response, copy the values for `issuer` and `token_endpoint`.

For example:

```
{
  ...
  "openid-configuration" : {
    "issuer" : "https://identity.oraclecloud.com/",
    "authorization_endpoint" : "https://idcs-GUID.identity.oraclecloud.com/
oauth2/v1/authorize",
    "token_endpoint" : "https://idcs-GUID.identity.oraclecloud.com/
oauth2/v1/token",
    ...
```

17. Return to the Oracle Identity Cloud Service console, click **Settings**, and then click **Default Settings**.
18. If **Access Signing Certificate** was disabled previously, then disable it again. Click **Save**, and then click **Yes**.

Configure OAuth for the Web Services Provider

Establish trust between Oracle Web Services Manager in your provider domain and Oracle Identity Cloud Service by importing certificates and creating global policy attachments.

The global policy affects all web services deployed to this domain. Alternatively, you can create policies for individual web services.

1. Copy the files `idcs.cert` and `idcs_ca.cert` to the Administration Server node in your provider domain as the `opc` user.

   Place the files in the `/tmp` directory.

   ```
   scp -i <path_to_private_key> idcs.cert opc@<node_public_ip>:/tmp
   scp -i <path_to_private_key> idcs_ca.cert opc@<node_public_ip>:/tmp
   ```

2. Connect to the Administration Server node in your domain as the `opc` user.

   ```
   ssh -i <path_to_private_key> opc@<node_public_ip>
   ```

3. Change the owner of the certificate files to `oracle`.

   ```
   sudo chown oracle:oracle /tmp/*.cert
   ```

4. Change to the `oracle` user.

   ```
   sudo su - oracle
   ```

5. Copy and paste the following text into a new file named `config_owsm_provider.py`.

   ```
   ip='<admin_server_IP>'
   port='<admin_server_port>'
   user='<admin_user>'
   pwd='<password>'
   dn_list='<idcs_dn_list>'
   trusted_issuer='<issuer>'
   ```

```
idcs_cert_path = '/tmp/idcs.cert'
idcs_ca_cert_path = '/tmp/idcs_ca.cert'
```

6. Update the variables in `config_owsm_provider.py`.
   - The IP address of this node
   - The port number of the Administration Server (the default is `7001`)
   - The user name and password of the domain administrator
   - The distinguished name (DN) from the Subject field in `idcs.cert`.
   - The Oracle Identity Cloud Service issuer name

   Reverse the order of the DN entries. For example, change `CN = abc,CN = def` to `CN = def,CN = abc`.

   For example:

```
ip='203.0.113.20'
port='7001'
user='weblogic'
pwd='<password>'
dn_list='CN=idcs-GUID,CN=Cloud9,CN=sslDomains'
trusted_issuer='https://identity.oraclecloud.com/'
```

7. Copy and paste the following text into `config_owsm_provider.py`, and after the variable declarations.

```
connect(user, pwd,'t3://' + ip + ':' + port)

try:
   beginWSMSession()
   createWSMPolicySet('oauth-ps-ws-service','ws-
service','Domain("*")','Global policy for default interactions for ws-
service',true)
   attachWSMPolicy('oracle/
wss11_saml_or_username_token_with_message_protection_service_policy')
   validateWSMPolicySet('oauth-ps-ws-service')
   displayWSMPolicySet('oauth-ps-ws-service')
finally:
   commitWSMSession()

try:
   beginWSMSession()
   createWSMPolicySet('oauth-ps-rest-resource','rest-
resource','Domain("*")','Global policy for default interactions for rest-
resource',true)
   attachWSMPolicy('oracle/multi_token_over_ssl_rest_service_policy')
   validateWSMPolicySet('oauth-ps-rest-resource')
   displayWSMPolicySet('oauth-ps-rest-resource')
finally:
   commitWSMSession()

try:
   beginWSMSession()
   createWSMTokenIssuerTrustDocument('trust-doc',None)
   setWSMConfiguration(None, 'TokenIssuerTrust', 'name', None, ['trust-
doc'])
   selectWSMTokenIssuerTrustDocument('trust-doc')
   setWSMTokenIssuerTrust('dns.jwt',trusted_issuer,[dn_list])
   setWSMTokenIssuerTrust('dns.sv',trusted_issuer,[dn_list])
   setWSMTokenIssuerTrust('dns.hok',trusted_issuer,[dn_list])
   setWSMTokenIssuerTrust('dns.jwt','www.oracle.com',[])
   setWSMTokenIssuerTrust('dns.sv','www.oracle.com',[])
   setWSMTokenIssuerTrust('dns.hok','www.oracle.com',[])
finally:
   commitWSMSession()

refreshWSMCache()

svc=getOpssService(name='KeyStoreService')
try:
   svc.createKeyStore(appStripe='owsm', name='keystore', 
password='',permission=true)
except Exception, ex:
   ex_msg = str(ex)
   if 'Keystore keystore in stripe owsm already exists' in ex_msg:
      print 'Keystore keystore in stripe owsm already exists. Continue...'
   else:
      raise
svc.importKeyStoreCertificate(appStripe='owsm', name='keystore', 
password='', alias='idcs-oauthkey', keypassword='', 
type='TrustedCertificate', filepath=idcs_cert_path)
svc.importKeyStoreCertificate(appStripe='owsm', name='keystore', 
password='', alias='idcs-oauthkeyca', keypassword='', 
type='TrustedCertificate', filepath=idcs_ca_cert_path)
```

8. Run `config_owsm_provider.py` using WLST.

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh 
config_owsm_provider.py
```

Verify that the script ran successfully:

```
...
Session started for modification.
The policy set was created successfully in the session.
Policy reference "oracle/
wss11_saml_or_username_token_with_message_protection_service_policy" added.
...
Creating policy set oauth-ps-ws-service in repository.
Session committed successfully.
Session started for modification.
The policy set was created successfully in the session.
Policy reference "oracle/multi_token_over_ssl_rest_service_policy" added.
...
Creating policy set oauth-ps-rest-resource in repository.
Session committed successfully.
Session started for modification.
New Token Issuer Trust document named trust-doc created.
...
Creating tokenissuertrust trust-doc in repository.
```

```
Session committed successfully.
...
Keystore created
Certificate imported.
Certificate imported.
```

Update the Confidential Application for the Web Services Provider

Update the resources that are protected by OAuth in the domain's confidential application found in Oracle Identity Cloud Service.

1. Access the Oracle Identity Cloud Service console.
2. From the navigation menu, click **Applications**.
3. Click the confidential application that was created for your provider domain.
4. Click the **Configuration** tab.
5. Under Resources, click **Register Resources**.
6. For **Primary Audience**, enter the URL of the load balancer that directs traffic to the provider domain.

   ```
   https://<lb_public_ip>:443
   ```

7. For **Scopes**, click **Add**.
8. Set the name of the new scope to `external`, and then click **Add**.
9. Click **Save**.

Configure OAuth for the Web Services Client

Establish trust between Oracle Web Services Manager in your client domain and Oracle Identity Cloud Service by importing certificates and creating global policy attachments.

The global policy affects all web service clients deployed to this domain. Alternatively, you can create policies for individual applications.

1. Connect to the Administration Server node in your client domain as the `opc` user.

   ```
   ssh -i <path_to_private_key> opc@<node_public_ip>
   ```

2. Change to the `oracle` user.

   ```
   sudo su - oracle
   ```

3. Copy and paste the following text into a new file named `config_owsm_client.py`.

   ```
   def config_policyset(policy_set_name,subject_type):
       try:
           beginWSMSession()

   createWSMPolicySet(policy_set_name,subject_type,resource_scope,desc,is_enab
   led)
           attachWSMPolicy(policy)
           setWSMPolicyOverride(policy,'token.uri',token_uri)
   ```

```
        setWSMPolicyOverride(policy,'oauth2.client.csf.key',csf_key)
        validateWSMPolicySet(policy_set_name)
    finally:
        commitWSMSession()
        displayWSMPolicySet(policy_set_name)

ip='<admin_server_IP>'
port='<admin_server_port>'
user='<admin_user>'
pwd='<password>'
client_id='<idcs_app_client_id>'
client_secret='<idcs_app_client_secret>'
token_uri = '<token_endpoint>'
dn_list = '<dn_list>'
app_resource = 'resource=<client_app_name>'
```

4. Update the variables in `config_owsm_client.py`.

   - The IP address of this node
   - The port number of the Administration Server (the default is `7001`)
   - The user name and password of the domain administrator
   - The client ID and secret of the confidential application in Oracle Identity Cloud Service that was created for the domain
   - The Oracle Identity Cloud Service token endpoint
   - The distinguished name (DN) to use for the generated certificate in Oracle Web Services Manager
   - The name of the web service client application that is deployed to the domain

   For example:

```
ip='203.0.113.30'
port='7001'
user='weblogic'
pwd='<password>'
client_id='ABCD1234efgh5678IJKL9012'
client_secret='<idcs_app_client_secret>'
token_uri = 'https://
idcs-1234abcd5678EFGH9012ijkl.identity.oraclecloud.com/oauth2/v1/token'
dn_list = 'CN=OWSM, OU=ST, O=Oracle, L=RedWood, ST=CA, C=US'
app_resource = 'resource=oauth-client'
```

5. Copy and paste the following text into `config_owsm_client.py`, and after the variable declarations.

```
connect(user, pwd,'t3://' + ip + ':' + port)

csf_key = 'idcs.oauth2.client.credentials'
createCred(map='oracle.wsm.security',key=csf_key,user=client_id,password=cl
ient_secret)

is_enabled = true
policy = 'oracle/oauth2_config_client_policy'
resource_scope = 'Domain("*")'
desc = ''
```

```
config_policyset('oauth-ps-config-rest-connection','rest-connection')
config_policyset('oauth-ps-config-rest-client','rest-client')
config_policyset('oauth-ps-config-ws-connection','ws-connection')
config_policyset('oauth-ps-config-ws-client','ws-client')
config_policyset('oauth-ps-config-ws-callback','ws-callback')
refreshWSMCache()

svc = getOpssService(name='KeyStoreService')
try:
   svc.createKeyStore(appStripe='owsm', name='keystore',
password='',permission=true)
except Exception, ex:
   ex_msg = str(ex)
   if 'Keystore keystore in stripe owsm already exists' in ex_msg:
      print 'Keystore keystore in stripe owsm already exists. Continue...'
   else:
      raise
svc.generateKeyPair(appStripe='owsm', name='keystore', password='',
dn=dn_list, keysize='2048', alias='orakey', keypassword='')

svc.exportKeyStoreCertificate(appStripe='owsm', name='keystore',
password='', alias='orakey', keypassword='', type='TrustedCertificate',
filepath='/tmp/orakey.cert')

grantPermission(appStripe=None, codeBaseURL='file:$
{common.components.home}/modules/oracle.wsm.common/wsm-agent-
core.jar',principalClass=None,principalName=None,permClass='oracle.wsm.secu
rity.WSIdentityPermission',permTarget=app_resource, permActions='assert')
```

6. Run `config_owsm_client.py` using WLST.

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
config_owsm_client.py
```

Verify that the script ran successfully:

```
...
Credential created successfully.
Session started for modification.
The policy set was created successfully in the session.
Policy reference "oracle/oauth2_config_client_policy" added.
...
Creating policy set oauth-ps-config-rest-connection in repository.
Session committed successfully.
...
Session started for modification.
The policy set was created successfully in the session.
Policy reference "oracle/oauth2_config_client_policy" added.
...
Creating policy set oauth-ps-config-rest-client in repository.
Session committed successfully.
...
Keystore created
Key pair generated
Certificate exported.
```

7. Change the owner of the generated certificate file to the `opc` user.

   ```
   exit
   sudo chown opc:opc /tmp/orakey.cert
   ```

8. Download the certificate file as the `opc` user.

   ```
   scp -i <path_to_private_key> opc@<node_public_ip>:/tmp/orakey.cert .
   ```

Update the Confidential Application for the Web Services Client

Import the Oracle Web Services Manager certificate into the domain's confidential application found in Oracle Identity Cloud Service.

1. Access the Oracle Identity Cloud Service console.
2. From the navigation menu, click **Applications**.
3. Click the confidential application that was created for your client domain.
4. Click the **Configuration** tab.
5. Under Client Configuration, select **Trusted** for **Client Type**.
6. For **Certificate**, click **Import**.
7. For **Certificate Alias**, enter a unique name.

   For example, `orakey_oauthclient`
8. Select the file `orakey.cert`, and then click **Import**.
9. Click **Add Scope**.
10. Select the confidential application of the domain that is the web services provider, and then click **Add**.
11. Click **Save**.

Test the Sample Web Service Client

If you deployed the sample client application, you can use it to quickly verify the OAuth integration between Oracle Web Services Manager and Oracle Identity Cloud Service.

1. Use the load balancer to access the sample application on your client domain.

   ```
   https://<client_lb_public_ip>/oauth-client
   ```

2. For **Address**, enter the URL of your web service provider.

   ```
   https://<provider_lb_public_ip>/<service_endpoint>
   ```

   For example:

   ```
   https://203.0.113.21/myapp/myservice
   ```

3. For **Scope**, enter this value.

   ```
   https://<provider_lb_public_ip>:443external
   ```

For example:

```
https://203.0.113.21:443external
```

4. Click the **Test** button.

Integrate OPSS User and Group APIs with Identity Cloud Service

Update your domain's confidential application in Oracle Identity Cloud Service to support the user and group lookup APIs in Oracle Platform Security Services.

This configuration is applicable only for domains that you created with Oracle WebLogic Server for OCI, and that meet all of these requirements:

- Is JRF-enabled
- Uses Oracle Identity Cloud Service for authentication. See Access the Sample Application Using Identity Cloud Service.

All JRF-enabled domains include Oracle Platform Security Services (OPSS), which provides an abstraction layer in the form of APIs that insulates developers from security and identity management implementation details. For example, developers do not need to know the details of accessing the security repository or managing keys and certificates. See Introduction to Oracle Platform Security Services in *Securing Applications with Oracle Platform Security Services*.

A domain that uses Oracle Identity Cloud Service is associated with a confidential application, which grants WebLogic Server one or more Oracle Identity Cloud Service client roles. By default, the confidential application for a JRF domain is created with the `Authenticator Client` and `Cloud Gate` roles, which enable Java applications to use the OPSS authentication APIs.

> **Note:**
>
> For a non-JRF domain, the confidential application has a single role, `Authenticator Client`.

Depending on the access required by your Java applications, you may need to add more roles to the confidential application. See AppRole Permissions in *REST API for Oracle Identity Cloud Service*.

> **Note:**
>
> Oracle recommends that you secure Java applications that access user and group information to ensure that they are accessed only by authorized users.

1. Access the Oracle Identity Cloud Service console.
2. From the navigation menu, click **Applications**.
3. Click the confidential application that was created for your domain.
4. Click the **Configuration** tab.

5. Under Client Configuration, locate **Grant the client access to Identity Cloud Service Admin APIs**, and then click **Add**.
6. Select one or more roles.

Role	Allowed Operations
Cloud Gate	Query users
User Administrator	Query and manage users and groups
Identity Domain Administrator	Access to all Identity Cloud Service operations

7. Click **Add**.
8. Click **Save**.

Upgrade the Oracle Identity Cloud Service App Gateway Version

If your Oracle WebLogic Server for OCI domain uses Oracle Identity Cloud Service for authentication, you must upgrade the App Gateway on each compute instance in the domain as an `opc` user. The latest App Gateway version is 23.4.44.

The upgrade steps are required only if both of these are true:

- You selected the **Enable Authentication Using Identity Cloud Service** option when creating the domain.
- The Oracle Identity Cloud Service App Gateway version is later than 19.2.1.

To upgrade the Oracle Identity Cloud Service App Gateway version, perform the following steps on each compute instance in the domain:

> **Note:**
>
> You must delete the existing container and recreate the container with a new version of the image.

1. Download the App Gateway Docker Image for Oracle Identity Cloud Service .
   a. Sign in to the Oracle Cloud Infrastructure Console.
   b. Expand the **Navigation Drawer**, click **Identity and Security.** From the Identity group, click **Domains**.
   c. Select any domain. If you do not see any domains in your compartment you can navigate to the root compartment.
   d. On the menu to the left click **Settings** and then click **Downloads**.
   e. In the Downloads page, click **Download** to the right of **App Gateway Docker Image for Identity Cloud Service**, and download the file, `idcs-appgateway-docker<version>.zip` to a location on your system.
   f. Navigate to the directory where you downloaded the file, and extract the contents of the zip file.

      Example:

      ```
      unzip idcs-appgateway-docker-<version>.zip
      ```

After unzip, the file, `appgateway-<version>.tar.gz` is created.

2. Download the App Gateway wallet tool (optional).
   1. Repeat steps a and b from step 1.
   2. From the App Gateway wallet tool tile, click **Download** to download the wallet file. For example, `idcs-appgateway-wallet-tool-<`*`version`*`>.zip`.
3. Copy the App Gateway Docker Image for Identity Cloud Service and App Gateway wallet tool to one of the virtual machines in the Oracle WebLogic Server for OCI instance.

   For example, copy the files, `appgateway-<`*`version`*`>.tar.gz` and `idcs-appgateway-wallet-tool-<`*`version`*`>.zip`.
4. Deploy the App Gateway Docker Image for Identity Cloud Service.
   1. In the Oracle WebLogic Server for OCI virtual machine (VM) instance, load the `.tar.gz` file to the local Podman registry.

      ```
      sudo podman load -i <.tar.gz file>
      ```

      Example:

      ```
      sudo podman load --input /tmp/appgateway-<version>.tar.gz
      ```

   2. Verify that you see the image in the local Docker registry.

      ```
      sudo podman images
      ```

      For example:

      ```
      REPOSITORY                  TAG                IMAGE ID
      CREATED       SIZE
      local.local/idcs-appgateway-docker  23.4.44-2310291619  06cb679d8b32  8
      months ago  505 MB
      localhost/idcs/idcs-appgateway      21.2.2-2105050509   58ed62ca635c  3
      years ago   624 MB
      ```

   3. Record the value from the `REPOSITORY` column after the first slash for the newly added image. Using the example from above this would be "idcs-appgateway-docker". This will be required in step 8 when setting the "idcs_cloudgate_docker_image_name" metadata value.
   4. Record the value from the `TAG` column for the newly added image. Using the example from above this would be "23.4.44-2310291619". This will be required in step 8 when setting the "idcs_cloudgate_docker_image_version" metadata value
   5. Record the value from the `REPOSITORY` column for the older image. Using the example from above this would be "localhost/idcs/idcs-appgateway". This will be required in step 10 when removing the older docker image.
   6. Record the value from the `TAG` column for the older image. Using the example from above this would be "23.4.44-2310291619". This will be required in step 10 when removing the older docker image.
5. Deploy the Oracle Identity Cloud Service App Gateway wallet file (optional).

a. Create a new `wallet_tool` directory, `/usr/lib/wallet_tool`.

```
sudo mkdir -p /usr/lib/wallettool/
```

b. Extract the `idcs-appgateway-wallet-tool zip` to `/usr/lib/wallet_tool`.

```
sudo unzip /tmp/idcs-appgateway-wallet-tool-<version>.zip -d /usr/lib/
wallet_tool/
```

6. Create the `cwallet.sso` file (optional).

   If the wallet file is not deleted, you can use the existing wallet file (`cwallet.sso`) to upgrade to the latest App Gateway version, or upgrade the App Gateway wallet tool and generate a new `cwallet.sso` file.

   Use one of the following methods to create the `cwallet.sso` file.

   - Manual:

     a. Retrieve the client ID and client secret of the `app_gateway` using information in the `idcs_artfacts.txt` in the `/u01/data` directory.

     ```
     cat /u01/data/.idcs_artifacts.txt
     ```

     b. Take a note of the `displayName` of the `app_gateway` in `/u01/data/.idcs_artifacts.txt`.

     Example:

     ```
     {
       "confidential_app": {
         "meta": {
           "location": "https://idcs-<GUID>.identity.oraclecloud.com:443/
     admin/v1/Apps/<confidential_app_ID>"
                 }
       },
       "app_gateway": {
         "meta": {
           "location": "https://idcs-<GUID>.identity.oraclecloud.com:443/
     admin/v1/CloudGates/<app_gateway_ID>"
                 },
         "displayName":
     "idcs0706_app_gateway_2021-06-07T14:57:22.297066",
         "id": "< app_gateway_ID>"
       },
       "enterprise_app": {
         "meta": {
           "location": "https://idcs-<GUID>.identity.oraclecloud.com:443/
     admin/v1/Apps/<enterprise_app_ID>"
                 }
       }
     }
     ```

> **Note:**
>
> Note: You must belong to the *Administrator* group in Oracle Identity Cloud Service to access this information.

c. In the Oracle Identity Cloud Service console, expand the **Navigation Drawer**, click **Security**, and then click **App Gateways**. In the App Gateways page, search for the App Gateway with the noted `displayName` and take a note of the client ID and client secret.

d. Navigate to `/u01/data/cloudgate_config/` directory and create the `cwallet.sso` file.

```
cd /u01/data/cloudgate_config/


export LD_LIBRARY_PATH=/usr/lib/wallet_tool/lib/:/usr/lib


echo <client_secret>  | /usr/lib/wallet_tool/cgwallettool --create -
i <client_id>
```

- Using Scripts:

> **Note:**
>
> You can use scripts to create the `cwallet.sso` file for Oracle WebLogic Server for OCI version 21.2.3 or later; version 24.3.1 has the latest scripts to support App Gateway Docker Image for Identity Cloud Service 23.4.44.

a. Add the Oracle Identity Cloud Service client ID and client secret to `/u01/data/cloudgate_config/appgateway-env`.

b. Run the `create_idcs_cloudgate_cwallet.sh` script as a root user.

> **Note:**
>
> Make sure you are using the latest version of the `create_idcs_cloudgate_cwallet.sh` script.

Example:

```
sudo echo "" >> /u01/data/cloudgate_config/appgateway-env
sudo echo "CG_APP_NAME=<client_id>" >> /u01/data/cloudgate_config/
appgateway-env
sudo echo "CG_APP_SECRET=<client_secret>" >> /u01/data/
cloudgate_config/appgateway-env
sudo sh /opt/scripts/idcs/create_idcs_cloudgate_cwallet.sh
```

7. Stop and remove the existing App Gateway container.

   ```
   sudo podman container stop appgateway
   ```

   ```
   sudo podman container rm appgateway
   ```

8. Create and start the new App Gateway container.

   Use one of the following methods to create and start the new App Gateway container:

   - Manual:

     a. Run the `update_metadata` script to update the metadata for `docker_image_version` to the value recorded from step 4d and `docker_image_name` to point to the values recorded from step 4c.

        Example:

        ```
        sudo python3 /opt/scripts/utils/update_metadata.py -k
        idcs_cloudgate_docker_image_version -v <version>
        sudo python3 /opt/scripts/utils/update_metadata.py -k
        idcs_cloudgate_docker_image_name -v idcs-appgateway-docker
        ```

        > **Note:**
        >
        > This step to update the metadata script is required if you upgrade the Oracle Identity Cloud Service App Gateway version during scale out.

     b. Navigate to the `/u01/data/cloudgate_config/` directory and change the permissions to `777` and the owner to `8000:8000` for this directory.

        Example:

        ```
        cd /u01/data/cloudgate_config/
        sudo chmod -R 777 /u01/data/cloudgate_config/
        sudo chown -R 8000:8000 /u01/data/cloudgate_config/*
        ```

     c. Start the App Gateway container using the `podman run` command.

        > **Note:**
        >
        > You must mount the local folder, `/u01/data/cloudgate_config` volume to the directory, `/usr/local/nginx/conf/` inside the container *`<my-container>`*.
        >
        > The `cwallet.sso` file that contains the client ID and client secret must be copied to the folder, `/usr/local/nginx/conf/` in the container so that the container can reference the wallet file.

Example:

```
sudo podman run -it -d --name appgateway --log-driver=journald --
security-opt label=disable --env-file /u01/data/cloudgate_config/
appgateway-env /
--env HOST_MACHINE=`hostname -f` --env CLOUDGATE_VERSION=<version> /
--volume /u01/data/cloudgate_config/:/usr/local/nginx/conf/:z /
--net=host idcs/idcs-appgateway:<version>
```

- Using Scripts:

  > **Note:**
  >
  > You can use scripts to create the `cwallet.sso` file for Oracle WebLogic Server for OCI version 21.2.3 or later; version 24.3.1 has the latest scripts to support Oracle Identity Cloud Service App Gateway version 23.4.44.

  a. Run the `update_metadata` script to update the metadata for `docker_image_version` and `docker_image_name` to point to latest version.

     Example:

     ```
     sudo python3 /opt/scripts/utils/update_metadata.py -k
     idcs_cloudgate_docker_image_version -v <version>
     sudo python3 /opt/scripts/utils/update_metadata.py -k
     idcs_cloudgate_docker_image_name -v idcs/idcs-appgateway
     ```

  b. Start the App Gateway container using `run_cloudgate.sh`.

     ```
     sudo sh /opt/scripts/idcs/run_cloudgate.sh
     ```

9. Verify the upgrade.

   a. Check the App Gateway container logs.

      ```
      sudo podman logs appgateway
      ```

   b. Log in and connect to the container using `bash`.

      Example:

      ```
      sudo podman exec -it appgateway bash
      ```

   c. Navigate to the `bin` folder in the container, and check the `cloudgate-env` file.

      Example:

      ```
      cd /usr/local/nginx/logs/
      cd /scratch/oracle/idcs-cloudgate/latest/bin/
      ./cg-env
      ```

10. Remove the existing App Gateway Docker Image by executing the following command using the value recorded in step 4e for "existing_repository_name" and the value recorded in step 4f for "existing_version".

    ```
    sudo podman image rm opc-delivery.docker.oraclecorp.com/idcs/
    <container_name:existing_version>
    ```

11. Repeat from **step 3** for all remaining compute instances in this domain.

Configure Session Persistence

If you created a domain with Oracle WebLogic Server for OCI, and your web applications require session persistence, you can update the load balancer for the domain.

Session persistence is a method to direct all requests originating from a single logical client to a single backend server. By default, session persistence is enabled on the load balancer with the `Enable load balancer cookie persistence` option, but you can update the load balancer after creating a domain.

Oracle Cloud Infrastructure Load Balancing supports two types of session persistence (stickiness):

- Application cookie persistence - The load balancer activates session persistence when a backend server sends a `Set-Cookie` response header containing a recognized cookie name.
- Load balancer cookie persistence - The load balancer inserts a cookie into the response, which enables session persistence. This method is useful when you have applications that cannot generate their own cookies.

To configure session persistence:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, and then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack is initially created, this is the compartment that contains the compute instances and network resources for the domain, or this is the network compartment that has only the network resources for the domain.
4. Click the name of the load balancer that's associated with your domain.

   The load balancer has `-lb` appended to the domain name. For example: `abcde1xy-lb`
5. Click **Backend Sets**.
6. Click the backend set for this load balancer.
7. Click **Edit**.
8. Modify the **Session Persistence** configuration.

   For example, if you select **Enable Application Cookie Persistence**, and set **Cookie Name** to `'*'`, then responses that contain any cookie will be directed to the same server in the domain.

See these topics in the Oracle Cloud Infrastructure documentation:

- Session Persistence
- Managing Backend Sets

Enable Session Affinity or Sticky Sessions

If you have clustered JMS resources and you access it by using an external client through a load balancer by rmi-tunneling, then you need to enable session affinity or sticky sessions.

Complete the following steps:

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Oracle WebLogic Remote Console.

1. Access the WebLogic console of the clone instance. See Access the WebLogic Server Administration Console.
2. Under **Domain Structure**, expand **Environment** > **Clusters**, and then click **Cluster-1**.
3. In the right pane, change the **Default Load Algorithm** property to `RoundRobinAffinity`.
4. Click **Save**.
5. Restart the WebLogic Server instances.

Update the Password Secret OCID, Policy, and User Name

If you have moved your password to new secret or vault or updated the password or user name, then you must update the password secret OCID and policy to read the secrets with new secrets OCID.

Moved your password to new secret or vault

If you have moved your password to new secret or vault, complete the following steps:

1. Update the metadata on each node by using the `update_metadata.py` script, which is located at `/opt/scripts/utils/`:

   ```
   python3 /opt/scripts/update_metadata.py -k wls_admin_password_ocid -v
   <new_secrets_ocid>
   ```

   > **Note:**
   >
   > The command creates a backup of the previous setup under `/opt/scripts/utils/metadata_backup_<time_stamp>.txt`.

2. Update the `servicename-secrets-policy` policy to read the secrets with new secrets OCID.

Updated your user name

If you have updated the WebLogic Server Administrator user name, update the metadata by using the `update_metadata.py` script, as shown below:

```
python3 /opt/scripts/update_metadata.py -k wls_admin_user -v
<new_wls_admin_username>
```

Update your password in the existing secret

If you have updated your password, complete the following steps:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Identity & Security**, and then click **Vault**.
3. Under **List Scope**, in the **Compartment** list, click the name of the compartment that contains the vault with the secret you want to provide with new secret contents.
4. From the list of vaults in the compartment, click the vault name.
5. Click **Secrets**, and then click the name of the secret with the secret contents you want to update.
6. Click **Create Secret Version**.
7. Specify the format of the secret contents you're providing by choosing a template type from the **Secret Type Template** list.
8. Click **Secret Contents**, and then enter the secret contents.
9. Click **Create Secret Version**.
10. Update the `servicename-secrets-policy` policy to read the secrets with new secrets OCID.

Update Password for the weblogic User

If you update the password for the *weblogic* user, ensure that Oracle WebLogic Server for OCI secrets and metadata are updated with the new password.

To update the password:

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Oracle WebLogic Remote Console.

1. Sign in to the WebLogic Server Administration Console.
2. Under **Domain Structure**, click **Security Realms**.
3. In the Summary of Security Realms page, select **myrealm**, and then click **Users and Groups** tab.
4. Click the **passwords** tab for the *weblogic* user and update the password.
5. Under **Domain structure**, select the domain name, and then select the **Security** tab.
6. Click the **Advanced** tab and change the password for the Node Manager (the user name here should be set to *weblogic* by default).

7. `ssh` to Oracle WebLogic Server for OCI administration VM, take a backup, and update the `boot.properties` file under `DOMAIN_HOME/servers/<Admin_server>/security` with the clear text user name (*weblogic*) and the updated password.
8. Take a backup and update the `boot.properties` file under the `$DOMAIN_HOME/servers/<AdminServer>/data/nodemanager` directory.
9. Take a backup and rename the `boot.properties` file for the managed server in the `$DOMAIN_HOME/servers/<managed_server>/data/nodemanager` directory.
10. Restart the entire domain using the Oracle WebLogic Server for OCI restart scripts. See Start and Stop a Domain.

> **Note:**
>
> Use the newly created password when you are prompted for *weblogic* and Node Manager passwords while running the restart script.

Update the new password in the existing Oracle WebLogic Server for OCI secret in your OCI Vault. For instructions, see Update the Password Secret OCID, Policy, and User Name.

About Deleting Secrets and Policies

Policies control your access to Oracle Cloud Infrastructure and secrets are used to store passwords such as the administrator password. These policies and secrets should be set up prior to creating the Oracle WebLogic Server for OCI instances. However, you have the option to delete these policies and secrets after you provision a Oracle WebLogic Server for OCI domain.
The following are the effects of deleting the policies and secrets post provisioning:

- Removing the administrator password secret policy will not allow you to scale out compute instances. To enable scale out, you will need to create a new secret and add back the policy that references the new secret. Additionally, ensure that you specify the new secret in the stack UI or Terraform if you are using CLI.
- Deleting the administrator password secret and policy would impact the functionality of the `restart_domain.sh` script to some extent. You would have to type in a password each time you use the start, stop, and restart options. However, a VM reboot will start the servers because the reboot does not rely on a password.
- Removing the database secret and policy would affect the functionality of the `delete_rcu.sh` script. You would need to type the WebLogic administrator password and the password for the database user with the SYSDBA role.
- Removing the IDCS secret and policy would affect the functionality of the `delete_idcs_application.sh` script. You would need to type in the client secret for the IDCS confidential application.

If you want to scale out an instance, you will need to add back the dynamic group policy statement provided below to the policy at the `root` compartment level.

```
Allow dynamic-group MyInstancesPrincipalGroup to read secret-bundles in
tenancy where target.secret.id = '<OCID_of_wls_password_secret>'
```

Update the Infrastructure Schema Password

When you log in to your administration instance, you receive a message about password expiry of your infrastructure schema passwords, so that you can update the password.

For JRF-enabled Oracle WebLogic Server for OCI instances created after release 23.1.1, if your Oracle Platform Security Services (OPSS) schema password has expired, you see the password expiry date message, and you must update your schema password. You also see a message if password expiry is less than or equal to 30 days, and you can update your schema password. However, if password expiry is greater than 30 days, no message is displayed.

You can update passwords for OPSS and other JRF data sources, if the password has expired (administration server is not running) or if the password has not expired (administration server is running).

Tasks:

- Update the Data Source Password if the WebLogic Administration Server Is Running
- Update the Data Source Password if the WebLogic Administration Server Is Not Running

Update the Data Source Password if the WebLogic Administration Server Is Running

You must update the password for all the data sources through the WebLogic console.

The following table lists the data sources and their corresponding user names.

> **Note:**
>
> As of release 14c (14.1.2.0.0), the WebLogic Server Administration Console has been removed. For comparable functionality, you should use the WebLogic Remote Console. For more information, see Oracle WebLogic Remote Console.

Table 3-1 Data Sources and User Names

Data Source	User Name
`LocalSvcTblDataSource`	`schema_prefix_STB`
`mds-owsm`	`schema_prefix_MDS`
`opss-audit-DBDS`	`schema_prefix_IAU_APPEND`
`opss-audit-viewDS`	`schema_prefix_IAU_VIEWER`
`opss-data-source`	`schema_prefix_OPSS`
`WLSSchemaDataSource`	`schema_prefix_WLS_RUNTIME`

To update the data source password:

> **Note:**
>
> You must perform steps 1, 2, and 4 to update the password for each of these data sources listed in Table 3-1, except the OPSS data source, `opss-data-source`. For OPSS data source, `opss-data-source`, perform steps 1 to 4.

1. Stop the data source.
   1. Access the WebLogic console. See Access the WebLogic Server Administration Console.
   2. Click **Lock & Edit**.
   3. From the **Domain Structure** panel, expand **Services**, and then click **Data Sources**.
   4. Click the Data Source name, and on the **Configuration** tab, click **Connection Pool**.
   5. In the **Properties** field, identify the schema prefix and the schema user that has the format `schema_prefix_user`.
      For example, if the format is `SP737755846_MDS`, the schema prefix is `SP737755846` and the schema user is `MDS`.
   6. Save the changes.
   7. Click the **Control** tab and select the instance that you want to stop.
   8. Click **Shutdown**, and then select **Force Shutdown Now**, and then click **Yes** to confirm.
   9. In the Change Center, click **Activate Changes**.
2. Change the password for the database schema.
   1. Connect to your database using SQL Plus or SQL Developer.
      - For an Oracle Cloud Infrastructure Database (DB System), connect to the database using SQL Plus as follows:
        1. Connect to the Administration Server node in your service instance using SSH.
        2. Switch to the `oracle` user.

           ```
           sudo su oracle
           ```

        3. Connect to the database using `sqlplus`.

           ```
           sqlplus / as sysdba
           ```

      - For an Autonomous database, connect to the database using SQL Developer. See Connect to Autonomous Database on Dedicated Exadata Infrastructure with Oracle SQL Developer in Oracle Cloud Infrastructure documentation.
   2. If your Oracle Cloud Infrastructure Database uses PDB, set the name of the pluggable database (PDB).

      ```
      alter session set container=PDB_name;
      ```

> **Note:**
>
> This step is not required if you are using an Autonomous Database.

c. List all the schemas (users) in the database.

```
select username from dba_users;
```

d. Locate the schema prefix like `schema_prefix_MDS`, and note the generated schema prefix. For example: `SP737755846_MDS`

e. Unlock and change the password for the user:

```
alter user schema_prefix_user account unlock;
alter user schema_prefix_user identified by new_password;
```

For example:

```
alter user SP737755846_MDS account unlock;
alter user SP737755846_MDS identified by new_password;
```

3. Update the data source password and start the data source.
   a. Access the WebLogic console.
   b. Click **Lock & Edit**.
   c. From the **Domain Structure** panel, expand **Services**, and then click **Data Sources**.
   d. Click the Data Source name, and on the **Configuration** tab, click **Connection Pool**.
   e. For the schema user name that you identified in step 1e, update the password.
   f. Save the changes.
   g. Click the **Control** tab and select the instances that you want to start.
   h. Click **Start**, and then click **Yes** to confirm.
   i. In the Change Center, click **Activate Changes**.
4. Use the WebLogic Scripting Tool (WLST) to update the bootstrap credentials for the OPSS database schema.

> **Note:**
>
> This step is applicable only for OPSS data source, `opss-data-souce`.

a. Connect to the Administration Server node in your service instance using SSH.

b. Switch to the

```
oracle
```

user.

```
sudo su oracle
```

c. Identify your domain’s name.

```
ls /u01/data/domains
```

d. Start a WLST prompt.

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
```

e. Run the `modifyBootStrapCredential` command. Specify the full path to the `jps-config-jse.xml` file, the OPSS schema name, and your new database password.

```
modifyBootStrapCredential(jpsConfigFile='/u01/data/domains/domain_name/
config/fmwconfig/jps-config-
jse.xml',username='schema_prefix_OPSS',password='new_password')
```

f. Exit WLST.

```
exit()
```

Update the Data Source Password if the WebLogic Administration Server Is Not Running

To update the data source password:

1. Stop the servers in your domain.

   You must stop the servers so that database account does not get locked again after changing the password due to too many bad password connections.

   a. Connect to the Administration Server node in your service instance using SSH.

   b. Switch to the `oracle` user.

   ```
   sudo su oracle
   ```

   c. Run the `restart_domain.sh` script.

   ```
   /opt/scripts/restart_domain.sh -o stop
   ```

   You must run the script on every compute instance.

2. Change the password for the database schema.

   a. Connect to your database using SQL Plus or SQL Developer.

   - For an Oracle Cloud Infrastructure Database (DB System), connect to the database using SQL Plus as follows:

     i. Connect to the Administration Server node in your service instance using SSH.

ii. Switch to the `oracle` user.

```
sudo su oracle
```

iii. Connect to the database using `sqlplus`.

```
sqlplus / as sysdba
```

- For an Autonomous database, connect to the database using SQL Developer. See Connect to Autonomous Database on Dedicated Exadata Infrastructure with Oracle SQL Developer in Oracle Cloud Infrastructure documentation.

b. If your Oracle Cloud Infrastructure Database uses PDB, set the name of the pluggable database (PDB).

```
alter session set container=PDB_name;
```

> **Note:**
>
> This step is not required if you are using an Autonomous Database.

c. List all the schemas (users) in the database.

```
select username from dba_users;
```

d. Locate the schema prefix like `schema_prefix_MDS`, and note the generated schema prefix. For example: `SP737755846_MDS`

e. Unlock and change the password for the user.

```
alter user schema_prefix_user account unlock;
alter user schema_prefix_user identified by new_password;
```

For example, to unlock the change the password for the `MDS` user:

```
alter user SP737755846_MDS account unlock;
alter user SP737755846_MDS identified by new_password;
```

Ensure that you unlock and change the password for the following users.

```
schema_prefix_STB
schema_prefix_MDS
schema_prefix_IAU_APPEND
schema_prefix_IAU_VIEWER
schema_prefix_OPSS
schema_prefix_WLS_RUNTIME
```

3. Update the domain's configuration files.

a. Encrypt your new schema password using the `weblogic.security.Encrypt` utility.

```
source $DOMAIN_HOME/bin/setDomainEnv.sh
java weblogic.security.Encrypt
```

When prompted, enter the new password.

b. Copy the encrypted password.

c. Navigate to the directory that contains your domain's data source configuration files.

```
cd config/jdbc
```

d. Edit the following files and update the `password-encrypted` element with the new encrypted value.

```
LocalSvcTblDataSource-jdbc.xml
opss-auditview-jdbc.xml
mds-owsm-jdbc.xml
opss-datasource-jdbc.xml
opss-audit-jdbc.xml
WLSSchemaDataSource-jdbc.xml
```

```
<password-encrypted>encrypted_password</password-encrypted>
```

4. Use the WebLogic Scripting Tool (WLST) to update the bootstrap credentials for the OPSS database schema.

> **Note:**
>
> This step is applicable only for OPSS data source, `opss-data-source`.

a. Connect to the Administration Server node in your service instance using SSH.

b. Switch to the oracle user.

```
sudo su oracle
```

c. Identify your domain's name.

```
ls /u01/data/domains
```

d. Start a WLST prompt.

```
/u01/app/oracle/middleware/oracle_common/common/bin/wlst.sh
```

e. Run the `modifyBootStrapCredential` command. Specify the full path to the `jps-config-jse.xml` file, the OPSS schema name, and your new database password.

```
modifyBootStrapCredential(jpsConfigFile='/u01/data/domains/domain_name/
config/fmwconfig/jps-config-
jse.xml',username='schema_prefix_OPSS',password='new_password')
```

f. Exit WLST.

```
exit()
```

5. Start the servers.

a. Connect to the Administration Server node in your service instance using SSH.
b. Switch to the `oracle` user.

```
sudo su oracle
```

c. Start the servers using the `restart_domain.sh` script.

```
/opt/scripts/restart_domain.sh -o start
```

You must run the script on every compute instance.

Manage Data Sources

After you create an Oracle WebLogic Server for OCI domain, you can create data sources that enable you to connect to either an Oracle Autonomous Database or an Oracle Cloud Infrastructure Database (DB System) database.

Creating a data source for a DB System database or Oracle Autonomous Database is similar with one exception. With an Oracle Autonomous Database, you need the Oracle Autonomous Database client credentials or wallet files.

Oracle WebLogic Server for OCI supports the same database versions and drivers as those for on-premise WebLogic Server installations. Refer to the following excel files (`xls`) at Oracle Fusion Middleware Supported System Configurations:

- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.2.0.0)
- System Requirements and Supported Platforms for Oracle WebLogic Server 14c (14.1.1.0.0)
- System Requirements and Supported Platforms for Oracle Fusion Middleware 12c (12.2.1.4.0)

> **Note:**
>
> From release 21.4.3 (December 9, 2021) onwards, you cannot provision a domain in Oracle Oracle WebLogic Server for OCI for Oracle WebLogic server versions 11g (10.3.6.0) and 12c (12.2.1.3) from the Marketplace.

Topics:

- Create a Data Source for an Oracle Autonomous Database
- Create a Data Source for a DB System Database

Create a Data Source for an Oracle Autonomous Database

Oracle WebLogic Server for OCI provides two utility scripts to help you create Oracle Autonomous Database:

- A download script that downloads the Oracle Autonomous Database wallet files to a node
- A create script that creates the data source using the downloaded Oracle Autonomous Database wallet files and data source properties you provide

- *Prerequisite*: At the `root` compartment level, create an OCI policy with the following policy statement:

  ```
  Allow dynamic-group <service-prefix>-wlsc-principal-group to use
  autonomous-transaction-processing-family in compartment id <compartment-id>
  ```

To run the scripts, you need to access the nodes in your WebLogic domain as the `opc` user. The scripts are located in `/opt/scripts/utils` and can only be run as the `oracle` user.

The Oracle Autonomous Database must allow the WebLogic Server compute instances to access the database listen port (1521 by default). Update your access control list (ACL), if necessary. See Security Tools for Serverless Deployments.

Tasks:

- Download the Oracle Autonomous Database Wallet
- Configure a Data Source for an Oracle Autonomous Database

Download the Oracle Autonomous Database Wallet

The download script unpacks and copies the Oracle Autonomous Database wallet contents to a node.

> **Note:**
>
> The download script must be run before the create script that configures a data source.

If the data source target is the domain cluster, you must run the download script on every node in the cluster. If the target is individual servers, then run the script on those servers.

You need the public IP address of each node on which you plan to run the download script. Find the IP address on the compute instance details page in the Oracle Cloud Infrastructure console. Look up the bastion's public IP address and the private IP address of a node if the WebLogic domain is in a private subnet.

1. Open an SSH connection to a node as the `opc` user.

   ```
   ssh -i <path_to_private_key> opc@<node_public_ip>
   ```

   Or,

   ```
   ssh -i <path_to_private_key> -o ProxyCommand="ssh -W %h:%p -i
   <path_to_private_key> opc@<bastion_public_ip>" opc@<node_private_ip>
   ```

2. Change to the `oracle` user.

   ```
   sudo su oracle
   ```

3. At the `root` compartment level, create an OCI policy with the following policy statement:

   ```
   Allow dynamic-group <service-prefix>-wlsc-principal-group  to read
   autonomous-database in compartment id <atp-compartment-id>
   ```

4. Run the script `download_atp_wallet.sh` by providing the following parameters:

   - OCID of the Oracle Autonomous Database- You can find the OCID from the Oracle Autonomous Database details page in the Oracle Cloud Infrastructure console.
   - Path to save the extracted Oracle Autonomous Database wallet files - The path to a directory on the domain where the script saves the extracted Oracle Autonomous Database wallet files. For example: `/u01/data/domains/thestack_domain/config/atp`

     The directory must be identical on every node where you run the script.

   Command:

   ```
   /opt/scripts/utils/download_atp_wallet.sh <atp_database_ocid>
   <path_to_extract_wallet_files>
   ```

   Example:

   ```
   /opt/scripts/utils/download_atp_wallet.sh
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v /u01/data/domains/
   servicename_domain/config/atp
   ```

   You will be prompted to enter the ATP Wallet Password that you would like to use. The password must be at least eight characters long, and include at least one letter and either one numeric character or one special character. Ensure that you record this password because you will need it at the time of creating the data source.

   The download script creates a subdirectory in the path you provide using the Oracle Autonomous Database OCID value. For example:

   ```
   /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v
   ```

   Seven files are extracted to the subdirectory. The following is an example of the script response:

   ```
   <Aug 22, 2019 10:39:50 PM GMT> <INFO> <oci_utils> <(host:servicename-
   wls-0.subnet_dns_domain_name) - <WLSC-VM-INFO-001> ATP Wallet downloaded>
   Archive:  /tmp/atp_wallet.zip
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/cwallet.sso
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/tnsnames.ora
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/truststore.jks
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/ojdbc.properties
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/sqlnet.ora
     inflating: /u01/data/domains/servicename_domain/config/atp/
   ```

```
ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/ewallet.p12
  inflating: /u01/data/domains/servicename_domain/config/atp/
ocid1.autonomousdatabase.oc1.phx.a1b2c3d4e56z7y8x9w10v/keystore.jks
```

5. Repeat steps 1 through 3 on each node where you have to run the download script. Depending on the data source target, run the download script on every node in the cluster or on individual servers.

Configure a Data Source for an Oracle Autonomous Database

The create script configures a JDBC data source using the downloaded Oracle Autonomous Database wallet files and the data source properties you provide.

> **Note:**
>
> You must run the download script before you run the create script to configure the data source.

When you execute the create script, you can let the script prompt you for the properties one at a time, or you can supply the properties in a configuration file.

1. Before you run the create script, prepare the required information for the following data source properties. If you plan to provide a properties configuration file when you run the create script, put each property name and value pair on one line. For example:

```
ds.name=myds1
ds.jndi.name=jdbc/myds1
```

Property Description	Property Name	Required
Name for the JDBC data source	`ds.name`	Y
OCID of the Oracle Autonomous Database	`atp.db.id`	Y
Name of the Oracle Autonomous Database	`atp.db.name`	Y
Path to the extracted Oracle Autonomous Database wallet files	`atp.wallet.path`	Y
Oracle Autonomous Databaseuser name	`db.user`	Y
Oracle Autonomous Databaseuser password	`db.password`	Y
Oracle Autonomous Database wallet password. This is the password you provided when you ran the download script.	`db.wallet.password`	Y
WebLogic Server administrator user name	`wls.admin.user`	Y
WebLogic Server administrator password	`wls.admin.password`	Y

Property Description	Property Name	Required
t3 or t3s URL to the WebLogic Administration Server node	wls.admin.url	Y
Type of target on which to deploy the data source. Value: Cluster or Server	ds.target.type	Y
Name of the cluster or a comma separated list of server names	ds.target.names	Y
JNDI namespace Default is jdbc/<ds.name>	ds.jndi.name	N
Oracle Autonomous Database service for the data source connection. Value: low (default), medium, tp, or tpurgent	db.level	N
JDBC driver class Default is oracle.jdbc.OracleDriver	ds.jdbc.driver	N
Indicates if the Oracle Autonomous Database is on Dedicated Exadata Infrastructure. Value: true or false (default).	is.atp.dedicated	N

2. Open an SSH connection to any node as the opc user, then change to the oracle user.
3. Run the script create_atp_datasource.sh in one of the following ways:
   - Execute and follow user prompts:

     ```
     /opt/scripts/utils/create_atp_datasource.sh
     ```

     At each prompt, enter a value or press Enter to accept the default value. The following is an example of the script response after entering the values:

     ```
     INFO: Found wallet config file
     INFO: Verifying existing datasources.
     INFO: Verified that no existing data source has the same name.

     INFO: Created datasource configuration file /tmp/.ds_config
     INFO: Creating the datasource ==> datasource1
     INFO: Connecting to the admin server [t3://servicename-wls-0:7001]...
     INFO: Adding properties to datasource
     INFO: Target Type : Server
     INFO: Targets : servicename_server_1
     INFO: Setting targets [[com.bea:Name=servicename_server_1,Type=Server]]
     INFO: Successfully create datasource [datasource1]
     INFO: Validating the Datasource [datasource1]
     INFO: Verify datasource on Server AdminServer
     -- Datasource datasource1 not found on server AdminServer.
     INFO: Verify datasource on Server servicename_server_1
     -- datasource1:      State[Running] Connection Test is OK
     ```

- Execute using a properties configuration file:

  ```
  /opt/scripts/utils/create_atp_datasource.sh
  <path_to_configuration_file_and_file_name>
  ```

  For example:

  ```
  /opt/scripts/utils/create_atp_datasource.sh /u01/.ds_config
  ```

  The following is an example of the script response:

  ```
  INFO: Creating the datasource ==> datasource1
  INFO: Connecting to the admin server [t3://servicename-wls-0:7001]...
  INFO: Adding properties to datasource
  INFO: Target Type : Server
  INFO: Targets : servicename_server_1,servicename_server_2
  INFO: Setting targets [[com.bea:Name=servicename_server_1,Type=Server,
  com.bea:Name=servicename_server_2,Type=Server]]
  INFO: Successfully create datasource [datasource1]
  INFO: Validating the Datasource [datasource1]
  INFO: Verify datasource on Server servicename_server_2
  -- datasource1:  State[Running] Connection Test is OK
  INFO: Verify datasource on Server servicename_server_3
  -- Datasource datasource1 not found on server servicename_server_3.
  INFO: Verify datasource on Server servicename_server_1
  -- datasource1:  State[Running] Connection Test is OK
  INFO: Verify datasource on Server AdminServer
  -- Datasource datasource1 not found on server AdminServer.
  ```

Create a Data Source for a DB System Database

Use the WebLogic Server Administration Console to create a data source and establish a connection with an Oracle Cloud Infrastructure Database (DB System).

The DB System must allow your Oracle WebLogic Server for OCI domain to access the database listen port (1521 by default). Update the network security group that is assigned to the database, or update the security lists for the subnet on which the database was created, if necessary. See Security Rules for the DB System.

Tasks:

- Verify the PDB Name
- Configure a Data Source for a DB System Database

Verify the PDB Name

You need the PDB name if you have a 12c database in your DB System.

If you did not specify a PDB name when you created the 12c database, the default PDB name could be the database name appended with `_pdb1`. For example: `<db_name>_pdb1`

To verify you have the correct PDB name:

1. Open an SSH connection to the database node.

2. Use SQL*Plus to connect to the database as the `sys` user.
3. Execute the command:

   ```
   show pdbs
   ```

Configure a Data Source for a DB System Database

You use the WebLogic Server Administration Console to create a Java Database Connectivity (JDBC) data source for the DB System database.

1. Access the WebLogic Server Administration Console.
2. In the Change Center box, click **Lock & Edit**.
3. In the Domain Structure box, expand Services (by clicking the + next to it) and click **Data Sources**.
4. On the Summary of JDBC Data Sources page , click **New**, and then select **Generic Data Source**.
5. On the first page of the Create a New JDBC Data Source wizard, do the following:
   a. In the **Name** field, enter a name for your data source.
   b. In the **JNDI Name** field, enter a name.
   c. In the **Database Type** dropdown list, accept the default value **Oracle**.
6. On the second page of the wizard, in the **Database Driver** dropdown list, select the value that's appropriate for your domain version.

   For example, select ***Oracle's Driver (Thin) for Service connections; Versions:Any** for a 12c domain.
7. On the third page of the wizard, accept all options.
8. On the fourth page of the wizard, do the following:
   a. Enter the **Database Name** and **Host Name**. Use the appropriate format for your database version and type.

Database Version	Database Type	Database Name	Host Name
12c	VM	`<pdb_name>.<db_domain>`	`<db_hostname>-scan.<db_domain>`
12c	Bare Metal	`<pdb_name>.<db_domain>`	`<db_hostname>.<db_domain>`

   b. In the **Port** field, enter the database listen port number (`1521` by default).
   c. In the **Database User Name** field, enter a user who has been granted access to the database.
   d. In the **Password** field, enter the password for the database user and re-enter the password in **Confirm Password** field.
9. On the fifth page of the wizard, verify that the **URL** has the appropriate format based on your database version and type:

Database Verion	Database Type	URL
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>`

10. Click **Test Configuration** to verify if a connection to the database can be established based on the information that you provided.
    - If the connection test fails, click **Back** and review the entries that you made for the data source and correct any errors. For example, make sure there is a `/` after the port number. If there are no errors in the entries and the test still fails, make sure that your database is running.
    - The connection is successful when the message `Connection test succeeded` is displayed.
11. On the last page of the wizard, select the target for the JDBC data source and then click **Finish**.
12. In the Change Center, click **Activate Changes**.

Connect to a Domain Using Oracle JDeveloper

Before you can use Oracle JDeveloper to connect to an Oracle WebLogic Server for OCI domain, add an ingress rule to restrict Internet access to a new network channel, and then create a network channel that supports the T3S protocol to enable remote connections.

Note:

- Oracle recommends you use Oracle Cloud Infrastructure FastConnect and private IP addresses to set up the connection between Oracle JDeveloper and your Oracle WebLogic Server for OCI domain. See FastConnect Overview.
- Install Oracle JDeveloper on a separate compute instance in the same VCN as the WebLogic subnet. This allows access from the JDeveloper instance to the ports on the WLS subnet.

Topics:

- Add an Ingress Rule
- Create a Network Channel
- Create a Connection to the Domain Through Oracle JDeveloper

Add an Ingress Rule

Before you establish a network channel that supports the T3 (and secure T3) protocol, use the Oracle Cloud Infrastructure Console to create an ingress rule to restrict access from the Internet.

> **Note:**
>
> RMI communications in Oracle WebLogic Server use the T3/T3S protocol to transport data between WebLogic Server and other Java programs, including clients. When using a port that supports T3/T3S, Oracle recommends you limit access only to known clients.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.
4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`
5. Click the **Subnet** value.
6. Click the name of the default security list.
7. Click **Add Ingress Rules**.
8. In the **Source CIDR** field, do one of the following:
   - For a domain in a public subnet, you must limit access to the T3S port by using a restricted CIDR matching the IP range for the running JDeveloper instances.
   - For a domain in a private subnet, you can enter `0.0.0.0/0` or set up a security rule using the IP range for the running JDeveloper instances.
9. In the **Destination Port Range** field, enter `8002`.
10. Click **Add Ingress Rules**.

Create a Network Channel

When you use Oracle WebLogic Server for OCI to create a domain, the administration server is configured with network channels that do not support the T3 and T3S protocols. To connect to the administration server with tools that use the T3 and T3S protocols, you must create a new network channel.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. From the **Compartment** dropdown, select the compartment in which your domain is created.

4. Click the name of the domain instance that has the Administration Server node.

   The instance with the Administration Server node has `wls-0` appended to the name. For example: `abcde7xy-wls-0`

5. Copy the **Public IP Address** value.
6. Copy the **Private IP Address** value.
7. Copy the **Internal FQDN** value.
8. In a browser, access the WebLogic Server Administration Console using the public IP address.
9. In the Change Center box, click **Lock & Edit**.
10. In the Domain Structure box, expand **Environment** (by clicking the + next to it) and click **Servers**.
11. Click the name of the Administration Server node.
12. On the Settings page for the node, click **Protocols** and then click **Channels**.
13. Under Network Channels, click **New**.
14. On the first page of the Create a New Network Channel wizard, do the following:
    a. In the **Name** field, enter a name for the channel.
    b. In the **Protocol** dropdown list, select **t3s**.
15. On the second page of the Create a New Network Channel wizard, do the following:
    a. In the **Listen Address** field, enter the Internal FQDN value you copied.
    b. In the **Listen Port** field, enter `8002`.
    c. In the **External Listen Address** field, enter the Private IP Address value you copied.
    d. In the **External Listen Port** field, enter `8002`.
16. Click **Finish**.
17. In the Change Center, click **Activate Changes**.

Create a Connection to the Domain Through Oracle JDeveloper

After creating the ingress rule and network channel, establish and test a connection between Oracle JDeveloper and your Oracle WebLogic Server for OCI domain.

You'll need to provide the name of your Oracle WebLogic Server for OCI domain.

1. Open the Oracle JDeveloper Console.
2. Open the Create Application Server Connection dialog.
3. On the Configuration page, enter the following:
   a. For **WebLogic Hostname (Administration Server)**, enter the private IP address of the compute instance that runs the WebLogic Administration Server.
   b. For **Port**, enter `0`.
   c. For **SSL Port**, enter `8002`.
   d. Select the **Always Use SSL** check box.
   e. For **WebLogic Domain**, enter your WebLogic Server domain name in Oracle Cloud Infrastructure. The name has the suffix `_domain`. For example: `thestack_domain`

4. Ensure **No Proxy** is selected in your Web Browser and Proxy setting.
5. Test the application server connection in JDeveloper.

Configure SSL for WebLogic Server

You can update the domain in Oracle WebLogic Server for OCI to use a generated, self-signed certificate, or a certificate that has been issued by a Certifying Authority (CA).

> **Note:**
>
> When the Secured Production Mode Enabled option is selected SSL is configured using the OCI Certificate Service. This section should be skipped.

If your service instance does not include a load balancer, and you want to use a different SSL certificate for communication between clients and your Java applications, update the configuration for the Managed Servers in your domain.

After scaling out your service instance, you will also need to update the SSL configuration for the new server.

> **Note:**
>
> Back up your `<*domain path*>/config/config.xml` and `<*domain path*>/nodemanager/nodemanager.properties` files. If the SSL configuration fails, you will be able to restore the service instance to a known working state.

Tasks:

- Create Keystores and Certificates for WebLogic Server
- Add the Oracle Identity Cloud Service Certificate to the Trust Keystore
- Associate Keystores and SSL Certificate with WebLogic Server
- Configure Node Manager to Use the SSL Certificate (Important: To ensure a successful SSL handshake)
- Configure SSL for New Servers After Scaling Out
- Replace Expiring Certificates

Create Keystores and Certificates for WebLogic Server

Use keytool to create your own public/private key pairs and self-signed certificates. Optionally, create a Certificate Signing Request (CSR) for each generated certificate and submit it to a CA to obtain a trusted certificate.

1. Connect to the Administration Server node in your service instance with a secure shell (SSH) client, and then switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

2. Create a directory /u01/data/keystores to hold the keystore files.

   ```
   cd /u01/data
   mkdir keystores
   cd keystores
   ```

   > **Caution:**
   >
   > Do not place your keystore and certificate files in the Middleware Home (MIDDLEWARE_HOME) or Java Home (JAVA_HOME) directories. Any modifications you make to these locations might be lost when you apply a patch.

   > **Caution:**
   >
   > Do not place your keystore and certificate files in the Domain Home (DOMAIN_HOME) or /u01/data/domains directories because they are included in backups. A restore operation might include an expired certificate and result in errors during a server restart.

3. Use the keytool command to create a new identity keystore file, and to add a self-signed certificate to the keystore named server_cert.

   ```
   keytool -genkeypair -alias alias -keyalg keyalg -sigalg sigalg -keysize
   size -dname dn -keystore keystore_file
   ```

   For example:

   ```
   keytool -genkeypair -alias server_cert -keyalg RSA -sigalg SHA256withRSA -
   keysize 2048 -dname
   "CN=example.com,OU=Support,O=Example,L=Reading,ST=Berkshire,C=GB" -
   keystore identity.jks
   ```

   Note that The X.500 Distinguished Name, which consists of the WebLogic Server host and DNS domain name, is *example.com*.

4. When prompted, enter a password for the keystore.
5. When prompted, enter a password for the private key, server_cert, or press **Enter** to use the same password as the keystore.
6. If you are using a self-signed certificate to configure SSL, then create a custom trust keystore file.
   a. Use keytool to export the self-signed certificate, server_cert, from the identity store to a file named server_cert.cer.

      ```
      keytool -exportcert -alias server_cert -file server_cert.cer -keystore
      keystore_file
      ```

      When prompted, enter the password for the keystore.

   b. Use `keytool` to create a trust keystore file, and to import `server_cert.cer` into this new keystore. Use the same alias, `server_cert`.

      ```
      keytool -importcert -alias server_cert -file server_cert.cer -keystore
      trust_keystore_file
      ```

      For example:

      ```
      keytool -importcert -alias server_cert -file server_cert.cer -keystore
      trust.jks
      ```

   c. When prompted, enter a password for the new keystore.

   d. When prompted to trust this certificate, enter **yes**.

7. If you are using a CA-issued certificate to configure SSL, then create a CSR file from the identity keystore.

   a. Use `keytool` to create a CSR file for the `server_cert` private key.

      ```
      keytool -certreq -alias alias -file certreq_file -keystore keystore
      ```

      For example:

      ```
      keytool -certreq -alias server_cert -file server_cert.csr -keystore
      identity.jks
      ```

   b. When prompted, enter the password for the keystore and the private key.

   c. Submit the CSR to a Certificate Authority of your choice in order to obtain a trusted certificate.

   d. Import the CA-issued certificate into the identity keystore.

8. Copy the keystore files to all the other nodes in your service instance.

   For example:

   ```
   ssh myinstance-wls-2
   mkdir /u01/data/keystores
   scp myinstance-wls-1:/u01/data/keystores/identity.jks /u01/data/keystores
   scp myinstance-wls-1:/u01/data/keystores/trust.jks /u01/data/keystores
   ```

Add the Oracle Identity Cloud Service Certificate to the Trust Keystore

If your Oracle Java Cloud Service instance is configured to use Oracle Identity Cloud Service for authentication, you must add the Oracle Identity Cloud Service certificate to your custom trust keystore.

1. Access the Oracle Java Cloud Service console.
2. Click **Manage this service** ☰ for your service instance, and then select **Open Fusion Middleware Control Console**.
3. Click **WebLogic Domain**, select **Security**, and then select **Keystore**.
4. Expand the **system** folder.
5. Click **trust**, and then click **Manage**.

6. Click **idcs_root_ca**, and then click **Export**.
7. Click **Export Certificate**, and then click **Close**.
8. SSH to the Administration Server node and switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

9. Navigate to the `/u01/data/keystores` folder.
10. Create a new file named `idcs_root_ca.cer`. Paste the contents of the exported `idcs_root_ca` certificate into this file.
11. Use `keytool` to import `idcs_root_ca.cer` into your custom trust keystore.

    ```
    keytool -import -alias idcs_root_ca -file idcs_root_ca.cer -keystore
    trust_keystore_file
    ```

    For example:

    ```
    keytool -import -alias idcs_root_ca -file idcs_root_ca.cer -keystore
    trust.jks
    ```

12. When prompted, enter the password for the keystore.
13. When prompted to trust this certificate, enter **yes**.
14. Copy the updated trust keystore file to all the other nodes in your service instance.

    For example:

    ```
    ssh myinstance-wls-2
    scp myinstance-wls-1:/u01/data/keystores/trust.jks /u01/data/keystores
    ```

Associate Keystores and SSL Certificate with WebLogic Server

Use the WebLogic Server Administration Console to update the location of each server’s identity and trust keystore files, and the name of the certificate in the identity keystore that the server uses for SSL communication.

By default, the servers in an Oracle Java Cloud Service instance are configured to use a demo identity keystore and a demo trust keystore. Oracle recommends that you use these demo keystores for development purposes only.

1. Access the Oracle Java Cloud Service console.
2. Click the name of your service instance.
3. From the Overview page, identify the host names of all the nodes in your service instance, and the names of all servers in your domain.
4. Click **Manage this service** ☰, and select **Open WebLogic Server Administration Console**.
5. Log in to the console using the credentials that you specified when provisioning your service instance.
6. Within the **Change Center** panel, click **Lock and Edit**.
7. Within the **Domain Structure** panel, expand **Environment**, and then click **Servers**.

8. Click the name of the server for which you want to configure SSL.
9. Verify that the **Configuration** tab is selected. Under **Configuration**, click the **Keystores** tab.
   1. For **Keystores**, click **Change**. Select **Custom Identity and Custom Trust**, and then click **Save**.
   2. For **Custom Identity Keystore**, enter the full path to your identity keystore.

      For example, `/u01/data/keystores/identity.jks`
   3. For **Custom Identity Keystore Type**, enter `JKS`.
   4. For **Custom Identity Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Identity Keystore Passphrase**.
   5. For **Custom Trust Keystore**, enter the full path to your trust keystore.

      For example, `/u01/data/keystores/trust.jks`
   6. For **Custom Trust Keystore Type**, enter `JKS`.
   7. For **Custom Trust Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Trust Keystore Passphrase**.
   8. Click **Save**.
10. Under **Configuration**, click the **SSL** tab.
    1. For **Private Key Alias**, enter the name of the certificate (private key) in the identity keystore, `server_cert`.
    2. For **Private Key Passphrase**, enter the password for this certificate in the keystore. Enter the same value for **Confirm Private Key Passphrase**.

       By default, the password for the certificate is the same as the identity keystore password.
    3. Click **Save**.
11. Under **Change Center**, click **Activate Changes**.
12. Click the **Control** tab.
13. Click **Restart SSL**. When prompted for confirmation, click **Yes**.
14. Repeat from **step 6** to update each server in your domain for which you want to configure SSL.

    After you have configured SSL for the WebLogic Server to use the keystore `CustomIdentityAndCustomTrust`, go to the `boot.properties` file located in `DOMAIN_HOME/servers/AdminServer/security` and `DOMAIN_HOME/servers/<server_name>/data/nodemanager` and remove the line

    `TrustKeyStore=DemoTrust`.

For more information, refer to *Overview of Configuring SSL* in Administering Security for Oracle WebLogic Server (12.2.1).

Configure Node Manager to Use the SSL Certificate

To ensure a successful SSL handshake among the Administration Server, Managed Servers, and Node Manager, you should configure Node Manager to use the custom keystores and the SSL certificate.

1. Connect to the Administration Server node with a secure shell (SSH) client, and then switch to the `oracle` user.

   ```
   sudo su - oracle
   ```

2. Edit the `nodemanager.properties` file located in the Domain Home directory.

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

3. Add the following lines to the end of the file.

   ```
   KeyStores=CustomIdentityAndCustomTrust
   CustomIdentityKeystoreType=jks
   CustomIdentityKeyStoreFileName=path_to_identity_keystore
   CustomIdentityKeyStorePassPhrase=keystore_password
   CustomIdentityPrivateKeyPassPhrase=server_cert_password
   CustomIdentityAlias=server_cert
   CustomTrustKeyStoreType=jks
   CustomTrustKeyStoreFileName=path_to_trust_keystore
   CustomTrustKeyStorePassPhrase=keystore_password
   ```

   For example:

   ```
   KeyStores=CustomIdentityAndCustomTrust
   CustomIdentityKeystoreType=jks
   CustomIdentityKeyStoreFileName=/u01/data/keystores/identity.jks
   CustomIdentityKeyStorePassPhrase=keystore_password
   CustomIdentityPrivateKeyPassPhrase=server_cert_password
   CustomIdentityAlias=server_cert
   CustomTrustKeyStoreType=jks
   CustomTrustKeyStoreFileName=/u01/data/keystores/trust.jks
   CustomTrustKeyStorePassPhrase=keystore_password
   ```

4. Regenerate the Node Manager startup files.
   a. Launch the WebLogic Scripting Tool (WLST).

      ```
      $MIDDLEWARE_HOME/oracle_common/common/bin/wlst.sh
      ```

   b. Connect to the Administration Server.

      ```
      connect('admin_user','password','t3://admin_server_host:9071')
      ```

      For example:

      ```
      connect('weblogic','password','t3://myinstance-wls-1:9071')
      ```

   c. Generate the `boot.properties` and `startup.properties` files for the server(s) on this node.

      ```
      nmGenBootStartupProps('server_name')
      ```

Both the Administration Server and the first Managed Server run on the first node in the service instance. For example:

```
nmGenBootStartupProps('myinstance_adminserver')
```

```
nmGenBootStartupProps('myinstance_server_1')
```

d. Exit WLST.

```
exit()
```

5. Run the restart script as the `oracle` user.

```
/opt/scripts/restart_domain.sh
```

> **Note:**
>
> If your instance was created before 23.3.2 (end of August 2023) you should edit the `setEnv.sh` file located in `/opt/scripts`:
>
> ```
> vi /opt/scripts/setEnv.sh
> ```
>
> Add the following properties to the `WLST_PROPERTIES` variable set in the file:
>
> ```
> -Dweblogic.security.TrustKeyStore=CustomTrust
> -
> Dweblogic.security.CustomTrustKeyStoreFileName=path_to_trust_keysto
> re
> -Dweblogic.security.CustomTrustKeyStoreType=JKS
> ```
>
> For example, after adding the properties, `WLST_PROPERTIES` would be:
>
> ```
> export WLST_PROPERTIES="${WLST_PROPERTIES}
> -Dweblogic.security.SSL.minimumProtocolVersion=TLSv1.2 -
> Dpython.path=${PYTHONPATH}
> -Dweblogic.security.SSL.ignoreHostnameVerification=true -
> Dweblogic.security.TrustKeyStore=DemoTrust
> -Djava.security.egd=file:///dev/urandom -
> Doracle.jdbc.fanEnabled=false -Dweblogic.ssl.JSSEEnabled=true
> -Dweblogic.security.SSL.enableJSSE=true -
> Dweblogic.security.TrustKeyStore=CustomTrust
> -Dweblogic.security.CustomTrustKeyStoreFileName=/u01/data/
> keystores/trust.jks
> -Dweblogic.security.CustomTrustKeyStoreType=JKS"
> ```

6. Repeat from **Step 1** for any other nodes in service instance for which you want to configure SSL.

Configure SSL for New Servers After Scaling Out

After scaling out a cluster in the WebLogic Sever for OCI instance, you must modify the new server's SSL configuration if you want the server to use the custom keystores.

Use the WebLogic Server Administration Console to update the new server. See Associate Keystores and SSL Certificate with WebLogic Server in *Administering Oracle Java Cloud Service*.

WebLogic Sever for OCI automatically performs the following tasks during a scale-out operation:

- Copies the custom keystore files to the new node.
- Copies the Node Manager configuration files to the new node.

Replace Expiring Certificates

You should replace an expiring certificate before it expires to minimize or avoid application downtime.

You can set a notification to remind you before your certificate expires. For instructions, see Setting Certificate Expiry Notifications. To replace a certificate, see Replacing Expiring Certificates.

Upgrade a Domain

For an existing Oracle WebLogic Server for OCI domain, you can upgrade the WebLogic Server release from 12c (12.2.1.3) to WebLogic Server release 12c (12.2.1.4).

> **Note:**
>
> Upgrading from 14.1.1.0 to 14.1.2.0 is not covered in this document.

Following are the advantages of this documented process over the traditional in-place upgrade:

- The stack has the latest scripts on disk volumes.
- There is an easier roll back process. Since the original stack with 12.2.1.3 is still available, only the database has to be rolled back for a JRF instance.
- The WebLogic binaries are the latest. You do not have to locate the WebLogic installers and apply the required patches.

Topics:

- Backup the Database
- Create a 12.2.1.4 Instance
- Stop the WebLogic Server Processes on the Source 12.2.1.3 Instance
- Replace the Domain on the Target 12.2.1.4 Instance with Cloning
- Set up VNC Server

- Perform Readiness Check
- Upgrade Infrastructure Schemas
- Reconfigure the Domain
- Upgrade the Domain
- Restart Servers
- Post Upgrade
- Roll Back Upgrade

Backup the Database

1. Complete the *Create an on-demand full backup of a database* procedure in Backing Up a Container Database to Oracle Cloud Infrastructure Object Storage in the Oracle Cloud Infrastructure documentation.
2. Return here to continue with the next step.

Create a 12.2.1.4 Instance

1. Create a 12.2.1.4 instance, which is identical to the 12.2.1.3 instance that you plan to upgrade.

   Ensure that you provide a unique name to the 12.2.1.4 instance and use the same networking as the 12.2.1.3 instance. See Create a Stack.

   > **Note:**
   >
   > If you are using a reserved IP with your load balancer, then delete the load balancer on the source instance to free up the reserved IP address.

2. Return here to continue with the next step.

Stop the WebLogic Server Processes on the Source 12.2.1.3 Instance

1. Log in to the source instance as an opc user.
2. Run the following command:

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

   Run the following command to confirm if there are any running processes:

   ```
   jps
   ```

   If there are any processes running, then run kill -9 against each of the processes.
3. Log in to each of the non-administration compute instance as an opc user.

4. Run the following command:

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

   Run the following command to confirm if there are any running processes:

   ```
   jps
   ```

   If there are any processes running, then run `kill -9` against each of the processes.
5. Log in to administration compute instances as an `opc` user.
6. Run the following command:

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

   Run the following command to confirm if there are any running processes:

   ```
   jps
   ```

   If there are any processes running, then run `kill -9` against each of the processes.

Replace the Domain on the Target 12.2.1.4 Instance with Cloning

1. Clone the source 12.2.1.3 domain contents. See Clone a JRF or non-JRF Instance Using a Script.

   > **Note:**
   >
   > - Clone only the data block volumes. *Do not* use to the `-m` argument to clone the Middleware volumes.
   > - Cloning might fail at the `StartServers` step, as your applications may not be upgraded to handle 12.2.1.4 binaries. If this error occurs, ignore the error and continue with the next step.
   >   Run the following command:
   >
   >   ```
   >   python3 create_clone.py -p StartAppGateway
   >   ```

2. After you have run the cloning script on all the VMs, stop the WebLogic servers by running the following command on each VM:

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

Run the following command to confirm if there are any running processes:

```
jps
```

If there are any processes running, then run `kill -9` against each of the processes.

3. Are you upgrading a JRF instance?
   - Yes: Continue with the next procedure.
   - No: Go to Restart Servers.

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

MRC_NAME	COMP_ID	OWNER	VERSION	STATUS	UPGRADED
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 servers.
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 12.2.1.3 instance. See Destroy Stack Resources.

> **WARNING:**
>
> Do not Delete a JRF Database Schema or Delete the Identity Cloud Service Resources as these resources are required in the upgraded cloned 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 Upgrade

If the upgrade fails, you can roll back the upgrade. Since the source 12.2.1.3 instance is still present, only the database needs to be rolled back if you upgraded a JRF domain.

1. Complete the *Restore a database using a specific backup from Object Storage* procedure in Recovering a Container Database from Object Storage in the Oracle Cloud Infrastructure documentation.
2. If you use a reserved IP address on the load balancer, then you must delete the load balancer on the source instance prior to creating the 12.2.1.4 instance.
   1. Delete the load balancer or destroy the 12.2.1.4 stack to release the reserved IP address. See Destroy Stack Resources.
   2. Create a load balancer in Oracle Cloud Infrastructure again using the reserved IP address. See Configure SSL for a Domain.

4 Scale a Stack

Resize your Oracle WebLogic Server for OCI domain to meet changing workload requirements.

> **Note:**
>
> If you have moved your password to new secret or vault or updated the password, then you must update the password secret OCID and policy to read the secrets with new secrets OCID. See Update the Password Secret OCID, Policy, and User Name. If you have removed the WebLogic administrator password secret or the dynamic group policy for accessing the secret, see About Deleting Secrets and Policies.

Topics

- Add or Remove WebLogic Server Nodes
- Add UCM WebLogic Server Node to a BYOL Stack
- Manage Autoscaling Resources
- Change the Shape of the Existing Compute Instances
- Change Reserved Public IP Usage
- Add a Load Balancer
- Remove the Load Balancer

Add or Remove WebLogic Server Nodes

You can change the number of nodes (compute instances) in your Oracle WebLogic Server for OCI stack to increase performance or to reduce costs. Add nodes to scale out, or remove nodes to scale in.

> **Note:**
>
> You cannot use this procedure to scale a domain that was created before June 29, 2020.

To scale the domain, edit the node count variable for the stack and then run an Apply job.

Oracle WebLogic Server for OCI performs these tasks for a scaling operation:

- Add or remove compute instances
- Add or remove managed servers in the domain configuration (optional)
- Scale out or scale in the new compute instances (optional)
  During scale out, the middleware binaries and the JDK are synchronized to the newly added compute instance, and the domain is packed and unpacked from the administration

server. To know more about pack and unpack operations, see Overview of the Pack and Unpack Commands in *Creating Templates and Domains Using the Pack and Unpack Commands*.

- Add managed servers to the existing cluster in the domain configuration, if the domain is not running Oracle WebLogic Server Standard Edition (optional)
- Update the backend set of the load balancer, if your stack includes a load balancer (optional)
- Mount the configured file system to any new compute instances added, if your stack includes a file system (optional)

Topics:

- Considerations for Scaling Out
- Considerations for Scaling In

Considerations for Scaling Out

If you customized your domain configuration after creating it (changing port numbers, changing server names, and so on), there is no guarantee that the domain modification part of the scaling job will succeed. You can disable this feature and perform these tasks manually after the scaling job completes.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. Edit the **Node Count**, and then change the shape of the new compute instances or use the default shape.

   For the compute instance shape:

   - If the default shape is a flexible shape and you choose to use this default shape, you can edit the OCPU count for the additional instances.

     > **Note:**
     >
     > If you do not increase the node count and edit the OCPU count for any of the flexible shapes, the OCPU count remains unchanged.

   - If you select the standard shape for the new compute instance, you cannot specify the OCPU count for the instance.

   If you increase the node count and modify the **SSH Public Key**, the key will be used for new compute instances only. The keys for existing compute instances remain unchanged.
8. Optional: If you have updated the WebLogic Server administrator password, then enter the OCID of the secret that contains the password for the WebLogic Server administrator. See Create Secrets for Passwords.

9. Optional: If you want Oracle WebLogic Server for OCI to create compute instances but not update your domain configuration, select **Do Not Update Domain Configuration For Scale Out**.
10. Click **Next**.
11. Click **Save Changes**.
12. On the Stack Details page, click **Apply**.
13. In the Apply panel, enter a name for the job and click **Apply**.
14. Periodically monitor the progress of the Apply job until it is finished.
15. Click **Outputs**.
16. Locate `WebLogic_Instances`, and verify the number of compute instances in the updated stack.

If you selected **Do Not Update Domain Configuration For Scale Out**, then you must manually update your domain configuration and add the managed servers. Use the WebLogic Server Administration Console or the WebLogic Server Scripting Tool (WLST).

If your domain is in a private subnet, the bastion compute instance is deleted and recreated. As a result, the bastion might have a different IP address. See Access the WebLogic Console in a Private Subnet.

If you scale out a domain created after June 24th, 2021, any OPatches you added after provisioning are automatically applied to the new virtual machines.

For domains that were created before June 25th, 2021 or the domains on which the scaling operation did not successfully add a new WebLogic managed server, perform the binary synchronization step manually on the new added virtual machines using the following commands:

```
adminHost=$(python3 /opt/scripts/databag.py wls_admin_host)

# Step 1: rsync middleware
rsync -e "ssh -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -o
LogLevel=ERROR" -a --delete --timeout 60 /
--exclude=user_projects --exclude=logs oracle@$adminHost:/u01/app/oracle/
middleware/ /u01/app/oracle/middleware
# Step 2: rsync jdk
rsync -e "ssh -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -o
LogLevel=ERROR" -a --delete --timeout 60 /
--exclude=user_projects --exclude=logs oracle@$adminHost:/u01/app/
oracle/jdk/ /u01/app/oracle/jdk
```

Considerations for Scaling In

When you perform a scale in operation to reduce the number of nodes or compute instances running on your Oracle WebLogic Server for OCI stack, you should remove certain components such as managed servers and machines manually. You should delete the server first followed by the target machine to avoid encountering an error.

To remove such components manually:

1. Log in to the Oracle WebLogic Server Administration Console.

   For information to access the console, see Access the WebLogic Server Administration Console.

2. Click **Lock & Edit**.
3. Got to **Domain**, click **Environment**, and then select **Servers**.
4. Select the server that is on the compute instance you want to scale in.

> **Note:**
>
> Ensure that the server is in the shutdown state before you delete it.

5. Click **Delete**.
6. Click **Yes** to confirm the deletion.
7. To delete the target machine, click **Machines** on the left navigation pane.
8. Click **Delete**.
9. Click **Activate Changes**.

You are now ready to perform the scale in operation.
To scale in a compute instance:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. Reduce the **Node Count** to the desired value.
8. Click **Next**.
9. Click **Save Changes**.
10. On the Stack Details page, click **Apply**.
11. In the Apply panel, enter a name for the job and click **Apply**.
12. Periodically monitor the progress of the Apply job until it is finished.
13. Click **Outputs**.
14. Locate `WebLogic_Instances`, and verify the number of compute instances in the updated stack.

Add UCM WebLogic Server Node to a BYOL Stack

You can add Universal Credits (UCM) WebLogic server nodes to a Bring Your Own License (BYOL) stack in your Oracle WebLogic Server for OCI stack.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.

4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. From the **Image for Scale Out** dropdown, select the required WebLogic server image for scale out.

> **Note:**
>
> - If you have a BYOL stack you can add either BYOL or UCM nodes. If you have a UCM stack you can add only UCM nodes.
> - If you have a BYOL stack with UCM nodes, to add BYOL nodes you must scale in the UCM nodes before adding BYOL nodes.
> - If you have a BYOL stack that is enabled for autoscaling, you can scale out the stack with UCM nodes or BYOL nodes.
> - If you select a BYOL image, it requires a WebLogic License with a valid support contract. If you select a UCM image, it is charged per OCPU per hour for the entitlement and WebLogic support.

8. Select **Terms of use**.
9. Click **Next**.
10. Click **Save Changes**.
11. On the Stack Details page, click **Apply**.
12. In the Apply panel, enter a name for the job and click **Apply**.
13. Periodically monitor the progress of the Apply job until it is finished.

Manage Autoscaling Resources

If you created an Oracle WebLogic Server for OCI with autoscaling enabled, you can create and update the Alarm Definition, configure the parameters of the scaling functions, and reenable autoscaling for a stack.

If you have not enabled autoscaling for your WebLogic instance, see Configure Autoscaling.

Topics:

- Create Alarm Definitions
- Update Alarm Definitions
- Configure Function Application
- Reenable Autoscaling for a Stack

Create Alarm Definitions

Alarms are used to push messages to configured destinations. You can create custom alarms to receive notifications for WebLogic domain metrics, in addition to the Scale Out Alarm

Definition and Scale In Alarm Definition that are created when autoscaling is enabled during provisioning.

For more information on alarm definitions, see Autoscaling.

To create the alarm definitions:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Observability & Management**. Under **Monitoring**, click **Alarm Definitions**.
3. Click **Create Alarm**.
4. Under **Define alarm**, enter an **Alarm name** and select the **Alarm severity**.
5. Under **Metric description**, select the **Compartment** where the Application Performance Monitoring domain is located, select `oracle_apm_monitoring` **Metric namespace**, and then for **Metric name**, select a WebLogic metric.

   For monitoring concepts, metric namespaces, and the default Application Performance Monitoring metrics, see Monitoring Concepts and Application Performance Monitoring Metrics in Oracle Cloud Infrastructure documentation.
6. Under **Metric dimensions**, do the following:
   a. For **Dimension name**, select *AppserverClusterName* and for **Dimension value**, select the WebLogic cluster created by the stack.
   b. Select **Aggregate metric streams** to return the combined value of all metric streams for the selected statistic.
7. Under **Trigger rule**, specify the **Operator**, **Value**, and **Trigger delay minutes**.

   The default value for **Trigger delay minutes** of the alarm definition is five minutes, which is the number of minutes that the condition is maintained before the alarm state changes from "Ok" to "Firing". You can specify the number of minutes that the condition must be maintained before the alarm is in firing state.
8. Under **Destinations**, select the following:
   - For **Destination service**, select *Notification Service*.
   - For **Compartment**, select your stack compartment.
   - For **Topic**, select the notification topic for scale out or scale in.
9. Click **Save alarm**. The new alarm is listed on the Alarm Definitions page.

   See Create Alarms in the Oracle Cloud Infrastructure documentation.

Update Alarm Definitions

You can update alarm definitions if you want to change the number of alarm definitions and the settings such as threshold value and alarm metrics that are configured when autoscaling is enabled during provisioning.

To update the alarm definitions:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Observability & Management**. Under **Monitoring**, click **Alarm Definitions**.
3. Click the name of the alarm definition.
4. Click **Actions**, and click **Edit alarm**.

5. Under **Define alarm**, **Metric description**, and **Metric dimensions**, update the alarm settings as needed.
6. Under **Trigger rule**, update the default settings for **Value** and **Trigger delay minutes**.

> **Note:**
>
> The default value for **Trigger delay minutes** of the alarm definition is five minutes, which is the number of minutes that the condition is maintained before the alarm state changes from "Ok" to "Firing". You can specify the number of minutes that the condition must be maintained before the alarm is in firing state.

7. Under **Notifications**, update the default settings for **Notification frequency**.

> **Note:**
>
> You can enter the **Notification frequency** only if **Repeat notification?** is selected. The default period of time to wait before resending the notification is 20 minutes for scale out and 30 minutes for scale in.

See Update Alarm in the Oracle Cloud Infrastructure documentation.

Configure Function Application

In autoscaling, you can use functions to invoke the Resource Manager APIs to scale the WebLogic Server domain in Oracle WebLogic Server for OCI. If you want to edit the Function Application configuration that is set when autoscaling is enabled during provisioning, you can configure the parameters of the Function Application.

To know the function application that are created during autoscaling, see Autoscaling.

To configure the parameters of the function application:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Developer Services**. Under **Functions**, click **Applications**.
3. Select the **Compartment** that contains your stack.
4. Click the name of the application.
5. Under **Resources**, click **Configuration**.
6. Update the following parameters:
   - *offline_ms1_from_lb* - Set this parameter to *true,* if you want the managed server coresiding on the backend of the administration node to be set offline temporarily. This setting is required when the threshold condition for metric is breached and when the administration server is unable to configure the node on the cluster as the WebLogic Server domain is overloaded.
     After the managed server is set to offline, the managed server is suspended (server changes from "Running" state to "Admin" state), and it only accepts administrative requests. As a result, the administration server has CPU time to activate the domain configuration on the new node as no traffic (requests) is sent to managed server during scale out. After stack apply job is complete, the managed server backend is set to online (server changes from "Admin" to "Running" state).

> **Note:**
>
> By default, *offline_ms1_from_lb* is set to false.

- *debug* - Set this parameter to *true* to enable debug logging for functions.
- *min_wls_node_count* - Provide the minimum WebLogic node count for scale in so that the scaling happens up to the *min_wls_node_count* value.
  By default, *min_wls_node_count* is set to the number of WebLogic nodes specified during initial stack provisioning. For example, if you selected two nodes during initial stack provisioning, the *min_wls_node_count* is set to two.

  You must not edit the parameters, *stack_ocid* and *wlsc_email_notification_topic_id*.

At the function-level, you can add a parameter to update and override any inherited parameter value, the selected function has inherited from the Function Application configuration.
To override the parameter values of the function application:

1. Under **Resources**, click **Functions**.
2. Click the name of the function.
3. Under **Resources**, click **Configuration**.
4. If you want to disable autoscaling, set the **use_autoscaling** parameter to *None*.

Reenable Autoscaling for a Stack

If the scale in or scale out fails, you receive an email notification with the job status and information about the tag **disabled_for_autoscaling:true** enabled for the stack.

You can check the Scale Apply job logs in the Oracle Cloud Infrastructure console and the Function logs to identify the cause of the failure. After you fix the issue, reenable autoscaling by setting the tag **disabled_for_autoscaling** to false or removing the tag from the stack.

To reenable autoscaling for the stack:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Developer Services**. Under **Resource Manager**, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click the **Tags** tab and do one of the following:
   - Set **disabled_for_autoscaling** to *false*.
   - Click **Remove Tag** to remove the tag from the stack.

Change the Shape of the Existing Compute Instances

You can scale up the existing compute resources for your Oracle WebLogic Server for OCI domain to increase performance, or you can scale down the existing compute resources to reduce costs.

> **Note:**
>
> Do not use Resource Manager to change the shape of the compute instances in your domain. You must use the Compute service.

When you change the shape of an existing compute instance, you select a different processor, number of cores, amount of memory, network bandwidth, and maximum number of VNICs for the instance. The instance's public and private IP addresses, volume attachments, and VNIC attachments remain the same. For example, changing the shape of an instance from `VM-Standard2.2` to `VM-Standard2.4` doubles the capacity of the node from two OCPUs to four OCPUs, and also doubles the amount of memory allocated to the node.

The original shape of the compute instance determines which shapes you can select as a target for the new shape. You cannot modify the shape of an instance that uses a bare metal shape or certain virtual machine (VM) shapes. See Changing the Shape of an Instance.

Oracle recommends that you use the same shape for all compute instances that comprise a single WebLogic Server cluster. This allows traffic to be distributed uniformly across the cluster.

When you change a shape, the compute instance must be restarted. To avoid downtime and ensure your applications remain available to users, Oracle recommends that you create a cluster with multiple compute instances, and that you change the shape for one compute instance at a time.

When you change the shape of the first compute instance, the domain's administration server will be temporarily unavailable. However, your applications do not depend on the administration server and will not be affected.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Compute**. Under the **Compute** group, click **Instances**.
3. Select the **Compartment** in which your domain was created.
4. Click the name of the first compute instance for your domain.

   The instance has `wls-0` appended to the name. For example: `mydomain-wls-0`
5. Click **Edit**, and then select **Edit Shape**.
6. Select a new shape, and then click **Save Changes** > **Reboot Instance**.
7. Repeat from **step 2** for the remaining compute instances in your domain.

Change Reserved Public IP Usage

You can update an existing Oracle WebLogic Server for OCI domain to either use a bastion compute instance with a reserved IP or remove the assigned reserved IP.

Edit the reserved IP variable for the stack and then run an *Apply* job.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. Configure Reserved IP:
   - If your want to use a bastion compute instance with a reserved public IP, then select **Assign Reserved Public IP to Bastion Instance**
   - If you have already used a bastion compute instance with a reserved public IP and *do not* want to use a reserved public IP now, then unselect **Assign Reserved Public IP to Bastion Instance**.
8. Click **Next**.
9. Click **Save Changes**.
10. On the Stack Details page, click **Apply**.
11. In the Apply panel, enter a name for the job and click **Apply**.
12. Periodically monitor the progress of the Apply job until it is finished.
13. Click **Outputs**.
14. Identify if the reserver IP address of the load balancer in the updated stack.

Add a Load Balancer

You can add a load balancer to an existing Oracle WebLogic Server for OCI domain that was originally created without a load balancer.

> **Note:**
>
> You cannot use this procedure on a domain that was created before June 29, 2020. If you enabled autoscaling for your domain and did not configure the load balancer on stack creation, you must add the load balancer when you edit the stack.

Oracle WebLogic Server for OCI configures the new load balancer to distribute application traffic to the managed servers in your domain.

Edit the load balancer variables for the stack and then run an Apply job.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. Select **Provision Load Balancer**.
8. Configure the load balancer as follows:
    - If you chose to create a new VCN or use an existing VCN with new subnets:
        a. Select **Create New Load Balancer**.
        b. Select **Private Load Balancer**, if you do not want to assign a public IP address to the load balancer.
        This option is available only if you use a private subnet for WebLogic Server. You cannot create a private load balancer in a public subnet.
        c. Select **Load Balancer with Reserved Public IP**, if you want to use a public load balancer with a reserved public IP. Then, specify the OCID of the public IP for the load balancer.
        d. Select a minimum and maximum flexible load balancer shape. By default, the minimum bandwidth size is set to 10Mbps and maximum to 400Mbps.

        > **Note:**
        >
        > You can update the shape to a maximum of 8000Mbps. Before you select the maximum bandwidth, ensure to check the available service limit for the flexible load balancer bandwidth.

        e. Configure the load balancer network.
            – If you want to use an existing regional subnet for WebLogic Server, then select an existing regional subnet from the list of regional and availability domain-specific subnets. A load balancer can have only one regional subnet, which is shared between both nodes.
            – If you are creating a regional subnet for WebLogic Server, then specify a CIDR for the new load balancer subnet.
        f. If you are using existing network security groups (NSGs) for an existing subnet, specify the NSG that is assigned to the load balancer.
    - If you chose to use an existing VCN and an existing subnet:
        a. Select **Use Existing Load Balancer**.
        b. Specify the OCID for the existing load balancer.
        c. Enter the name of the backend set for the existing load balancer that has a routing policy associated with the backend set. The backend set should not have backends. See Configure the Load Balancer.
        d. If you are using existing network security groups (NSGs), specify the NSG that is assigned to the load balancer.

9. Click **Next**.
10. Click **Save Changes**.
11. On the Stack Details page, click **Apply**.
12. In the Apply panel, enter a name for the job and click **Apply**.
13. Periodically monitor the progress of the Apply job until it is finished.
14. Click **Outputs**.
15. Identify the IP address of the load balancer in the updated stack.

If your domain includes the sample application, you can access it using the load balancer. See Access the Sample Application.

Remove the Load Balancer

If you created an Oracle WebLogic Server for OCI domain with a load balancer, and no longer require the load balancer, you can remove it.

> **Note:**
>
> You cannot use this procedure on a domain that was created before June 29, 2020.

Edit the load balancer variables for the stack and then run an Apply job.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.
7. Clear **Provision Load Balancer**.
8. Click **Next**.
9. Click **Save Changes**.
10. On the Stack Details page, click **Apply**.
11. In the Apply panel, enter a name for the job and click **Apply**.
12. Periodically monitor the progress of the Apply job until it is finished.

5 Clone a Stack

You can clone an Oracle WebLogic Server for OCI instance by cloning the block volumes that contain the middleware binaries (`mw`) and the domain configuration (`data`).

> **Note:**
>
> To ensure that a cloned instance uses the appropriate image, do not clone the boot volume.

Following are the different methods to clone an instance:

- **Move the instance**
  In this method, the binaries and data are moved from one instance to another. Here, the same database, load balancer, and IDCS application (if IDCS is enabled), is used in the source and the cloned instance.

  Example: You want a newer version of the OS and cannot get the current instance's kernel updated. In this use case, you would want to move binaries and data from one instance at a lower kernel version to another instance, where the kernel is newer.

  > **Note:**
  >
  > In the JRF case, when following the manual cloning method, where infra database is used, to ensure that the cloned instance can connect to the database, it is recommended to:
  >
  > – Verify that the cloned instance security rules allow access to the existing infra database.
  >
  > – Have the Compartment and VCN of the cloned instance be the same as the database.

- **Copy the instance**
  In this method, you will copy the binaries and data from one instance to another. Here, load balancer or IDCS application (if IDCS is enabled), is different for the source and instance.

  Example: You have a test instance and a production instance. After you have completed testing updates in the test instance you want to copy the contents to the production instance.

  > **Note:**
  >
  > Currently, in this method, cloning for an instance with JRF database is not supported.

Topics:

- Move a WebLogic Server Instance
- Copy a WebLogic Server Instance

Move a WebLogic Server Instance

Use the clone script to clone and move a WebLogic server instance, where the script performs various procedural steps to clone your instance. You can also manually clone an instance by following the steps provided for a JRF or non-JRF instance.

For a non-JRF instance, you can clone to move the binaries and data, and then reuse the load balancer and IDCS of the original instance. For a JRF instance, you can clone and check if it is able to connect to the database through the data source, as no database-specific configuration updates are required.

> **Note:**
>
> For instances created after 22.3.2 (August 2022), if Application Performance Monitoring (APM) is not configured on the `Original instance`, but Application Performance Monitoring is configured on the `Cloning instance`, then the clone script adds Application Performance Monitoring to the cloned stack.

Clone a JRF or non-JRF Instance Using a Script

Learn how to clone an instance by using the clone script.

In this procedure,

- `Original instance`, is the WebLogic Server instance you want to clone.
- `Cloning instance`, is the WebLogic Server instance you will create to clone the `Original instance`.

Methods: There are two methods to clone an instance.

- Method 1
- Method 2

Method 1: Do not destroy the `Original instance` stack and run the script on the `Cloning instance` to clone the required volumes from the `Original instance`.

1. Create a domain. This is the `Cloning instance`. See Create a Stack.

> **Note:**
>
> - You must create the `Cloning instance` using the same configuration as the `Original instance`, in the same compartment and region.
> - It is recommended to stop the original instance servers before running the cloning script. This is specifically recommended for JRF instances, as having 2 WebLogic domains using the same infrastructure schemas is not supported by Oracle WebLogic Server.
> - If your instance uses an existing VCN with a new subnet, then the CIDR range for the subnet created in the `Cloning instance` differs from the CIDR range of the subnet in the `Original instance`.

2. In the `Original instance`, delete any load balancer using a reserved IP. See Remove the Load Balancer.
3. In the `Original instance`, if you have assigned an SSL certificate to the original load balancer, then set up SSL certificates on the load balancer in the `Cloning instance`. See Add a Certificate to the Load Balancer.
4. Stop all the servers in the `Original instance`. See Start and Stop a Domain.
5. Log in to each node of the `Cloning instance` as an `opc` user.
6. If you are using the `CustomIdentityAndCustomTrust` keystore for SSL configuration in the WebLogic Server domain, and you have not placed the keystore in a location under the data volume (`/u01/data`), then, as the `oracle` user, you will need to copy the keystore to the compute instance (VM) before you invoke the `create_clone.py` script in the next step.
7. Run the `create_clone.py` script on each of the nodes. The `create_clone.py` script is located at `/opt/scripts/cloning`:
   - If you want to clone only the data block volumes:

     ```
     python3 create_clone.py -s <Original_instance_stack_OCID>
     ```

   - If you want to clone both the data block and Middleware volumes:

     ```
     python3 create_clone.py -s <Original_instance_stack_OCID> -m true
     ```

   > **Note:**
   >
   > Do not run the script on `Cloning instance` nodes that correspond to nodes added on the `Original instance` with the **Do Not Update Domain Configuration for Scale Out** selected. The metadata for these nodes will be missing, resulting in failures at various steps in the script. On such nodes you must extend the domain in the same manner as you did on the `Original instance`.

   If the clone script fails at a particular stage, complete the steps in Clone Script Failed, and then return to this procedure to continue with the next step.

8. Access the WebLogic console of the `Cloning instance` to verify the updates you performed to the instance. See Access the WebLogic Server Administration Console.
9. Destroy and Delete the `Original instance`. See Delete a Stack.

Method 2: Manually create cloned volumes, destroy the `Original instance` stack, and run the script on the `Cloning instance` to attach the cloned volumes.

> **Note:**
>
> This method is recommended when you are using VCN peering or running low on limits, such as, compute or load balancer.

1. Clone the data block volumes of the `Original instance`. See Cloning a Volume. As required, you can also clone the Middleware volumes.
2. Destroy and then delete the stack in `Original instance`. See Destroy Stack Resources and Delete the Stack.
3. Create a domain. This is the `Cloning instance`. See Create a Stack.

   > **Note:**
   >
   > You must create the `Cloning instance` using the same configuration as the `Original instance`, in the same compartment and region.

4. Log in to each node of the `Cloning instance` as an `opc` user.
5. If you are using the `CustomIdentityAndCustomTrust` keystore for SSL configuration in the WebLogic Server domain, and you have not placed the keystore in a location under the data volume (`/u01/data`), then, as the `oracle` user, you will need to copy the keystore to the compute instance (VM) before you invoke the `create_clone.py` script in the next step.
6. Run the `create_clone.py` script on each of the nodes. The `create_clone.py` script is located at `/opt/scripts/cloning`:
   - If you have cloned only the data block volumes:

     ```
     python3 create_clone.py -d <Original_instance_data_volume_OCID>
     ```

   - If you have cloned both the data block and Middleware volumes:

     ```
     python3 create_clone.py -d <Original_instance_data_volume_OCID> -m
     <Original_instance_Middleware_volume_OCID>'
     ```

> **Note:**
>
> Do not run the script on `Cloning instance` nodes that correspond to nodes added on the `Original instance` with the **Do Not Update Domain Configuration for Scale Out** selected. The metadata for these nodes will be missing, resulting in failures at various steps in the script. On such nodes you must extend the domain in the same manner as you did on the `Original instance`.

If the clone script fails at a particular stage, complete the steps in Clone Script Failed, and then return to this procedure to continue with the next step.

7. Access the WebLogic console of the `Cloning instance` to verify the updates you performed to the instance. See Access the WebLogic Server Administration Console.

You have successfully cloned the instance.

If you run terraform apply after cloning, complete the following steps:

1. If you add nodes, it restores any volumes that was previously destroyed by the cloning script.
   The restored volumes can be identified if they have the following name format:
   - `<clone_prefix>-data-block-<number>`
   - `<clone_prefix>-mw-block-<number>`

   Detach and delete the restored volumes. See Deleting a Volume.
2. If OCI logging is added then OCI logging has to be run independently on the compute instance running the administration server.
   Run the following commands on the administration server:

   ```
   python3 update_metadata.py -k use_oci_logging -v true
   python3 update_logging.py
   ```

Copy a WebLogic Server Instance

For a non-JRF instance, you can clone to copy the binaries and data from the original instance, and then use the load balancer and IDCS in the cloned instance. In this setup, the cloned instance would have the same topology as the original instance.

> **Note:**
>
> Currently, in this method, cloning for an instance with JRF database is not supported.

Topics:

- Clone a non-JRF Instance
- Scale Out the Cloned Instance

Clone a non-JRF Instance

Complete the following steps:

In this procedure:

- `Original instance` is the instance you want to clone.
- `Cloning instance` is the instance you will create to clone the original instance.

1. Create a non-JRF domain. This is the `Cloning instance`. See Create a Basic Domain.

   > **Note:**
   >
   > If the source instance uses IDCS, the IDCS configuration steps assume that the new instance was created with the **Enable Authentication Using Identity Cloud Service** option selected.

2. Complete the following steps if you have created an instance by using the Identity Cloud Service:

   a. Open the `/u01/data/domains/idcss_domain/config/config.xml` file and make a note of the `idcs:client-id` value.

   b. Take a backup of the following to the `/tmp` location:

   ```
   /u01/data/cloudgate_config
   ```

3. Detach the block volumes that were created in the cloning instance.

   a. Stop the Domain. See Start and Stop a Domain.

   b. Remove the mount volume:
   ```
   sudo umount <mount>
   ```
   Where *`<mount>`* is `/u01/data` or `/u01/app`.

   c. From the navigation menu, click **Compute**. Under the **Compute** group, click **Instances**.

   d. From the **Compartment** drop-down list, select the compartment in which your instance is created.

   e. Click the instance you created.

   f. In the instance page, under Resources, click **Attached Block Volume**.

   g. Against the block volume that was created when creating the domain, click the menu icon and then click **Detach**.
   The `Detach Block Volume` dialog box is displayed. It provides information about your volume and the iSCSI commands you will need. The commands are ready to use with the appropriate information included.

   Copy these commands.

   h. Click **Continue Detachment**.

   i. Access the node and then run the iSCSI commands that you copied in step 3e.

   j. Repeat step 3e through step 3g for all the other block volumes that were created when creating the domain.

4. Clone both the middleware and data block volumes of the `Original instance`. See Cloning a Volume.

   > **Note:**
   >
   > Follow the next steps, to first attach the middleware volume and then attach the data block volume to the `Cloning instance`. This sequence ensures that the `mountVolume.sh` script works as desired, which you will be running later in this procedure.

5. To the `Cloning instance`, attach the middleware volume that your cloned in step 4.
   a. In the instance page, under Resources, click **Attached Block Volume** > **Attach Block Volume**.
   b. In the **Attach Block Volume** page, select the compartment where you cloned the block volume earlier.
   c. Click the drop-down and select the block volume you clone earlier.
   d. Select the Attachment Type as **ISCSI**.
   e. Click **Attach**.
      An `Attach Block Volume` message appears.
   f. Click **Close**.
6. To the `Cloning instance`, attach the data block volume that your cloned in step 4.
   a. In the instance page, under Resources, click **Attached Block Volume** > **Attach Block Volume**.
   b. In the **Attach Block Volume** page, select the compartment where you cloned the block volume earlier.
   c. Click the drop-down and select the block volume you clone earlier.
   d. Select the Attachment Type as **ISCSI**.
   e. Click **Attach**.
      An `Attach Block Volume` message appears.
   f. Click **Close**.
7. Run the following script:

   ```
   /opt/scripts/cloning/mountVolume.sh -m <cloned-middleware-volume-name> -d
   <cloned-data-volume-name>
   ```

   This script runs the iSCSI commands to attach both the middleware and data volumes. Also, the script updates the UUID entries of both the volumes in `/etc/fstab`, which ensures that the mount is persistent across reboot.
8. Update the metadata on each node by using `update_metadata.py` script, which is located at `/opt/scripts/utils/`:

We need to update the metadata for reboot to work correctly. This also starts the node manager and administration server automatically after every restart. There are 5 domain related values in metadata that need to be updated for reboot to work.

```
python3 /tmp/update_metadata.py -k wls_domain_name -v
<resource_prefix>_domain
python3 /tmp/update_metadata.py -k wls_machine_name -v
<resource_prefix>_machine_
python3 /tmp/update_metadata.py -k wls_cluster_name -v
<resource_prefix>_cluster
python3 /tmp/update_metadata.py -k wls_admin_server_name -v
<resource_prefix>_adminserver
python3 /tmp/update_metadata.py -k wls_ms_server_name -v
<resource_prefix>_server_
```

The script updates one value at time. Also, for every command, it creates the backup of previous setup under `/opt/scripts/utils/metadata_backup_<timestamp>.txt`.

> **Note:**
>
> After the restart works as desired, you can delete these backed up files.

9. Run the following `update_hostname.sh` script, which is located at `/opt/scripts/cloning`:

   ```
   ./update_hostname.sh <FQDN-of-source-instance>
   ```

   For example: `./update_hostname.sh source12c-wls-0.subnet1fd5ed7.idcsvcn.oraclevcn.com`

   This updates the hostname to the cloned hostname in the `nodemanager.properties` and `config.xml` files. It also updates the `startup.properties` file to reflect the correct admin url.

10. If the Demo Identity certificates have the incorrect host name, then, as the `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
    ```

11. If you are using the `CustomIdentityAndCustomTrust` keystore for SSL configuration in the WebLogic Server domain, and you have not placed the keystore in a location under the data volume (`/u01/data`), then, as the `oracle` user, you will need to copy the keystore to the compute instance (VM).

12. Reboot the instance.

    This will automatically first start the node manager and then start the administration server.

13. Repeat step 2 through step 10 on all the nodes.

    You have now cloned the WebLogic Server Instance. Continue with the next steps to configure the load balancer and IDCS.

> **Caution:**
>
> If you are not using IDCS, then do not continue with the next steps.

14. Access the IDCS console and copy the `idcs:client-secret` value:
    1. Access the IDCS console.
    2. From the left navigation, Click **Applications**.
    3. Search for the confidential application that is used by the Authentication Provider for the instance. It will have the instance name and the word `confidential` in it.
    4. Select the application, and then click **Configuration**.
    5. Under **General Information**, click **Show Secret** against **Client Secret**.
       The Client Secret information is displayed.
    6. Copy the **Client Secret** value.
15. In the WebLogic console, update the `idcs:client-id` and `idcs:client-secret`.
    1. Access the WebLogic console of the clone instance. See Access the WebLogic Server Administration Console.
    2. Under **Domain Structure**, click **Security Realms**.
    3. In the **Summary of Security Realms** page, select **myrealm** > **Providers**.
    4. Click **IDCSIntegrator** > **Provider Specific**.
    5. In the left navigation, under **Change Center**, click **Lock & Edit**.
    6. In the **Provider Specific** page, enter the values for `Client Id`, `Client Secret`, and `Confirm Client Secret`.
       Where,
       - `Client Id` : is the client ID that you made a note of in step 2a.
       - `Client Secret` or `Confirm Client Secret`: is the Client Secret that you copied in step 12.
    7. Click **Save**.
    8. In the left navigation, under **Change Center**, click **Release Configuration.**
16. In the cloned instance, restore the backup of `/u01/data/cloudgate_config` from the `/tmp` to `/u01/data` location.

    This is the backup which you performed in step 2b.
17. Restart the container.

    ```
    sudo systemctl status appgateway.service
    sudo systemctl start appgateway.service
    sudo podman ps -a
    sudo systemctl status appgateway.service
    sudo /opt/scripts/idcs/run_cloudgate.sh
    ```
18. Verify if `nginx` is redirecting traffic to WebLogic server through port `9999`:

    ```
    ip=$(hostname -i)
    cloudgate_url=${ip}:9999
    ```

```
curl -v ${cloudgate_url}
curl -s -o /dev/null -w "%{http_code}\n" ${cloudgate_url}
```

19. Repeat step 14 through step 16 on all nodes.

> **Note:**
>
> Here you will use the load balancer of the cloned instance. So, there is no update required to the load balancer.

20. Verify if you can access sample app. See Access the Sample Application.
21. After every reboot, manually start the `appgateway`:

```
sudo systemctl start appgateway
sudo podman ps -a
ip=$(hostname -i)
cloudgate_url=${ip}:9999
curl -v ${cloudgate_url}
curl -v ${cloudgate_url}/sample-app
```

You have successfully cloned the instance.

Access the respective WebLogic console to verify the updates you performed to the instance. See Access the WebLogic Server Administration Console.

After you verified the updates you performed to the instance, delete the block volumes that you detach in step 6. See Deleting a Volume.

Scale Out the Cloned Instance

You can scale out the instance that you cloned.

> **Note:**
>
> If you have updated the WebLogic Server password, then create a version of the Secret and update the metadata scripts to use the new Secret. See *To update a secret's contents to create a new secret version* under Managing Secrets in the Oracle Cloud Infrastructure documentation.

To scale out the cloned instance, edit the node count variable for the stack and complete the required steps:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the **Compartment** that contains your stack.
4. Click the name of your stack.
5. Click **Variables**.
6. Click **Edit Variables**.

7. Edit **WebLogic Server Node Count** to the increase the number of compute instances.
8. Select **Do Not Update Domain Configuration for Scale Out**.
9. Click **Next**.
10. Click **Save Changes**.

    The apply job is run to update the stack.
11. Periodically monitor the progress of the Apply job until it is finished.
12. Click **Outputs**.
13. Locate `WebLogic_Instances`, and verify the number of compute instances in the updated stack.
14. Update the metadata on each node by using `update_metadata.py` script, which is located at `/opt/scripts/utils/`:

    We need to update the metadata for reboot to work correctly. This also starts the node manager and administrator server automatically after every restart. There are 5 domain related values in metadata that need to be updated for reboot to work.

    ```
    python3 /tmp/update_metadata.py -k wls_domain_name -v
    <resource_prefix>_domain
    python3 /tmp/update_metadata.py -k wls_machine_name -v
    <resource_prefix>_machine_
    python3 /tmp/update_metadata.py -k wls_cluster_name -v
    <resource_prefix>_cluster
    python3 /tmp/update_metadata.py -k wls_admin_server_name -v
    <resource_prefix>_adminserver
    python3 /tmp/update_metadata.py -k wls_ms_server_name -v
    <resource_prefix>_server_
    ```

    The script updates one value at time. Also, for every command, it creates the backup of previous setup under `/opt/scripts/utils/metadata_backup_<timestamp>.txt`.

    > **Note:**
    >
    > After the restart works as desired, you can delete these backed up files.
15. Run the manual scale out by running extend domain scripts on the added instance.

    > **Note:**
    >
    > You must change `WLST_PROPERTIES` for a custom SSL set up, to avoid SSL handshake errors.
    >
    > For example, if you are using Custom Identity and Custom Trust, then `weblogic.security.TrustKeyStore=CustomTrust`, `weblogic.security.CustomTrustKeyStoreFileName`, and `weblogic.security.CustomTrustKeyStoreType` need to be set.

a. Run the following command:

```
export WLST_PROPERTIES="-
Dweblogic.security.SSL.minimumProtocolVersion=TLSv1.2 -
Dweblogic.security.SSL.ignoreHostnameVerification=true
-Dweblogic.security.TrustKeyStore=DemoTrust -
Djava.security.egd=file:///dev/urandom -Dweblogic.ssl.JSSEEnabled=true
-Dweblogic.security.SSL.enableJSSE=true -Dwlst.offline.log=/u01/logs/
wlst_extend_domain.log"
```

b. Run the following extend domain script for WebLogic Server:

```
/opt/scripts/decryptStrings.sh 1 | /u01/app/oracle/middleware/
oracle_common/common/bin/wlst.sh -skipWLSModuleScanning /opt/scripts/
extend_12c_domain.py
```

Access the respective WebLogic console to verify the updates you performed to the instances. See Access the WebLogic Server Administration Console.

6 Delete a Stack

Use Resource Manager to destroy and delete the stack when you no longer need an Oracle WebLogic Server for OCI domain.

You perform two separate actions:

- Destroy the stack – A destroy job terminates the compute instance or instances for the domain but the stack's state and job history remain.
- Delete the stack – A delete job permanently removes the stack and all related resources that were created for the domain, such as compute instances, networking components, and load balancer components.

If your domain includes the Java Required Files (JRF) components, then you must also delete the JRF schema before you destroy the stack.

If your domain uses Oracle Identity Cloud Service, then you must also delete the security resources before you destroy the stack.

Tasks:

- Delete a JRF Database Schema
- Delete the Identity Cloud Service Resources
- Delete Autoscaling Resources
- Destroy Stack Resources
- Delete the Stack
- Delete the Database Security List
- Delete the Database Network Security Group
- Delete the Identity Cloud Service Resources Manually

Delete a JRF Database Schema

If the Oracle WebLogic Server for OCI domain you want to delete was created with the Java Required Files (JRF) components, you must remove the JRF schema before you destroy the stack.

> **WARNING:**
>
> Skip this procedure for an instance that was cloned, as database schemas are required in the cloned instance.

You'll need the following to delete the JRF schema:

- The secure shell (SSH) private key that corresponds to the public key that was specified when you created the domain

- The public IP address to the Administration Server node. If the WebLogic domain is in a private subnet, look up the bastion's public IP address and the private IP address of the administration server node.
- The WebLogic Server administrator password
- The SYSDBA user password of the database associated with the domain

To delete the schema associated with a JRF-enabled domain:

1. From your computer, run the `ssh` command to connect to the domain's Administration Server node as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_IP_address
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i 
   path_to_private_key opc@bastion_public_IP" opc@node_private_IP
   ```

   For example:

   ```
   ssh -i /home/myuser/mykey.openssh opc@203.0.113.13
   ```

   ```
   ssh -i ~/.ssh/mykey.openssh -o ProxyCommand="ssh -W %h:%p -i ~/.ssh/
   mykey.openssh opc@198.51.100.1" opc@192.0.2.254
   ```

2. If prompted, enter the passphrase for the private key.
3. Change to the `oracle` user.

   ```
   sudo su - oracle
   ```

4. Run the following command to delete the JRF schemas, providing the passwords for the WebLogic Server administrator and SYSDBA user.

   ```
   /opt/scripts/delete_rcu.sh domain_password database_password
   ```

   For example:

   ```
   /opt/scripts/delete_rcu.sh wlsadminpassword dbadminpassword
   ```

5. Wait for the script to run. The operation is completed when you see output similar to the following:

```
Component schemas dropped:
Component                                    Status       Logfile
Common Infrastructure Services               Success      /tmp/RCU2019-06-10_numstring/
logs/stb.log
Oracle Platform Security Services            Success      /tmp/RCU2019-06-10_numstring/
logs/opss.log
User Messaging Service                       Success      /tmp/RCU2019-06-10_numstring/
logs/ucsums.log
Audit Services                               Success      /tmp/RCU2019-06-10_numstring/
logs/iau.log
```

```
Audit Services Append                         Success      /tmp/RCU2019-06-10_numstring/
logs/iau_append.log
Audit Services Viewer                         Success      /tmp/RCU2019-06-10_numstring/
logs/iau_viewer.log
Metadata Services                             Success      /tmp/RCU2019-06-10_numstring/
logs/mds.log
WebLogic Services                             Success      /tmp/RCU2019-06-10_numstring/
logs/wls.log
Repository Creation Utility - Drop : Operation Completed
>
<Jun 11, 2019 05:15:12 PM GMT> <INFO> <cleanup.py> <(host:resourcename-
wls-0.mysubnet.ocidbvcnterrafo.oraclevcn.com) - Successfully deleted rcu schemas for
prefix = SP1560222222>
```

6. If necessary, review the entire output for exceptions or failures that caused a failed or incomplete deletion. For example, a failure to connect to the domain could be caused by an invalid WebLogic Server administrator password. Re-execute the script after fixing the issues.

Delete the Identity Cloud Service Resources

If the Oracle WebLogic Server for OCI domain you want to delete was configured to use Oracle Identity Cloud Service for authentication, then you must delete the security resources for the domain before you destroy the stack.

> **WARNING:**
>
> Skip this procedure for an instance that was cloned, as Oracle Identity Cloud Service resources are required in the cloned instance.

You'll need the client ID and secret of an existing confidential application in Oracle Identity Cloud Service. See Create a Confidential Application.

1. From your computer, run the `ssh` command to connect to the domain's Administration Server node as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_IP_address
   ```

   For example:

   ```
   ssh -i /home/myuser/mykey opc@203.0.113.13
   ```

2. If prompted, enter the passphrase for the private key.
3. Run the following command to delete the security resources for this domain.

   Provide the client ID and secret of the confidential application in Oracle Identity Cloud Service.

   ```
   sudo su oracle -c '/opt/scripts/idcs/delete_idcs_applications.sh
   idcs_app_client_id idcs_app_client_secret'
   ```

Sample output:

```
Deactivating App Gateway gateway_name...
Deleting App Gateway gateway_name...
Deactivating application enterprise_app_name...
Deleting application enterprise_app_name...
Deactivating application confidential_app_name...
Deleting application confidential_app_name...
```

Delete Autoscaling Resources

The autoscaling resources, Functions, Event Rule, and Notification Subscriptions are created using Oracle Cloud Infrastructure SDK APIs from WebLogic Administration instance during provisioning. So, you must destroy these autoscaling resources before destroying the stack.

Run the command in Cloud Shell to destroy autoscaling resources using `remove_resources.py` script. To create `remove_resources.py`, see Script File to Delete Resources .

> **Note:**
>
> A user who is not an administrator can also run the `pre-destroy` command.

```
python3 remove_resources.py pre-destroy <service_name_prefix> -f autoscaling
```

If you run this script from a nonhome region, use the following command:

```
python3 remove_resources.py pre-destroy <service_name_prefix> -f autoscaling -
r <region_name>
```

Example:

```
python3 remove_resources.py pre-destroy abcstack -f autoscaling -r us-
phoenix-1
```

Destroy Stack Resources

To delete an Oracle WebLogic Server for OCI domain, use Resource Manager to destroy the stack associated with the domain before you execute the Delete Stack action.

> **Note:**
>
> If your domain includes the Java Required Files (JRF) components, be sure to delete the JRF schema before you destroy the stack.
> If autoscaling is enabled, you must destroy the autoscaling resources before you destroy the stack. See Delete Autoscaling Resources.

A destroy action terminates the compute instance or instances for the domain but the stack's state and job history remain until you execute the Delete Stack action.

To destroy a stack:

1. Sign in to the Oracle Cloud Infrastructure Console.
2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. From the **Compartment** dropdown, select the compartment where your stack is located.
4. Click the name of your stack.
5. Click **Terraform Actions**, then click **Destroy**.
6. When prompted for confirmation, click **Destroy**.

   A job with type `Destroy` and state `Accepted` is added to the top of the table under **Jobs**. After a few minutes, the state changes to `In Progress`.
7. Wait for the destroy job state to change to `Succeeded` before you delete the stack.

To verify the stack has been destroyed, navigate to the Compute Instances page. Instances associated with a destroyed stack are labeled `Terminated`.
If there are networking resources created by the stack and those resources are still in use by other compute instances (not created by the stack):

- The destroy action on the stack fails because related networking resources are still in use. Those networking resources will not be deleted.
- The compute instances created by the stack are terminated. You can proceed to delete the stack.

Delete the Stack

To delete the Oracle WebLogic Server for OCI stack, use Resource Manager to delete the stack associated with the stack.

> **NOT_SUPPORTED:**
>
> - Deleting a stack does not destroy the stack resources. Ensure that you first Destroy Stack Resources and then delete the stack.
> - To verify a stack has been destroyed, navigate to the Compute Instances page. Instances associated with a destroyed stack are labeled `Terminated`.

A delete action permanently removes the stack and all related resources that were created for the domain, such as compute instances, network components, and load balancer components. If any of the network resources are being used by other stacks, the delete action:

- Does not remove those network components
- Does remove the compute instances created by the stack for the domain

To delete a stack:

1. Sign in to the Oracle Cloud Infrastructure Console.

2. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. From the **Compartment** dropdown, select the compartment where your stack is located.
4. Click the name of your stack.
5. Click **Delete Stack**.

   > **Note:**
   >
   > Ensure that you first Destroy Stack Resources and then delete the stack.

6. When prompted for confirmation, click **Delete**.

The Stacks page redisplays immediately. You'll no longer see your stack listed on the page.

Delete the Database Security List

If your Oracle WebLogic Server for OCI domain is JRF-enabled and is connected to an Oracle Cloud Infrastructure Database (DB System), then you can delete the security list that grants the domain access to the database.

> **WARNING:**
>
> Do not the delete this security list if other domains are in the same VCN and are using the same database.

This security list is not a component of the stack for your domain, and is not automatically deleted when you destroy the stack.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, and then select **Virtual Cloud Networks**.
3. Select the **Compartment** where your database's virtual cloud network (VCN) is located.
4. Click the name of the database's VCN.
5. Click **Security Lists**.
6. Click the security list for your domain, `*servicename*-wls-to-db-seclist`.

   `*servicename*` is the resource name prefix you provided during stack creation.
7. Click **Terminate**.
8. When prompted for confirmation, click **Delete**.

Delete the Database Network Security Group

If your Oracle WebLogic Server for OCI domain is JRF-enabled and is connected to an Autonomous Database, then you can delete the security rules for the network security group (NSG).

> **WARNING:**
>
> Do not the delete the NSG security rules if other domains are in the same VCN and are using the same database.

1. Sign in to the Oracle Cloud Infrastructure Console.
2. From the navigation menu, click **Networking**, and then select **Virtual Cloud Networks**.
3. Select the **Compartment** where your database's virtual cloud network (VCN) is located.
4. Click the name of the database's VCN.
5. Click **Network Security Groups**.
6. Click the network security group for your domain.
7. Under **Security Rules**, select the security rule that has the description like *Allow ingress traffic from WLS subnet on DB port*.
8. Click **Remove**.

   If you to want to remove multiple security lists, then select the required check boxes, and then click **Remove**.
9. When prompted for confirmation, click **Remove**.

Delete the Identity Cloud Service Resources Manually

If the Oracle WebLogic Server for OCI domain you deleted was configured to use Oracle Identity Cloud Service for authentication, you can manually delete the security resources for the domain.

If the domain's compute instances still exist, you can delete the security resources using a script. See Delete the Identity Cloud Service Resources.

1. Delete the App Gateway that's associated with your domain.

   From Identity Cloud Service console, expand the navigation drawer, click **Security**, and then click **App Gateways**.

   The name of the gateway is `*servicename*_app_gateway_*timestamp*`. For example, `mywls_app_gateway_2019-08-01T01:02:01.123456`.
2. Delete the enterprise application that's associated with your domain.

   From Identity Cloud Service console, expand the navigation drawer, and then click **Applications**.

   The name of the application is `*servicename*_enterprise_idcs_app_*timestamp*`. For example, `mywls_enterprise_idcs_app_2019-08-01T01:02:01.123456`.
3. Delete the confidential application that's associated with your domain.

The name of the application is `servicename_confidential_idcs_app_timestamp`. For example, `mywls_confidential_idcs_app_2019-08-01T01:02:01.123456`.

7

Troubleshoot

Identify common problems in Oracle WebLogic Server for OCI and learn how to diagnose and solve them.

Topics

- Check Known Issues
- Missing uuid for /u01/data or /u01/app on Reboot
- Clone Script Failed
- Cleanup Resources of a Deleted Instance
- Stack Creation Failed
- Error in Mounting the Volume During Provisioning
- Unable to Access the WebLogic Console From the Internet
- Unable to Access the Sample Application Using the Load Balancer
- Unable to Access the Fusion Middleware Control Console or the Enterprise Manager Console From the Internet
- Load Balancer Does Not Send Cookie `X-Oracle-BMC-LBS-Route`
- Reapply Fails for Load Balancer Configuration
- Autoscaling Failed to Create Functions
- Management Agents Are Not Deleted on Instance Termination
- Enterprise Manager Console Is Not Loading
- Scale Out Fails on the Administration Compute Instance
- #unique_241
- Warnings for Schema Password
- Security Checkup Tool Warnings
- Running `python3` Command Fails
- Unrecognized Arguments When Using the Patching Utility Tool
- Get Additional Help and Contact Support
- Unable to Fetch the Password Expiry Date for the OPSS User

Check Known Issues

Learn about known problems in Oracle WebLogic Server for OCI and how to work around them.

See Known Issues in Oracle WebLogic Server for Oracle Cloud Infrastructure.

Delete RCU Fails for Autonomous Database in 14.1.2.0 Domains

Use the OCI Console for Autonomous Database to delete JRF schemas.

Issue: Delete RCU fails for Autonomous Database in 14.1.2.0 domains.

Workaround: Delete the JRF schemas from the OCI console for Autonomous Database.

Clone Script Failed

Issue: When you run the clone script it might fail at a particular stage.

Workaround: By using the error message, you can identify the stage where the error occurred, fix the error, and then run the following command to continue the cloning script from the required stage.

```
python3 /opt/scripts/cloning/create_clone.py -p <stage_name>
```

AD Mismatch:

In an Availability Domain (AD), if the threshold of set limits are reached, then when you create the `Cloning instance` the Compute instances might be placed in another Availability Domain that does not match the Availability Domain of the `Original instance`. Due to this mismatch, the cloned volumes cannot be attached.

Workaround: Use the Oracle WebLogic Server for OCI console to create backup of data volumes, and optionally, the Middleware volumes. Make a note of the OCIDs of backed up volumes and then complete the steps in Method 2: Manually create cloned volumes and destroy the source stack.

Cleanup Resources of a Deleted Instance

If you create an Oracle WebLogic Server for OCI instance and delete some resources outside of terraform, the terraform destroy may fail.

Issue:

You might have deleted an instance without destroying the instance. In this scenario, some of the resources you had created for the instance are not deleted.

Workaround:

In such scenarios, you can run the following script to remove all the resources of the instance.

1. Copy the following script in Cloud Shell.
   For example, copy the script and save the file as `remove_resources.py`.

   ```
   """
   #
   # Copyright (c) 2021, Oracle Corporation and/or its affiliates.
   # Licensed under the Universal Permissive License v 1.0 as shown at
   https://oss.oracle.com/licenses/upl.
   """

   import os
   ```

```
import sys
import oci

"""
Lists and deletes the resources like instances, policies, volumes, VCN
related resources, logs and tags etc..
"""


class CleanUpResources:

    def __init__(self):
        # delegate token should be present at /etc/oci/delegation_token in
cloud shell
        if os.path.exists('/etc/oci/delegation_token'):
            with open('/etc/oci/delegation_token', 'r') as file:
                delegation_token = file.read()
            self.signer =
oci.auth.signers.InstancePrincipalsDelegationTokenSigner(delegation_token=d
elegation_token)
        else:
            print("ERROR: In the Cloud shell the delegation token does not
exist at location /etc/oci/delegation_token."
                  "Run the script from the Cloud shell, where you need to
delete the resources.")
            sys.exit(1)
        self.vcn_client = oci.core.VirtualNetworkClient(config={},
signer=self.signer)
        self.virtual_network_composite_operations =
oci.core.VirtualNetworkClientCompositeOperations(self.vcn_client)
        self.log_client = oci.logging.LoggingManagementClient(config={},
signer=self.signer)
        self.log_composite_operations =
oci.logging.LoggingManagementClientCompositeOperations(self.log_client)
        self.identity_client = oci.identity.IdentityClient(config={},
signer=self.signer)
        self.identity_client_composite_operations =
oci.identity.IdentityClientCompositeOperations(self.identity_client)

    # Lists all the resources based on the service name prefix
    def list_all_resources(self, service_name_prefix):
        search_client =
oci.resource_search.ResourceSearchClient(config={}, signer=self.signer)
        running_resources = ["RUNNING", "Running", "AVAILABLE", "STOPPED",
"Stopped", "ACTIVE", "CREATED", "INACTIVE"]
        resource_not_required = ["PrivateIp", "Vnic"]
        structured_search =
oci.resource_search.models.StructuredSearchDetails(
            query="query all resources where displayname =~
'{}'".format(service_name_prefix),
            type='Structured',

matching_context_type=oci.resource_search.models.SearchDetails.MATCHING_CON
TEXT_TYPE_NONE)

        resources = search_client.search_resources(structured_search)
        resources_details = []
        no_of_resources = 0
        tagname_resource = "wlsoci-" + service_name_prefix
        default_rt = "Default Route Table for " + service_name_prefix
        print(
            "Resource Name                              Resource 
Type                        Resource Lifecycle State                  
OCID         DOC")
        print(

"==========================================================================
=======================================================================")
        for resource in resources.data.items:
            resource_name = resource.display_name
            if (resource_name.startswith(service_name_prefix) or 
tagname_resource in resource_name or default_rt in resource_name) and (
                    resource.lifecycle_state in running_resources) and (
                    resource.resource_type not in resource_not_required):
                resources_details.append(resource)
                no_of_resources = no_of_resources + 1
                print("{}            {}          {}          {}                 
{}".format(resource.display_name,

           resource.resource_type,

           resource.lifecycle_state,

           resource.identifier,

           resource.time_created))
        print(

"==========================================================================
======================================================================")
        print("Total number of resources 
{}".format(len(resources_details)))
        return resources_details

    # Removes all resources based on the service name prefix
    def cleanup_resources(self, delete_list):
        print("Deleting the resources")
        self.delete_policies(delete_list)
        self.delete_instance(delete_list)
        self.delete_block_volumes(delete_list)
        self.delete_load_balancer(delete_list)
        self.delete_subnet(delete_list)
        self.delete_sec_list(delete_list)
        self.delete_route_table(delete_list)
        self.delete_dhcp_options(delete_list)
        self.delete_internet_gateway(delete_list)
        self.delete_service_gateway(delete_list)
        self.delete_local_peering_gateway(delete_list)
        self.delete_nat_gateway(delete_list)
        self.delete_vcn_resources(delete_list)
        self.delete_unified_agent_configuration(delete_list)
        self.delete_log(delete_list)
        self.delete_log_group(delete_list)
        self.delete_mount_targets(delete_list)
        self.delete_fss(delete_list)
        self.delete_tag_namespace(delete_list)
        self.delete_boot_volumes(delete_list)

    # Delete Policies
    def delete_policies(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Policy":
                policy_ocid = resource.identifier
                print("Deleting policy: {0}, with ocid:
{1}".format(resource.display_name, policy_ocid))
                try:

self.identity_client_composite_operations.delete_policy_and_wait_for_state(
                        policy_ocid,

wait_for_states=[oci.identity.models.Policy.LIFECYCLE_STATE_DELETED])
                    print("Deleted policy successfully!")
                except Exception as e:
                    print("Error while deleting the policy {0}, policy id
{1}, Error message {2}".format(
                        resource.display_name, policy_ocid, str(e)))

    # Delete Dynamic Group
    def delete_dynamic_group(self, service_name_prefix):
        tenancy = os.environ['OCI_TENANCY']
        dynamic_group_list =
self.identity_client.list_dynamic_groups(tenancy).data
        for d_group in dynamic_group_list:
            if service_name_prefix in d_group.name:
                print("Deleting the dynamic group: {0}, with ocid:
{1}".format(d_group.name, d_group.id))
                try:

self.identity_client_composite_operations.delete_dynamic_group_and_wait_for
_state(
                        d_group.id,
wait_for_states=[oci.identity.models.DynamicGroup.LIFECYCLE_STATE_DELETED])
                    print("Deleted the dynamic group successfully!")
                except Exception as e:
                    print("Error while deleting the dynamic group name {},
ocid {}, Error message {}".format(
                        d_group.name, d_group.id, str(e)))

    # Delete Block Volumes
    def delete_block_volumes(self, delete_list):
        bv_client = oci.core.BlockstorageClient(config={},
signer=self.signer)
        bv_composite_operations =
oci.core.BlockstorageClientCompositeOperations(bv_client)
        for resource in delete_list:
            if resource.resource_type == "Volume":
                bv_ocid = resource.identifier
                try:
                    print(
                        "Deleting the block volume: {0}, with ocid
{1}".format(resource.display_name, bv_ocid))

bv_composite_operations.delete_volume_and_wait_for_state(
                        bv_ocid,
wait_for_states=[oci.core.models.Volume.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the block volume successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the block volume {0}, ocid
{1}, Error message {2}".format(
                            resource.display_name, bv_ocid, str(e)))

    # Delete all compute instances
    def delete_instance(self, delete_list):
        compute_client = oci.core.ComputeClient(config={},
signer=self.signer)
        compute_composite_operations =
oci.core.ComputeClientCompositeOperations(compute_client)
        for resource in delete_list:
            if resource.resource_type == "Instance":
                instance_ocid = resource.identifier
                instance_name = resource.display_name
                print("Deleting the compute instance: {0}, with ocid
{1}".format(instance_name, instance_ocid))
                try:

compute_composite_operations.terminate_instance_and_wait_for_state(
                        instance_ocid,
wait_for_states=[oci.core.models.Instance.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the compute instance successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the instance {0}, ocid {1},
Error message {2}".format(
                            instance_name, instance_ocid, str(e)))

    # Delete all Subnets in the VCN
    def delete_subnet(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Subnet":
                subnet_ocid = resource.identifier
                print(
                    "Deleting subnet: {0}, with ocid
{1}".format(resource.display_name, resource.identifier))
                try:

self.virtual_network_composite_operations.delete_subnet_and_wait_for_state(
                        subnet_ocid,

wait_for_states=[oci.core.models.Subnet.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted subnet successfully!")
                except Exception as e:
                    print("Error while deleting the subnet {0}, ocid {1},
Error message {2}".format(resource.display_name,
                   subnet_ocid, str(e)))

    # Delete Security lists
    def delete_sec_list(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "SecurityList":
                sec_list_name = resource.display_name
                sec_list_ocid = resource.identifier
                if not ("Default" in sec_list_name):
                    print(
                        "Deleting the security list: {0}, with ocid
{1}".format(resource.display_name,

    resource.identifier))
                    try:

self.virtual_network_composite_operations.delete_security_list_and_wait_for
_state(
                            sec_list_ocid,

wait_for_states=[oci.core.models.SecurityList.LIFECYCLE_STATE_TERMINATED])
                        print("Deleted the security list successfully!")
                    except Exception as e:
                        print(
                            "Error while deleting the security list {0},
ocid {1}, Error message {2}".format(
                                resource.display_name, sec_list_ocid,
str(e)))

    # Delete Load balancers
    def delete_load_balancer(self, delete_list):
        lb_client = oci.load_balancer.LoadBalancerClient(config={},
signer=self.signer)
        lb_composite_operations =
oci.load_balancer.LoadBalancerClientCompositeOperations(lb_client)
        for resource in delete_list:
            if resource.resource_type == "LoadBalancer":
                lb_name = resource.display_name
                lb_ocid = resource.identifier
                print("Deleting Load balancer {0} with ocid
{1}".format(lb_name, lb_ocid))
                try:

lb_composite_operations.delete_load_balancer_and_wait_for_state(
                        lb_ocid,

wait_for_states=[oci.load_balancer.models.WorkRequest.LIFECYCLE_STATE_SUCCE
EDED])
                    print("Load balancer deleted successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the loadbalancer {0}, ocid
{1}, Error message {2}".format(
                            lb_name, lb_ocid, str(e)))
    # Delete Route tables
    def delete_route_table(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "RouteTable":
                route_table_name = resource.display_name
                route_table_ocid = resource.identifier
                # Removing the route rules from the tables
                rt_details = oci.core.models.UpdateRouteTableDetails()
                rt_details.route_rules = []

self.virtual_network_composite_operations.update_route_table_and_wait_for_s
tate(
                    route_table_ocid, rt_details,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_AVAILABLE])
                self.vcn_client.update_route_table(route_table_ocid,
rt_details)
                # Default route table can't be deleted from VCN
                if not ("Default" in route_table_name):
                    print(
                        "Deleting the route table: {0}, with ocid
{1}".format(resource.display_name,

  resource.identifier))
                    try:

self.virtual_network_composite_operations.delete_route_table_and_wait_for_s
tate(
                            route_table_ocid,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_TERMINATED])

                        print("Deleted the route table successfully!")
                    except Exception as e:
                        print("Error while deleting the route table {0},
ocid {1}, Error message {2}".format(
                            resource.display_name, route_table_ocid,
str(e)))
                        if "associated with Subnet" in str(e):
                            try:

self.delete_subnet_route_table_association(route_table_ocid)
                                # After removing the association again
retrying the removal of route table
                                # This is for Db subnet route table

self.virtual_network_composite_operations.delete_route_table_and_wait_for_s
tate(
                                    route_table_ocid,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_TERMINATED])
                                print("Deleted the route table
successfully!")
                            except Exception as e:
                                print("Error while deleting the route
table after removing the association "
```

```
                                      "{0}, ocid {1}, Error message 
{2}".format
                                      (resource.display_name, 
route_table_ocid, str(e)))

    # Delete Subnet and route table association to remove route table
    def delete_subnet_route_table_association(self, route_table_ocid):
        default_rt_id_in_vcn = ""
        print("Route table is associated with a subnet. Removing the 
association between the subnet and route table")
        rt_res = self.vcn_client.get_route_table(route_table_ocid).data
        vcn_id = rt_res.vcn_id
        compartment_id = rt_res.compartment_id
        list_route_rables_vcn = 
self.vcn_client.list_route_tables(compartment_id=compartment_id,

vcn_id=vcn_id).data
        for rt in list_route_rables_vcn:
            if "Default Route" in rt.display_name:
                default_rt_id_in_vcn = rt.id
        list_subnets = 
self.vcn_client.list_subnets(compartment_id=compartment_id, 
vcn_id=vcn_id).data
        for subnet in list_subnets:
            subnet_ocid = subnet.id
            if subnet.route_table_id == route_table_ocid:
                subnet_details = oci.core.models.UpdateSubnetDetails()
                subnet_details.route_table_id = default_rt_id_in_vcn
                try:

self.virtual_network_composite_operations.update_subnet_and_wait_for_state(
                        subnet_ocid, subnet_details,

wait_for_states=[oci.core.models.Subnet.LIFECYCLE_STATE_AVAILABLE])
                    print("Removed the association between the subnet and 
route table.")
                except Exception as e:
                    print("Error while removing the association between 
the subnet and route table {}".format(str(e)))

    # Delete DHCP Options
    def delete_dhcp_options(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "DHCPOptions":
                dhcp_name = resource.display_name
                dhcp_ocid = resource.identifier
                if not ("Default" in dhcp_name):
                    print(
                        "Deleting the DHCP options: {0}, with ocid 
{1}".format(resource.display_name, dhcp_ocid))
                    try:

self.virtual_network_composite_operations.delete_dhcp_options_and_wait_for_
state(dhcp_ocid,

                              wait_for_states=[
oci.core.models.DhcpOptions.LIFECYCLE_STATE_TERMINATED])
                        print("Deleted the DHCP options successfully!")
                    except Exception as e:
                        print(
                            "Error while deleting the DHCP options {0},
ocid {1}, Error message {2} ".format(
                                resource.display_name, dhcp_ocid, str(e)))

    # Delete Internet Gateway
    def delete_internet_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "InternetGateway":
                ig_ocid = resource.identifier
                print("Deleting the Internet Gateway: {0}, with ocid
{1}".format(resource.display_name,

     ig_ocid))
                try:

self.virtual_network_composite_operations.delete_internet_gateway_and_wait_
for_state(ig_ocid,

                              wait_for_states=[


oci.core.models.InternetGateway.LIFECYCLE_STATE_TERMINATED])

                    print("Deleted the Internet Gateway successfully!")
                except Exception as e:
                    print("Error while deleting the Internet Gateway {0},
ocid {1}, Error message {2}".format(
                        resource.display_name,
                        ig_ocid, str(e)))

    # Delete Service Gateway
    def delete_service_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "ServiceGateway":
                svc_gateway_ocid = resource.identifier
                print("Deleting the service gateway: {0}, with ocid
{1}".format(resource.display_name,

    svc_gateway_ocid))
                try:

self.virtual_network_composite_operations.delete_service_gateway_and_wait_f
or_state(
                        svc_gateway_ocid,
wait_for_states=[oci.core.models.ServiceGateway.LIFECYCLE_STATE_TERMINATED]
)

                    print("Deleted the service gateway successfully!")
                except Exception as e:
                    print("Error while deleting the service gateway {0},
ocid {1}, Error message {2}".format(
                        resource.display_name,
                        svc_gateway_ocid, str(e)))

    # Delete Local Peering Gateway
    def delete_local_peering_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "LocalPeeringGateway":
                lpg_ocid = resource.identifier
                print("Deleting the local peering gateway: {0}, with ocid
{1}".format(resource.display_name,

          lpg_ocid))
                try:

self.virtual_network_composite_operations.delete_local_peering_gateway_and_
wait_for_state(
                        lpg_ocid,
wait_for_states=[oci.core.models.LocalPeeringGateway.LIFECYCLE_STATE_TERMIN
ATED])

                    print("Deleted local peering gateway successfully!")
                except Exception as e:
                    print("Error while deleting the local peering gateway
{0}, ocid {1}, Error message {2}".format(
                        resource.display_name, lpg_ocid, str(e)))

    # Delete Nat Gateway
    def delete_nat_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "NatGateway":
                nat_ocid = resource.identifier
                print("Deleting the NAT gateway: {0}, with ocid
{1}".format(resource.display_name,

nat_ocid))
                try:

self.virtual_network_composite_operations.delete_nat_gateway_and_wait_for_s
tate(
                        nat_gateway_id=nat_ocid,

wait_for_states=[oci.core.models.NatGateway.LIFECYCLE_STATE_TERMINATED]
                    )
                    print("Deleted the NAT gateway successfully!")
                except Exception as e:
                    print("Error while deleting the NAT gateway {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, nat_ocid, str(e)))

    # Delete VCN
    def delete_vcn_resources(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Vcn":
                vcn_ocid = resource.identifier
                vcn_name = resource.display_name
                print("Deleting the VCN: {0}, with ocid
{1}".format(vcn_name, vcn_ocid))
                try:

self.virtual_network_composite_operations.delete_vcn_and_wait_for_state(vcn
_ocid,

                 oci.core.models.Vcn.LIFECYCLE_STATE_TERMINATED)
                    print("Deleted the VCN successfully!")
                except Exception as e:
                    print("Error while deleting the VCN {0}, VCN id {1},
Error message {2}".format(vcn_name, vcn_ocid,

                       str(e)))

    # Deleting the Unified Agent Configuration
    def delete_unified_agent_configuration(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "UnifiedAgentConfiguration":
                uac_ocid = resource.identifier
                print("Deleting the unified agent configuration: {}, with
ocid {}".format(resource.display_name, uac_ocid))
                try:

self.log_composite_operations.delete_unified_agent_configuration_and_wait_f
or_state(
                        uac_ocid,

wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])
                    print("Deleted the unified agent configuration
successfully!")
                except Exception as e:
                    print("Error while deleting the unified agent
configuration name {0}, ocid {1} - Error message {2}".format(
                        resource.display_name, uac_ocid, str(e)))

    # Delete logs in a Log groups
    def delete_log(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "LogGroup":
                log_group_ocid = resource.identifier
                list_logs = self.log_client.list_logs(log_group_ocid).data
                for log in list_logs:
                    print("Deleting the log name {0}, with log ocid
{1}".format(log.display_name, log.id))
                    try:

self.log_composite_operations.delete_log_and_wait_for_state(
                            log_group_ocid, log.id,
wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])
                        print("Deleted the log successfully!")
                    except Exception as e:
                        print("Error while deleting the log name {}, log
ocid {}, Error message {}".format(
                            log.display_name, log.id, str(e)))
    # Delete Log Group
    def delete_log_group(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "LogGroup":
                log_group_ocid = resource.identifier
                print("Deleting the log group: {0}, with ocid
{1}".format(resource.display_name,

log_group_ocid))
                try:

self.log_composite_operations.delete_log_group_and_wait_for_state(
                        log_group_ocid,
wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])

                    print("Deleted log group successfully!")
                except Exception as e:
                    print("Error while deleting the log group {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, log_group_ocid, str(e)))

    # Delete the Mount targets
    def delete_mount_targets(self, delete_list):
        mt_client = oci.file_storage.FileStorageClient(config={},
signer=self.signer)
        mt_composite_operations =
oci.file_storage.FileStorageClientCompositeOperations(mt_client)
        for resource in delete_list:
            if resource.resource_type == "MountTarget":
                mt_ocid = resource.identifier
                print("Deleting the mount target {0}, with ocid
{1}".format(resource.display_name, mt_ocid))
                try:

mt_composite_operations.delete_mount_target_and_wait_for_state(
                        mt_ocid,
wait_for_states=[oci.file_storage.models.MountTarget.LIFECYCLE_STATE_DELETE
D])
                    print("Deleted the mount target successfully!")
                except Exception as e:
                    print("Error while deleting the mount target {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, mt_ocid, str(e)))

    # Delete FSS
    def delete_fss(self, delete_list):
        fss_client = oci.file_storage.FileStorageClient(config={},
signer=self.signer)
        fss_composite_operations =
oci.file_storage.FileStorageClientCompositeOperations(fss_client)
        for resource in delete_list:
            if resource.resource_type == "FileSystem":
                fss_ocid = resource.identifier
                try:
                    # Get the list of exports to delete
                    list_exports =
fss_client.list_exports(file_system_id=fss_ocid).data
                    for export in list_exports:
                        export_ocid = export.id
                        print("Deleting the export id
{}".format(export_ocid))

fss_composite_operations.delete_export_and_wait_for_state(
                            export_id=export_ocid,

wait_for_states=[oci.file_storage.models.Export.LIFECYCLE_STATE_DELETED])
                        print("Deleted the exports successfully!")
                except Exception as e:
                    print("Error while deleting the export, Error message
{}".format(str(e)))
                try:
                    print("Deleting the FSS: {0}, with ocid
{1}".format(resource.display_name, fss_ocid))

fss_composite_operations.delete_file_system_and_wait_for_state(
                        fss_ocid,
wait_for_states=[oci.file_storage.models.FileSystem.LIFECYCLE_STATE_DELETED
])
                    print("Deleted the FSS successfully!")
                except Exception as e:
                    print("Error while deleting the FSS name {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, fss_ocid, str(e)))

    # Deletion of TagNamespace
    def delete_tag_namespace(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "TagNamespace":
                tag_ns_name = resource.display_name
                tag_ns_ocid = resource.identifier
                print("Deleting the tag namespace {0}, with ocid
{1}".format(tag_ns_name, tag_ns_ocid))
                try:
                    # Retiring the tag namespace
                    tag_status =
self.identity_client.get_tag_namespace(tag_namespace_id=tag_ns_ocid).data
                    print("Tag namespace: {} and isRetired:
{}".format(tag_ns_name, tag_status.is_retired))

                    if not tag_status.is_retired:
                        print("Retiring the tag namespace
{}".format(tag_ns_name))
                        tag_ns_details =
oci.identity.models.UpdateTagNamespaceDetails()
                        tag_ns_details.is_retired = True

self.identity_client_composite_operations.update_tag_namespace_and_wait_for
_state(
                            tag_namespace_id=tag_ns_ocid,
                            update_tag_namespace_details=tag_ns_details,
                            wait_for_states=[
oci.identity.models.TagNamespace.LIFECYCLE_STATE_INACTIVE])
                        tag_status = 
self.identity_client.get_tag_namespace(tag_namespace_id=tag_ns_ocid).data
                        print("Tag status before deleting 
{}".format(tag_status.is_retired))
                    print("Deleting the tag namespace 
{}".format(tag_ns_name))
                    # Tag namespace deletion is taking too long time. So 
not waiting for the completion.

self.identity_client.cascade_delete_tag_namespace(tag_namespace_id=tag_ns_o
cid)
                    print("Asynchronous deletion of Tag namespaces is 
enabled."
                          "Check the deletion status manually. Tag name 
{0} with ocid {1}".format(tag_ns_name,

                      tag_ns_ocid))
                except Exception as e:
                    print("Error while deleting the Tag namespace {0}, 
ocid {1}, Error message {2} "
                          .format(tag_ns_name, tag_ns_ocid, str(e)))

    # Deleting the unattached boot volumes
    def delete_boot_volumes(self, delete_list):
        bv_client = oci.core.BlockstorageClient(config={}, 
signer=self.signer)
        bv_composite_operations = 
oci.core.BlockstorageClientCompositeOperations(bv_client)
        for resource in delete_list:
            if resource.resource_type == "BootVolume" and 
resource.lifecycle_state == "AVAILABLE":
                bv_ocid = resource.identifier
                bv_name = resource.display_name
                print("Deleting the boot volume {}, with ocid {}
".format(bv_name, bv_ocid))
                try:

bv_composite_operations.delete_boot_volume_and_wait_for_state(
                        boot_volume_id=bv_ocid,

wait_for_states=[oci.core.models.BootVolume.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the boot volume successfully!")
                except Exception as e:
                    print("Error while deleting the boot volume name {}, 
ocid {}, Error message {}".format(bv_name,

                                bv_ocid,

                                str(e)))


if __name__ == '__main__':
    no_of_args = len(sys.argv)
    if no_of_args < 2:
        print("Usage: ")
        print("To list all the resources based on service name prefix:")
        print("python3 remove_resources.py <service_name_prefix>")
        print("To remove all the resources based on service name prefix:")
        print("python3 remove_resources.py <service_name_prefix> delete")
        sys.exit(1)

    service_prefix = sys.argv[1]
    print("Service prefix name:" + service_prefix)
    cleanup_util = CleanUpResources()
    if len(service_prefix) >= 16:
        service_prefix = service_prefix[0:16]
    service_prefix = service_prefix + "-"
    if no_of_args < 3:
        print("Listing all resources with service prefix name" +
service_prefix)
        cleanup_resources = cleanup_util.list_all_resources(service_prefix)
    elif no_of_args < 4 and sys.argv[2] == "delete":
        print("Deleting all resources with service prefix name" +
service_prefix)
        cleanup_resources = cleanup_util.list_all_resources(service_prefix)
        cleanup_util.cleanup_resources(cleanup_resources)
        cleanup_util.delete_dynamic_group(service_prefix)
```

2. Run the following command to list all the resources:

   ```
   python3 remove_resources.py <full_service_prefix_name>
   ```

3. Check that list and ensure that you want to delete these resources.
4. Run the following command to deleted all the resources of the instance:

   ```
   python3 remove_resources.py <full_service_prefix_name> delete
   ```

Missing uuid for /u01/data or /u01/app on Reboot

Oracle WebLogic Server for OCI adds the UUID for volume mounts to /etc/fstab . In some instances no UUID value is returned for the mount, so an empty value is entered for UUID into /etc/fstab.

To see if this is the cause of the failed mount, run the following:

```
cat /etc/fstab | grep /u01/
```

Example output for /u01/app:

```
UUID=5341b4d8-9905-479d-be44-9241349add47 /u01/app ext4
auto,defaults,_netdev,nofail 0 2
```

Example output for /u01/data:

```
UUID= /u01/data ext4 auto,defaults,_netdev,nofail 0 2
```

Workaround

1. Locate the device to mount.

In almost all cases where this is observed, the mount only fails for one of the devices. Run the following to locate the device:

```
sudo lsblk | grep /u01/
```

Example output:

```
sdb 8:16 0 50G 0 disk /u01/app
```

In the above example `/u01/app` is mounted to device sdb. The other device WebLogic Server for OCI provides is sdc. Therefore, the device to mount `/u01/data` would be sdc. If the output had returned sdc for `/u01/app` then you would use sdb as the device.

2. Mount the device.
   Continuing with the example from above, this command would be:

   ```
   sudo mount -v -t ext4 /dev/sdc /u01/data
   ```

   If this command returns an error including "does not contain SELinux labels" and /u01/data is still empty, run the following:

   ```
   sudo restorecon -Rv /
   ```

   ```
   sudo mount -v -t ext4 /dev/sdc /u01/data
   ```

   If `/u01/data` is still empty, you can wait 30 minutes, and try the above commands again, or you can move on to step 3 and reboot after modifying `/etc/fstab`.
3. Get the UUID regardless of whether the device was mounted.
   Continuing with the example:

   ```
   sudo blkid -s UUID -o value /dev/sdc
   ```

   Example UUID value output:

   ```
   5341b4d8-9905-479d-be44-9241349add47
   ```
4. Update /etc/fstab with the acquired UUID so future reboots won't experience this problem. Either use sudo vi to update /etc/fstab with the value retrieved from step 3 or use a sed command like below:

   ```
   sudo sed -i -e "s;UUID= /u01/data;UUID=5341b4d8-9905-479d-
   be44-9241349add47 /u01/data;" /etc/fstab
   ```
5. If the device was not mounted in step 2, then reboot the instance.
   Due to completion of step 4, it should be mounted on reboot.

Error in Mounting the Volume During Provisioning

Troubleshoot a failed Oracle WebLogic Server for OCI domain due to Oracle Cloud Infrastructure (OCI) policy not in effect.

Issue:

When you create a domain with the **OCI Policies** check box selected, at times, the policy creation takes longer time than it takes to reach the first OCI API call, that is, the call to create a volume mount, and an error message is displayed in the bootstrap log.

Example message:

```
Exception stacktrace [
{'opc-request-id': '<GUID>', 'code': 'NotAuthorizedOrNotFound', 'message':
'Authorization failed or requested resource not found.', 'status': 404}
]
Error executing volume mounting.. Exiting provisioning
```

Workaround:

1. Create a dynamic group using the following rule:
   `All { instance.compartment.id = '<resource_compartment ocid>') }`

   To create a dynamic group, see Create a Dynamic Group.
2. Create the policy for the dynamic group as follows:
   Example policy:

   ```
   Allow dynamic-group MyInstancesPrincipalGroup to use volumes in
   compartment MyCompartment where any { request.operation = 'AttachVolume',
   request.operation = 'DetachVolume'}"
   ```

   To create other policies for the dynamic group, see Create Policies for the Dynamic Group.
3. Wait for 10 to 15 minutes and then create a domain without selecting the **OCI Policies** check box.
4. Delete the dynamic group and policy created manually after the domain is created.

Unable to Access the WebLogic Console From the Internet

Troubleshoot problems accessing the WebLogic Console of an Oracle WebLogic Server domain after it's successfully created.

By default the WebLogic Server Administration Console is accessed through port 7001 or 7002.

To check port access:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Networking**, and then click **Virtual Cloud Networks**.
3. Select the compartment in which you created the domain.
4. Select the virtual cloud network in which the domain was created.
5. Select the subnet where the WebLogic Server compute instance is provisioned.
6. Select the security list assigned to this subnet.
7. For a domain that's not on a private subnet, make sure the following ingress rules exist:

   ```
   Source: 0.0.0.0/0
   IP Protocol: TCP
   ```

```
Source Port Range: All
Destination Port Range: 7002


Source: 0.0.0.0/0
IP Protocol: TCP
Source Port Range: All
Destination Port Range: 7001
```

For a domain on a private subnet, set the `Source` to the CIDR of the bastion instance subnet.

Unable to Access the Sample Application Using the Load Balancer

Troubleshoot problems accessing the load balancer of an Oracle WebLogic Server domain after it's successfully created.

Cannot access the sample application using the load balancer: Not Found

On a domain running Oracle WebLogic Server Standard Edition, the sample application is deployed only to the first Managed Server. If your Standard Edition domain has multiple Managed Servers and you access the sample application using a load balancer, the Managed Servers that aren't hosting the sample application will respond with the code `404` (Not Found).

You can use the WebLogic Server Administration Console to update the targets for the sample application, and add the remaining Managed Servers.

Cannot access applications using the load balancer: Bad Gateway

If you restart the compute instances running your Managed Servers, or you restart the compute instances running the App Gateway, the backend set of the load balancer will temporarily be in an unhealthy state. By default, a load balancer in this state will respond with the code `502` (Bad Gateway). After the WebLogic Server and App Gateway processes are running, the load balancer should return to the OK state.

To check the status of the load balancer and backend servers:

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Networking**, and then click **Load Balancers**.
3. Click the load balancer that was created for your domain, *`prefix`*`-lb`.
4. Click **Backend Sets**, and then click *`prefix`*`-lb-backendset`.
5. Click **Backends**, and then check the state of each backend.
6. Access the WebLogic compute instances using a secure shell (SSH) client. Check that the Managed Server process is listening on its assigned port (the default is 7003).

   ```
   curl -s -o /dev/null -w "%{http_code}\n" http://private_ip:7003
   ```

   A `404` response indicates that the Managed Server is running.

7. If you enabled authentication with Oracle Identity Cloud Service, then access the App Gateway compute instances using an SSH client. Check that the App Gateway process is listening on its assigned port (the default is 9999).

   ```
   curl -s -o /dev/null -w "%{http_code}\n" http://private_ip:9999
   ```

   A `404` response indicates that the App Gateway is running.

See Managing Backend Servers in the Oracle Cloud Infrastructure documentation.

Unable to Access the Fusion Middleware Control Console or the Enterprise Manager Console From the Internet

Troubleshoot problems accessing the Fusion Middleware Control Console and the Enterprise Manager Consoles of an Oracle WebLogic Server domain after it is created successfully.

If you have enabled authentication with Oracle Identity Cloud Service on a WebLogic Server 14.1.2.0 or 12.2.1.4 domain, you might be redirected to an error page when you try to log in to the Fusion Middleware Control Console or the Enterprise Manager Console.

Example error message:

```
<Error> <oracle.help.web.rich.OHWFilter>
<BEA-000000> <ADFSHARE-00120: Error encountered while creating the MDS
Session. Application state will be reset. Please logout and log back in if
problem persists.
oracle.adf.share.ADFShareException: ADFSHARE-00120: Error encountered while
creating the MDS Session. Application state will be reset. Please logout and
log back in if problem persists.
```

You need to add the Cloud Gate privilege to access the consoles.

To access the Consoles:

1. Add the Cloud Gate App Role to your confidential application that you created for the domain.
   a. Access the Oracle Identity Cloud Service console.
   b. From the navigation menu, click **Applications**.
   c. Click the confidential application that was created for your domain.
   d. Click the **Configuration** tab.
   e. Under Client Configuration, locate **Grant the client access to Identity Cloud Service Admin APIs**, and then click **Add**.
   f. Select the **Cloud Gate** App Role and click **Add**.
   g. Click **Save**.
2. Restart your WebLogic Server domain and log in to the Fusion Middleware Control Console or the Enterprise Manager Console again.

See Create a Confidential Application.

> **Note:**
>
> This issue is seen only in releases earlier than 23.2.3. From release 23.2.3 onward, the confidential application for a JRF domain is set with the `Authenticator Client` and `Cloud Gate` roles by default. Hence, this workaround is not required.

Load Balancer Does Not Send Cookie `X-Oracle-BMC-LBS-Route`

When you setup Oracle WebLogic Server for OCI with a load balancer, the load balancer does not send cookie `X-Oracle-BMC-LBS-Route`.

Scenario:

1. Create a 2-node WebLogic instance with load balancer by using a Oracle WebLogic Server for OCI listings in marketplace.
2. Access the sample app through load balancer.
3. In your web browser, go to **WebDeveloper** > **Web Console** > **Network**.
4. Click **Reload**.
5. Click on **Get request** of the sample app, then select the **Cookies** tab
   The cookies tab is empty.

Workaround:

1. Sign in to the Oracle Cloud Infrastructure console.
2. From the navigation menu, click **Networking**, and then click **Load Balancers**.
3. Click the name of the **Compartment** that contains the load balancer you want to modify, and then click the load balancer's name.
4. , and then click the name of the backend set you want to modify.
5. In the **Resources** menu, click **Backend Sets**. Deselect **HTTP Only**.
6. Save the changes.
7. Undo the changes you did in step 5 and then save the changes.
8. Reload the sample app browser.
   Now you can view that the cookie with name `X-Oracle-BMC-LBS-Route` is passed properly.

Reapply Fails for Load Balancer Configuration

Troubleshoot the problem for load balancer configuration in Oracle WebLogic Server for OCI during stack reapply.

Issue:

For a stack that is provisioned with a new load balancer, reapply fails, if you modify the following load balancer configuration:

- Change the CIDR for the load balancer subnet
- Change the load balancer from public to private or vice versa

Workaround:

1. Sign in to the Oracle Cloud Infrastructure Console.

2. Click the navigation menu , select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
3. Select the Compartment that contains your stack.
4. Click the name of your stack.
5. Click **Variables** and then click **Edit Variables**.
6. Deselect **Provision Load Balancer**.
7. Click **Next** and save the changes.
8. On the Stack Details page, click **Apply**.
9. In the Apply panel, enter a name for the job and click **Apply**.
10. Periodically monitor the progress of the Apply job until it is finished.
11. Go to the Stack Details page, and click **Variables** and then click **Edit Variables**.
12. Select **Provision Load Balancer**.
13. Configure the load balancer based on your requirements.
    - Change the CIDR of the load balancer subnet.
    - Select **Private Load Balancer** to change from public to private or deselect **Private Load Balancer** to change from private to public.

      > **Note:**
      >
      > This option is available only if you used a private subnet for WebLogic Server.
14. Click **Next** and save the changes.
15. On the Stack Details page, click **Apply**.
16. In the Apply panel, enter a name for the job and click **Apply**.
17. Periodically monitor the progress of the Apply job until it is finished.

The load balancer configuration changes are now applied on the stack.

Autoscaling Failed to Create Functions

When you create an Oracle WebLogic Server for OCI domain with autoscaling enabled, functions may not be created during provisioning.

Issue:

During provisioning, if autoscaling failed to create the functions due to invalid OCIR auth token value, provisioning is successful but the following error is displayed in the log:

```
module.provisioners.null_resource.print_service_info[0] (remote-exec):
************************************************************
module.provisioners.null_resource.print_service_info[0] (remote-exec): This
service is configured with the following options .....
module.provisioners.null_resource.print_service_info[0] (remote-exec):
WebLogic Server for OCI Version : 22.1.1-220208100422
module.provisioners.null_resource.print_service_info[0] (remote-exec):
```

```
WebLogic Server Version: 12.2.1.4
module.provisioners.null_resource.print_service_info[0] (remote-exec):
WebLogic Server Edition: SUITE
module.provisioners.null_resource.print_service_info[0] (remote-exec):
Virtual Cloud Network : NEW VCN
module.provisioners.null_resource.print_service_info[0] (remote-exec):
Network Type: PRIVATE Network with BASTION
module.provisioners.null_resource.print_service_info[0] (remote-exec): Domain
Type: Plain WebLogic Server Domain (non-JRF)
module.provisioners.null_resource.print_service_info[0] (remote-exec): APM
agent enabled : [True]
module.provisioners.null_resource.print_service_info[0] (remote-exec): APM
agent installed : [True]
module.provisioners.null_resource.print_service_info[0] (remote-exec): Failed
to create autoscaling resources, Check provisioning logs for details
module.provisioners.null_resource.print_service_info[0] (remote-exec): User
can invoke remove_resources.py script and then rerun the
configure_autoscaling.sh script on Admin VM to recreate the resources.
module.provisioners.null_resource.print_service_info[0] (remote-exec):
************************************************************
```

Workaround:

Perform the following steps to create the function resources:

1. Update the OCIR auth token secret to the valid OCIR auth token as follows:
   - In the script `/opt/scripts/observability/autoscaling/configure_autoscaling.sh`, replace the line `ocir_auth_token=$(python3 /opt/scripts/wls_credentials.py ocirAuthToken)` with `ocir_auth_token='<VALID_AUTH_TOKEN>'`.
2. Run the script to delete resources to clean up any resources created by autoscaling during provisioning from WebLogic administration instance.
   `python3 remove_resources.py pre-destroy <service_name_prefix> -f autoscaling`
3. Log in as a `root` user to the Administration server and run the `configure_autoscaling.sh` script.
   `/opt/scripts/observability/autoscaling/configure_autoscaling.sh`

   This script creates the autoscaling functions using the valid OCIR auth token from step 1. If you encounter any errors when you run the script, see `/u01/logs/provisioning.log`.
4. In the OCI console, verify if:
   - Autoscaling functions are created under the function application.
   - Notification subscriptions are created for Scale Out and Scale In notification topics.
   - Event rule is created for the stack.

Management Agents Are Not Deleted on Instance Termination

In case of an Oracle WebLogic Server for OCI domain with autoscaling enabled, when you destroy the stack, the management agent resources associated with the compute instance are not destroyed.

Issue:

When you destroy the stack, the compute instance for the domain is terminated but the associated management agent resources are active.

Workaround:

Perform the following steps to manually delete the management agent resources:

> **Note:**
>
> You must run the following commands from the Cloud Shell. You can also install OCI CLI and create the config file on your host, and then run the following commands. See Installing the CLI.

1. List the management agents in the stack compartment.

   ```
   oci management-agent agent list --compartment-id <stack_compartment_ocid>
   oci management-agent agent list --compartment-id <stack_compartment_ocid>
   | grep ocid1.managementagent | grep '"id":'
   ```

2. Delete the management agents associated with your service.

   ```
   oci management-agent agent delete --agent-id
   <OCID_of_the_managementagent_from_step1> --force
   ```

Enterprise Manager Console Is Not Loading

Issue: After you create a JRF-enabled domain without Oracle Identity Cloud Service in Oracle WebLogic Server for OCI, you are unable to log into the Enterprise Manager console.

> **Note:**
>
> This issue is applicable for stacks created between 31st August, 2022 and 20th September, 2022 (22.3.2 release).

Workaround:

1. Edit the `jps-config.xml` located in `/u01/data/domain/<domain-name>/config/fmwconfig/`. Replace `idstore.scim` with `idstore.ldap`.

   ```
   <serviceInstanceRef ref="idstore.ldap"/>
   ```

2. Execute the `restart_domain.sh` script.

   ```
   /opt/scripts/restart_domain.sh -o restart
   ```

Scale Out Fails on the Administration Compute Instance

Scale out fails due to high CPU usage on the administration compute instance.

Issue:

If the administration compute instance is stressed to 100 percent utilization, the scale out fails as the `pack` commands that are run during scale out on the administration compute instance do not give any results and time out.

Workaround:

Verify if the control group `wlsmcg` exists under `/sys/fs/cgroup/cpu`. If it exists, assign process IDs for the administration server and managed server to the control group, and assign CPU shares to the control group to manage the CPU usage. So, you can successfully run the `pack` command.

> **Note:**
>
> If you start the servers on the administration compute instance using scripts or through the WebLogic console, the new process IDs for the server are not added to the control group.

To reassign the process ID for the administration server and managed server to the control group:

1. Check if current server process ID is part of the control group `wlsmcg`.

   ```
   cat /sys/fs/cgroup/cpu/wlsmscg/tasks | grep "<processID for managed
   server>"
   ```

2. If the process ID is not found in the step 1, run the following script as the `opc` user to create control groups and assign process IDs for the administration server and managed server.

   ```
   sudo /opt/scripts/create_control_groups.sh
   ```

3. Verify that managed server process ID is assigned to the control group, *wlsmscg*.

   ```
    cat /sys/fs/cgroup/cpu/wlsmscg/tasks | grep "<processID for managed
   server>"
   ```

Warnings for Schema Password

You receive warning messages about password expiry for your infrastructure schema password when you log in to the administration instance. If you want to reset the password, follow the instructions in Update the Infrastructure Schema Password.

Security Checkup Tool Warnings

Learn about the security check warnings that are displayed in the Oracle WebLogic Server Administration console and how to troubleshoot them.

At the top of the WebLogic Server Administration console, the message `Security warnings detected. Click here to view the report and recommended remedies` is displayed for Oracle WebLogic Server for OCI instances created after July 20, 2021, or the instances on which the July 2021 PSUs are applied.

When you click the message, a list of security warnings are displayed as listed in the following table.

The warning messages listed in the table are examples.

Security Warnings

Warning Message	Resolution
`The configuration for key stores for this server are set to Demo Identity and Demo Trust. Trust Demo certificates are not supported in production mode domains.`	Configure the identity and trust keystores for each server and the name of the certificate in the identity keystore that the server uses for SSL communication. See Configure Keystore Attributes for Identity and Trust. **Note:** This warning is displayed for Oracle WebLogic Server for OCI instances created after October 20, 2021, or the instances on which the October PSUs are applied.
`SSL hostname verification is disabled by the SSL configuration.` You see the SSL host name verification warnings in case of existing Oracle WebLogic Server for OCI instances created before release 21.3.2 (August 17, 2021).	Review your applications before you make any changes to address these SSL host name security warnings. For applications that connect to SSL endpoints with a host name in the certificate, which does not match the local machine's host name, the connection fails if you configure the BEA host name verifier in Oracle WebLogic Server. See Using the BEA Host Name Verifier in Administering Security for Oracle WebLogic Server. For applications that connect to Oracle provided endpoints such as Oracle Identity Cloud Service (for example,`*.identity.oraclecloud.com`), the connection fails if you did not configure the wildcard host name verifier or a custom host name verifier that accepts wildcard host names. If you are not sure of the SSL configuration settings you should configure to address the warning, Oracle recommends that you configure the wildcard host name verifier. See Using the Wildcard Host Name Verifier in *Administering Security for Oracle WebLogic Server*. **Note**: For WebLogic Server 14.1.1.0.0 and 14.1.2.0.0, the default host name verifier is set to the wildcard host name verifier.
`Remote Anonymous RMI T3 or IIOP requests are enabled. Set the RemoteAnonymousRMIT3Enabled and RemoteAnonymousRMIIIOPEnabled attributes to false.`	Disable the anonymous RMI T3 and IIOP requests in the WebLogic Server Administration Console as soon as possible unless your deployment requires anonymous T3 or IIOP (not typical). See Disable Remote Anonymous RMI T3 and IIOP Requests.

After you address the warnings, you must click **Refresh Warnings** to see the warnings removed in the console.

For Oracle WebLogic Server for OCI instances created after July 20, 2021, though the java properties to disable anonymous requests for preventing anonymous RMI access are configured, the warnings still appear. This is a known issue in Oracle WebLogic Server.

If you want to perform anonymous RMI requests, you must disable the java properties. Go to the `nodemanager.properties` file located under `DOMAIN_HOME/nodemanager` and remove the `weblogic.startup.Arguments` property.

Disable Remote Anonymous RMI T3 and IIOP Requests

To disable the remote anonymous RMI T3 and IIOP requests in the WebLogic Server Administration console:

1. Locate the **Change Center** and click **Lock & Edit** to lock the editable configuration hierarchy for the domain.
2. Under **Domain structure**, select the domain name, and then select the **Security** tab.
3. Expand **Advanced** and deselect **Remote anonymous RMI access via IIOP** and **Remote anonymous RMI access via T3**.

After saving the changes, return to **Change Center** and click **Activate Changes**.

Configure Keystore Attributes for Identity and Trust

To configure the identity and trust keystore files and the name of the certificate in the identity keystore in the WebLogic Server Administration console:

1. Locate the **Change Center** and click **Lock & Edit** to lock the editable configuration hierarchy for the domain.
2. Under **Domain structure**, select **Environment** and then select **Servers**.
3. In the Servers table, select the server you want to configure.
4. On the **Configuration** tab, click **Keystores**, and then click **Change**.
5. Select *Custom Identity and Custom Trust*, and then click **Save**.
6. Under **Identity**, provide the following details:
   a. Enter the full path of your identity keystore.

      For example: `/u01/data/keystores/identity.jks`

   b. For **Custom Identity Keystore Type**, enter *JKS*.

   c. For **Custom Identity Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Identity Keystore Passphrase**.
7. Under **Trust**, provide the following details:
   a. Enter the full path of your identity keystore.

      For example, `/u01/data/keystores/trust.jks`

   b. For **Custom Trust Keystore Type**, enter *JKS*.

   c. For **Custom Trust Keystore Passphrase**, enter your keystore password. Enter the same value for **Confirm Custom Trust Keystore Passphrase**.
8. Click **Save**.
9. Click the **SSL** tab.
10. Under **Identity**, provide the following details:
    a. For **Private Key Alias**, enter the name of the certificate (private key) in the identitykeystore, *server_cert*.

    b. For **Private Key Passphrase**, enter the password for this certificate in the keystore. Enter the same value for **Confirm Private Key Passphrase**.

       By default, the password for the certificate is the same as the identity keystore password.
11. Click **Save**.

    After saving the changes, return to **Change Center** and click **Activate Changes**.
12. Repeat steps 3 to 9 to configure each server in the domain.

Running `python3` Command Fails

Issue: When you run a command that uses `python3`, it might fail on the stacks that were created earlier than May 25, 2022.

Workaround: If your stack is created earlier than May 25, 2022, then in the command use `python`. For stacks created after May 25, 2022, use `python3`.

Unrecognized Arguments When Using the Patching Utility Tool

When you run the patching utility tool with some of the documented arguments, you see the `unrecognized arguments` message.

Issue:

Run the patching utility tool to list latest patches, list pending patches, download latest patches, apply patches, and roll back patches using `patch-utils` with the following arguments:

```
#List patches
patch-utils list -L
#List pending patches
patch-utils diff
#Download latest patches
patch-utils download -L -p /tmp/<Location to download>
#Apply patches
patch-utils apply -L
#Roll back patches
patch-utils rollback -L
```

The following message is displayed:

```
usage: patch-utils <action> [options]
patch-utils: error: unrecognized arguments:
```

The listed arguments correspond to latest features added to the patching utility tool for Oracle WebLogic Server for OCI instances created after December 15, 2022 (22.4.3). So, if you are using Oracle WebLogic Server for OCI instances created before release December 15, 2022, you see the `unrecognized arguments` message.

Workaround:

Run `patch-utils upgrade` to upgrade the patching tool, if you are using the latest features of the patching utility tool for your existing instances (created before release December 15, 2022). See Upgrade Patching Tool.

Get Additional Help and Contact Support

Use online help, email, customer support, and other tools if you have questions or problems with Oracle WebLogic Server for OCI.

For customer support, you can create support tickets using the Oracle Cloud Infrastructure (OCI) console or My Oracle Support.

Create Support Ticket Using OCI Console

Use the Support Center in Oracle Cloud Infrastructure console to create a support ticket for your technical issues for Oracle WebLogic Server for OCI service in the Marketplace.

> **Note:**
>
> Make sure to provision your support account before you create a support request. See Configuring Your Oracle Support Account in Oracle Cloud Infrastructure documentation.

To create a support ticket:

1. Sign in to the Oracle Cloud Infrastructure console.
2. Click the navigation menu ☰, and select **Governance & Administration**. Under **Support**, click **Support Center**.
3. Click **Create Support Request**.
   The **Technical Support** tab on the Support Options page is displayed.
4. For **Issue Summary**, enter the a title that summarizes your issue.
5. For **Describe Your Issue**, enter a brief description of your issue.
6. Select the severity level of the issue based on the impact of service.
7. Select *Compute* from the **Select Service** list.
8. Select *Oracle WebLogic Cloud* from the **Select Category** list.
9. Select the type of issue you are experiencing.
10. Click **Create Support Request**.

After you submit the request, My Oracle Support sends a confirmation email to the address provided in the primary contact details. A follow-up email is sent if additional information is required.

Optionally, you can create a support ticket using the Help menu ? and the Support button 🛟 in Oracle Cloud Infrastructure console. See Support Ticket Management in Oracle Cloud Infrastructure documentation.

However, when you a create support ticket using these options, the support ticket may not be assigned to a specific service or component for resources like compute instances, networks and load balancers. So, it is recommended to use Support Center in Oracle Cloud Infrastructure console to create support tickets.

Create Support Ticket Using My Oracle Support

Use the Service Request in My Oracle Support to create a support ticket for your technical issues for Oracle WebLogic Server for OCI service in the Marketplace.

> **Note:**
>
> Make sure you have a Support Identifier which verifies your eligibility for Support services, and an account at My Oracle Support.

To create a support ticket:

1. Sign in to My Oracle Support.
2. On the **Service Requests** tab, click **Create Technical SR**.
3. Enter the **Problem Summary** and the **Problem Description**.
4. Under **Where is the Problem**, click **Cloud**.
5. Select *Oracle WebLogic for Cloud Infrastructure* from the **Service Type** list.
6. Select the tenancy from the **Service** list.
7. Select a **Problem Type** and provide the **Support Identifier** details.
8. Click **Next** until you have provided all the mandatory information.
9. Click **Submit**.
   Your service request is created.

For general help with Oracle Cloud Marketplace, see How Do I Get Support in Oracle Cloud Infrastructure documentation.

Unable to Fetch the Password Expiry Date for the OPSS User

For JRF-enabled Oracle WebLogic Server for OCI instances created after release 23.1.1, if your Oracle Platform Security Services (OPSS) schema password is nearing expiry, you will get a notification about the password expiry date so you can update your schema password. However, if you are using a database connection string to provision a domain an Oracle WebLogic Server domain, you will encounter an error.
Issue

If you create a Oracle WebLogic Server for OCI instance with JRF using the database connections string, you will encounter the following error message:

```
<WLSC-ERROR> : Unable to get schema password expiry date. Failed to get password expiry
date for OPSS User. For details, refer the log, /u01/logs/opss_pwd_expiry.log
```

The `opss_pwd_expiry.log` file will display the following:

```
SEVERE: ERROR: Failed to get password expiry date for OPSS User.ORA-65015: missing or
invalid container name
```

Workaround

This error occurs because the pluggable database (PDB) name is not set in the metadata. Therefore, when you run the `opss_password_expiry_date.sh` script, it fetches an empty string for the <*pdb_name*> attribute.

To avoid the error:

1. Set the metadata attribute `pdb_name` when you use a database connection string to provision an Oracle WebLogic Server for OCI instance.

   ```
   python3 /opt/scripts/databag.py pdb_name
   ```

2. If the above command results in an empty value, update the database value:

   ```
   python3 /opt/scripts/utils/update_metadata.py -k pdb_name -v <pdb_name>
   ```

3. Execute Step 1 again to confirm that the value is updated:

   ```
   python3 /opt/scripts/databag.py pdb_name
   ```

4. Run the following command to confirm that the workaround is successful:

   ```
   bash /opt/scripts/validation/opss_password_expiry_date.sh
   ```

OS Management Service Agent Not Enabled

Use these steps to manually enable an OS Management Service Agent.

You selected **Enable OS Management Hub** during Stack creation, but when you access the OS Management tab in the Compute Instance details page in the OCI Console, you see the followiing message:

```
OS Management Hub Agent is not enabled for this instance
```

Issue

When a Compute instance is created by a WebLogic Server for OCI stack it will enable OS Management Hub Agent. When this agent is enabled it will create an OS Management Hub managed instance. The stack start up (aka bootstrapping) must wait for the managed instance to be created in order to register the OS Management Hub Profile with the instance. On occasion, the time it takes for the OS Management Hub to create the managed instance will exceed the wait time set in the stack instance.

Workaround:

1. Open the OCI Console navigation menu and select **Compute**. Under Compute, select **Instances**.
2. Select the OCI Compute instance where OS Management Hub is not enabled.
3. Select the "Management" tab.
4. Disable "OS Management Service Agent" if it is enabled.
5. Enable "OS Managment Hub Agent".
6. You will be moved to a Set Profile page.
7. Select the compartment for the profile you created or selected during stack creation.
8. Select the profile you created or selected during stack creation.
9. Click **Set Profile**.
10. The agent will show it in a "Stopped" state.
11. Wait 5 to 30 minutes for the state to change to "Running".

If the state hasn't changed to "Running" verify that you have this dynamic group policy targeted to your Stack compartment. Replace MyInstancePrincipalGroup with the dynamic group targeted for the Stack compartment and MyCompartment with your Stack compartment name:

```
Allow dynamic-group MyInstancesPrincipalGroup to
{OSMH_MANAGED_INSTANCE_ACCESS} in compartment MyCompartment where
request.principal.id = target.managed-instance.id
```

If the dynamic group is configured consult the Examining Log Files on an Instance documentation. If the Agent shows the status Registration Failed consult the Registration Failed for Instance or Management Station documentation.

OS Management Monitored Resources are Not Available

You selected "Enable OS Management Hub" during Stack creation but see the message "Monitored resources are not available" when you access the "OS Management" tab in the Compute Instance details page in the OCI Console.

Issue

The dynamic group policy required to monitor resources is not set up.

Workaround

Set the following dynamic group policy targeted to your Stack compartment. Replace `MyInstancePrincipalGroup` with the dynamic group targeted for the Stack compartment and `MyCompartment` with your Stack compartment name:

```
Allow dynamic-group MyInstancesPrincipalGroup  to
{MGMT_AGENT_DEPLOY_PLUGIN_CREATE, MGMT_AGENT_INSPECT, MGMT_AGENT_READ} in
compartment MyCompartment
```

Stack Creation Failed

Troubleshoot a failed Oracle WebLogic Server domain that you attempted to create with Oracle WebLogic Server for OCI.

View the stack log files

Use the Terraform job logs in Resource Manager to identify the cause of the failure.

1. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Jobs**.
2. Identify and click the job for your stack.
   - The Type is Apply.
   - The State is Failed.
   - The Stack is the name of your Oracle WebLogic Server for OCI stack.
3. From the Logs section, search the log for error messages.
   You can optionally **Download** the log files and search the files offline.
4. See below for details about specific error messages.

Modify the stack configuration

If necessary, delete the current stack resources, modify your stack configuration, and then apply the changes.

1. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
2. Click the name of your stack.

3. Click **Terraform Actions** and select **Destroy**.
   Wait for the destroy job to complete.
4. Click **Edit Stack**.
5. When done, click **Save Changes**.
6. Click **Terraform Actions** and select **Apply**.

Cannot launch a stack in Marketplace

Example message: `Unable to accept Terms of Use`

In Marketplace, you might see the message when you click **Launch Stack**, after you've selected a stack version and compartment, and checked the Oracle Standard Terms and Restrictions box.

You likely don't have permission to:

- Create Marketplace applications in the selected compartment. Verify that this policy exists in the compartment where you want to create the stack.
  `Allow group *Your_Group* to manage app-catalog-listing in compartment *Your_Compartment*`
- Access the selected compartment. Choose another compartment or ask your administrator.

Cannot determine home region

Example message:

```
data.oci_core_app_catalog_subscriptions.mp_image_subscription[0]: Refreshing
state...
Error: Null value found in list ... "oci_identity_regions" "home-region"
```

If you are not an administrator, ask them to verify that the following root-level policy exists in your tenancy:

```
Allow group Your_Group to inspect tenancies in tenancy
```

Cannot find dynamic group and secrets policy

Example messages:

```
Error: Service error:NotAuthorizedOrNotFound. Authorization failed or
requested resource not found. http status code: 404.
 Opc request id: request_id on modules/policies/groups.tf line 8, in
resource...
 "oci_identity_dynamic_group" "wlsc_instance_principal_group" {


Error: Service error:NotAuthorizedOrNotFound. Authorization failed or
requested resource not found. http status code: 404.
 Opc request id: request_id on wlsc-policies.tf line 10, in resource...
 "oci_identity_policy" "wlsc_secret-service-policy" {
```

When the **OCI Policies** check box is selected (by default), Oracle WebLogic Server for OCI creates a dynamic group and one or more root-level policies in your tenancy.

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level permissions to create domains. If you are not an administrator, ask them to verify that root-level policies exist in your tenancy. For example:

```
Allow group Your_Group to manage dynamic-groups in tenancy
Allow group Your_Group to manage policies in tenancy
Allow group Your_Group to use secret-family in tenancy
```

See:

- Create Root Policies
- Create Policies for the Dynamic Group

Maximum number of dynamic groups has exceeded

Example message:

```
<WLSC-VM-ERROR-0119> : Failed to get secret content for
[ocid1.vaultsecret.oc1.iad.alongstring123]: [{'status': 400, 'message': "This
instance principal matches more than '5' dynamic groups, update your dynamic
groups' matching rules"...'}]>
```

When the **OCI Policies** check box is selected (by default), Oracle WebLogic Server for OCI creates a dynamic group and one or more root-level policies in your tenancy. The maximum number of dynamic groups allowed is 5.

Solution:

1. Add the following policy that uses the existing dynamic group to access the new secrets for the new stack:

   ```
   Allow dynamic-group <existing-dyanmic-group-name> to read secret-bundles
   in tenancy where target.secret.id = '<OCID_of_the_secret>'
   ```

2. Deselect the **OCI Policies** check box and try to create the stack again.

Unable to get secret content or decrypted credential

Example messages:

- `Failed to get secret content for Your_vault_secret_OCID`
- `Authorization failed or requested resource not found`
- `Error retrieving %s password from Secret Vault`
- `Failed in create domain due to exception [Failed to retrieve WebLogic Password from Secrets Vault]`
- `Failed to retrieve IDCS Client Secret from Secrets Vault`
- `Unable to get decrypt credential`
- `Key or Vault does not exist or you are not authorized to access them.`

When you create a domain with Oracle WebLogic Server for OCI, you provide the OCID values of the secrets that contain the passwords to use for the domain and during provisioning. The compute instances use this information to decrypt the passwords. The compute instances are granted access to vault secrets using policies.

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level permissions to create domains. If you are not an administrator, ask them to verify that relevant vault secret policies exist in your tenancy and compartment. For example:

```
Allow group Your_Group to use secret-family in tenancy
Allow dynamic-group Your_DynamicGroup to use secret-family in compartment
MyCompartment
Allow dynamic-group Your_DynamicGroup to use keys in compartment MyCompartment
Allow dynamic-group Your_DynamicGroup to use vaults in compartment
MyCompartment
```

If the policies exist, check that the OCID of the compartment in listed in dynamic group.

See:

- Create Secrets for Passwords
- Create Policies for the Dynamic Group
- Create Root Policies

Unable to get decrypted credential when creating a stack in a private subnet

Example message: `<WLSC-VM-ERROR-001> Unable to get decrypt credential [HTTPSConnectionPool(host='auth.us-phoenix-1.oraclecloud.com', port=443): Max retries exceeded with url: /v1/x509 (Caused by ConnectTimeoutError(<oci._vendor.urllib3.connection.VerifiedHTTPSConnection object at 0x1e5110>, 'Connection to auth.us-phoenix-1.oraclecloud.com timed out. (connect timeout=10)'))]>`

When you create a domain with Oracle WebLogic Server for OCI in an existing private subnet, provisioning fails if the WebLogic Server subnet is using a route table that does not include a service gateway or a Network Address Translation (NAT) gateway.

Modify the private subnet, and select a route table that uses a service gateway or NAT gateway. Or select a virtual cloud network (VCN) whose default route table uses a service gateway or NAT gateway. Refer to these topics:

- VCNs and Subnets
- Access to Oracle Services: Service Gateway

Failed to download Oracle Autonomous Database wallet

Example message: `module.provisioners.null_resource.status_check[0] (remote-exec): <Nov 23, 2019 09:37:17 PM GMT> <ERROR> <oci_api_utils> <(host:stackname-wls-0.subnetxxx.stacknamevcn.oraclevcn.com) - <WLSC-VM-ERROR-0052> : Unable to download atp wallet. [{'status': 403, 'message': u'Forbidden', 'code': u'Forbidden', 'opc-request-id': 'FA6C16D8B'}]`

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level and compartment-level permissions to create domains. Access to the database wallet is needed when you create a JRF-enabled domain that uses an autonomous database. If you are not an administrator, ask them to verify that relevant policies for autonomous databases exist in your tenancy and compartment. For example:

```
Allow group Your_Group to inspect autonomous-transaction-processing-family in
compartment Your_ATP_Compartment
```

```
Allow dynamic-group Your_DynamicGroup to inspect autonomous-transaction-
processing-family in compartment Your_ATP_Compartment
```

See:

- Create Policies for the Dynamic Group
- Create Root Policies
- Create Compartment Policies

Failed to validate DB connectivity

When you create a domain that includes the Java Required Files (JRF) components, you must select an existing database and provide connection details. The compute instances use this information to connect to the database and provision the JRF database schemas.

Possible causes for this error include:

- You entered the wrong database password or a plain text password.
- The database does not allow the compute instances to access its listen port (1521 by default).
  - Oracle Autonomous Database - Check your access control list (ACL).
  - Oracle Cloud Infrastructure Database - Check the network security group that was assigned to the database, and the security lists for the subnet on which the database was created.
- You selected an Oracle Cloud Infrastructure Database running Oracle Database 12c or later, and you did not provide the name of a pluggable database (PDB).

Invalid or overlapping network CIDR

Stack provisioning fails if you specify subnets with overlapping CIDRs or use the same subnet for WebLogic Server and the load balancer.

Example messages:

```
Error: module.network-wls-public-subnet.oci_core_subnet.wls-subnet: 1 error(s)
occurred: oci_core_subnet.wls-subnet: Service error:InvalidParameter. The
requested CIDR 10.0.3.0/24 is invalid: subnet ocid1.subnet.oc1.iad.aaan4a with
CIDR 10.0.3.0/24 overlaps with this CIDR.. http status code: 400.
```

```
Error: module.validators.null_resource.duplicate_lb2_subnet_cidr: : invalid or
unknown key: WLSC-ERROR: Load balancer subnet 2 CIDR has to be unique value.
```

```
Error: module.validators.null_resource.duplicate_wls_subnet_cidr: : invalid or
unknown key: WLSC-ERROR: Weblogic subnet CIDR has to be unique value.
```

Possible causes for these errors include:

- You chose to create new subnets for WebLogic Server, the load balancer, or the bastion, and the CIDR you specified for these subnets overlaps with the CIDRs for existing subnets in the same virtual cloud network (VCN).
- You chose to use an existing subnet when provisioning a stack with a load balancer, and you specified the same subnet for WebLogic Server and the load balancer.
- You created a JRF-enabled domain, your Oracle Cloud Infrastructure Database and WebLogic domain are in different VCNs, and the VCNs have overlapping CIDRs. For

example, you cannot create a WebLogic domain on VCN `10.0.0.0/16` that uses a database on VCN `10.0.0.1/24`.

Job is still running or has timed out

Most stack creation jobs for Oracle WebLogic Server for OCI should complete within an hour. Some internal provisioning problems might cause the job to run indefinitely until it eventually times out after 24 hours.

After the current Apply job times out, run a new Apply job on the same stack. This will destroy any resources that were created, and then attempt to create the resources again. If the problem occurs again, contact support.

Failed to check database port is open for Exadata DB system

When you create a domain that includes Java Required Files (JRF) components, for Exadata DB systems, the database port open check does not work if the **Create DB Security List** checkbox is selected. In this case, the provisioning fails if the database subnet has more than five security lists.

So, when provisioning, deselect the **Create DB Security List** check box to avoid creating an additional security list for the database port in the VCN, and manually open the database port (1521 by default).

View the Stack Log Files

Use the Terraform job logs in Resource Manager to identify the cause of the failure.

1. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Jobs**.
2. Identify and click the job for your stack.
   - The Type is Apply.
   - The State is Failed.
   - The Stack is the name of your Oracle WebLogic Server for OCI stack.
3. From the Logs section, search the log for error messages.
   You can optionally **Download** the log files and search the files offline.
4. See the following topics for details about specific error messages.

Modify the Stack Configuration

If necessary, delete the current stack resources, modify your stack configuration, and then apply the changes.

1. Click the navigation menu ☰, select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
2. Click the name of your stack.
3. Click **Terraform Actions** and select **Destroy**.
   Wait for the destroy job to complete.
4. Click **Edit Stack**.
5. When done, click **Save Changes**.

6. Click **Terraform Actions** and select **Apply**.

Cannot Launch a Stack in Marketplace

Example message: `Unable to accept Terms of Use`

In Marketplace, you might see the message when you click **Launch Stack**, after you've selected a stack version and compartment, and checked the Oracle Standard Terms and Restrictions box.

You likely don't have permission to:

- Create Marketplace applications in the selected compartment. Verify that this policy exists in the compartment where you want to create the stack.
  `Allow group Your_Group to manage app-catalog-listing in compartment Your_Compartment`
- Access the selected compartment. Choose another compartment or ask your administrator.

Cannot Determine Home Region

Example message:

```
data.oci_core_app_catalog_subscriptions.mp_image_subscription[0]: Refreshing
state...
Error: Null value found in list ... "oci_identity_regions" "home-region"
```

If you are not an administrator, ask them to verify that the following root-level policy exists in your tenancy:

```
Allow group Your_Group to inspect tenancies in tenancy
```

Cannot Find Dynamic Group and Secrets Policy

Example messages:

```
Error: Service error:NotAuthorizedOrNotFound. Authorization failed or
requested resource not found. http status code: 404.
 Opc request id: request_id on modules/policies/groups.tf line 8, in
resource...
 "oci_identity_dynamic_group" "wlsc_instance_principal_group" {


Error: Service error:NotAuthorizedOrNotFound. Authorization failed or
requested resource not found. http status code: 404.
 Opc request id: request_id on wlsc-policies.tf line 10, in resource...
 "oci_identity_policy" "wlsc_secret-service-policy" {
```

When the **OCI Policies** check box is selected (by default), Oracle WebLogic Server for OCI creates a dynamic group and one or more root-level policies in your tenancy.

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level permissions to create domains. If you are not an administrator, ask them to verify that root-level policies exist in your tenancy. For example:

```
Allow group Your_Group to manage dynamic-groups in tenancy
Allow group Your_Group to manage policies in tenancy
Allow group Your_Group to use secret-family in tenancy
```

See:

- Create Root Policies
- Create Policies for the Dynamic Group

Maximum Number of Dynamic Groups Has Exceeded

Example message:

```
<WLSC-VM-ERROR-0119> : Failed to get secret content for
[ocid1.vaultsecret.oc1.iad.alongstring123]: [{'status': 400, 'message': "This
instance principal matches more than '5' dynamic groups, update your dynamic
groups' matching rules"...'}]>
```

When the **OCI Policies** check box is selected (by default), Oracle WebLogic Server for OCI creates a dynamic group and one or more root-level policies in your tenancy. The maximum number of dynamic groups allowed is 5.

Solution:

1. Add the following policy that uses the existing dynamic group to access the new secrets for the new stack:

   ```
   Allow dynamic-group <existing-dyanmic-group-name> to read secret-bundles
   in tenancy where target.secret.id = '<OCID_of_the_secret>'
   ```

2. Deselect the **OCI Policies** check box and try to create the stack again.

Unable to Get Secret Content or Decrypted Credential

Example messages:

- `Failed to get secret content for Your_vault_secret_OCID`
- `Authorization failed or requested resource not found`
- `Error retrieving %s password from Secret Vault`
- `Failed in create domain due to exception [Failed to retrieve WebLogic Password from Secrets Vault]`
- `Failed to retrieve IDCS Client Secret from Secrets Vault`
- `Unable to get decrypt credential`
- `Key or Vault does not exist or you are not authorized to access them.`

When you create a domain with Oracle WebLogic Server for OCI, you provide the OCID values of the secrets that contain the passwords to use for the domain and during provisioning. The compute instances use this information to decrypt the passwords. The compute instances are granted access to vault secrets using policies.

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level permissions to create domains. If you are not an administrator, ask them to verify that relevant vault secret policies exist in your tenancy and compartment. For example:

```
Allow group Your_Group to use secret-family in tenancy
Allow dynamic-group Your_DynamicGroup to use secret-family in compartment
MyCompartment
Allow dynamic-group Your_DynamicGroup to use keys in compartment MyCompartment
Allow dynamic-group Your_DynamicGroup to use vaults in compartment
MyCompartment
```

If the policies exist, check that the OCID of the compartment in listed in dynamic group.

See:

- Create Secrets for Passwords
- Create Policies for the Dynamic Group
- Create Root Policies

Unable to Get Decrypted Credential When Creating a Stack in a Private Subnet

Example message: `<WLSC-VM-ERROR-001> Unable to get decrypt credential [HTTPSConnectionPool(host='auth.us-phoenix-1.oraclecloud.com', port=443): Max retries exceeded with url: /v1/x509 (Caused by ConnectTimeoutError(<oci._vendor.urllib3.connection.VerifiedHTTPSConnection object at 0x1e5110>, 'Connection to auth.us-phoenix-1.oraclecloud.com timed out. (connect timeout=10)'))]>`

When you create a domain with Oracle WebLogic Server for OCI in an existing private subnet, provisioning fails if the WebLogic Server subnet is using a route table that does not include a service gateway or a Network Address Translation (NAT) gateway.

Modify the private subnet, and select a route table that uses a service gateway or NAT gateway. Or select a virtual cloud network (VCN) whose default route table uses a service gateway or NAT gateway. Refer to these topics:

- VCNs and Subnets
- Access to Oracle Services: Service Gateway

Failed to Download Oracle Autonomous Database Wallet

Example message: `module.provisioners.null_resource.status_check[0] (remote-exec): <Nov 23, 2019 09:37:17 PM GMT> <ERROR> <oci_api_utils> <(host:stackname-wls-0.subnetxxx.stacknamevcn.oraclevcn.com) - <WLSC-VM-ERROR-0052> : Unable to download atp wallet. [{'status': 403, 'message': u'Forbidden', 'code': u'Forbidden', 'opc-request-id': 'FA6C16D8B'}]`

You must be an Oracle Cloud Infrastructure administrator, or be granted root-level and compartment-level permissions to create domains. Access to the database wallet is needed when you create a JRF-enabled domain that uses an autonomous database. If you are not an administrator, ask them to verify that relevant policies for autonomous databases exist in your tenancy and compartment. For example:

```
Allow group Your_Group to inspect autonomous-transaction-processing-family in
compartment Your_ATP_Compartment
Allow dynamic-group Your_DynamicGroup to inspect autonomous-transaction-
processing-family in compartment Your_ATP_Compartment
```

See:

- Create Policies for the Dynamic Group
- Create Root Policies
- Create Compartment Policies

Failed to Validate DB Connectivity

When you create a domain that includes the Java Required Files (JRF) components, you must select an existing database and provide connection details. The compute instances use this information to connect to the database and provision the JRF database schemas.

Possible causes for this error include:

- You entered the wrong database password or a plain text password.
- The database does not allow the compute instances to access its listen port (1521 by default).
  - Oracle Autonomous Database - Check your access control list (ACL).
  - Oracle Cloud Infrastructure Database - Check the network security group that was assigned to the database, and the security lists for the subnet on which the database was created.
- You selected an Oracle Cloud Infrastructure Database running Oracle Database 12c or later, and you did not provide the name of a pluggable database (PDB).

Invalid or Overlapping Network CIDR

Stack provisioning fails if you specify subnets with overlapping CIDRs or use the same subnet for WebLogic Server and the load balancer.

Example messages:

```
Error: module.network-wls-public-subnet.oci_core_subnet.wls-subnet: 1 error(s)
occurred: oci_core_subnet.wls-subnet: Service error:InvalidParameter. The
requested CIDR 10.0.3.0/24 is invalid: subnet ocid1.subnet.oc1.iad.aaan4a with
CIDR 10.0.3.0/24 overlaps with this CIDR.. http status code: 400.
```

```
Error: module.validators.null_resource.duplicate_lb2_subnet_cidr: : invalid or
unknown key: WLSC-ERROR: Load balancer subnet 2 CIDR has to be unique value.
```

```
Error: module.validators.null_resource.duplicate_wls_subnet_cidr: : invalid or
unknown key: WLSC-ERROR: Weblogic subnet CIDR has to be unique value.
```

Possible causes for these errors include:

- You chose to create new subnets for WebLogic Server, the load balancer, or the bastion, and the CIDR you specified for these subnets overlaps with the CIDRs for existing subnets in the same virtual cloud network (VCN).
- You chose to use an existing subnet when provisioning a stack with a load balancer, and you specified the same subnet for WebLogic Server and the load balancer.
- You created a JRF-enabled domain, your Oracle Cloud Infrastructure Database and WebLogic domain are in different VCNs, and the VCNs have overlapping CIDRs. For example, you cannot create a WebLogic domain on VCN `10.0.0.0/16` that uses a database on VCN `10.0.0.1/24`.

Job Is Still Running or Has Timed Out

Most stack creation jobs for Oracle WebLogic Server for OCI should complete within an hour. Some internal provisioning problems might cause the job to run indefinitely until it eventually times out after 24 hours.

After the current Apply job times out, run a new Apply job on the same stack. This will destroy any resources that were created, and then attempt to create the resources again. If the problem occurs again, contact support.

Failed to Check Database Port Is Open for Exadata DB System

When you create a domain that includes Java Required Files (JRF) components, for Exadata DB systems, the database port open check does not work if the **Create DB Security List** checkbox is selected. In this case, the provisioning fails if the database subnet has more than five security lists.

So, when provisioning, deselect the **Create DB Security List** check box to avoid creating an additional security list for the database port in the VCN, and manually open the database port (1521 by default).

8 Patches

Each Oracle WebLogic Server for OCI release includes patches from several products, namely, Oracle WebLogic Server, Oracle JDeveloper, Oracle Java Development Kit, Oracle Platform Security Services and Oracle Web Services Manager.

> **Note:**
>
> If a `FUSER could not be located` error is displayed when applying a patch, complete the following steps:
>
> 1. Set the environment variable: `export OPATCH_NO_FUSER=TRUE`.
> 2. In the Unix shell where step 1 is run or `OPATCH_NO_FUSER=TRUE` is set, apply the required patch.

Patches in a new release of Oracle WebLogic Server for OCI are not automatically applied to existing domains that you created with Oracle WebLogic Server for OCI. You have to apply the patches manually if you wish to update your existing domain to match the latest release, or to match a specific supported release.

A Patch Set Update (PSU) is a group of related patches that is identified by a specific version number. When you create a domain with Oracle WebLogic Server for OCI, you choose a version of WebLogic Server in this format: `<major_version>.<patch_level>.<build>`. For example, `12.2.1.4.191121.01`.

> **Tip:**
>
> For a list of new features and enhancements that were added recently to improve your Oracle WebLogic Server for OCI experience, see What's New for Oracle WebLogic Server for OCI.

OS Patching

For your operating system, you can apply patches using `yum` or `dnf`, or by utilizing the OCI OS Management Hub (OMH). For more information, see:

- Listing Instances in the Oracle Cloud Infrastructure documentation for details on how to access the registered managed instances created by the stack and perform tasks such as updating OS software packages.
- Installing Individual Updates on an Instance for how to keep software packages up to date.

If you are unable to locate an OS Management Hub Managed Instance for your Compute instance see Enable OS Management Hub After Creating a Domain.

JDK Patching

To patch JDK, download and install JDK from Oracle Technology Network to the `/u01/app/oracle/jdk` location. Do not use `/u01/jdk` location as this location is symlink to `/u01/app/oracle/jdk`.

Patches for Oracle WebLogic Server for OCI

Table 8-1 lists the patches that are found in the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases. Use your Oracle Support account to locate and download the patch you wish to apply.

> **Note:**
>
> For a list of patches that are found in the Oracle WebLogic Server for OCI 11.1.1.7.0 and 12.2.1.3 releases, see Table 8-2.

Table 8-1 Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
14.1.2.0.250324	opatch: • 37743913 - WLS PATCH SET UPDATE 14.1.2.0.250324 • 37658370 - Coherence 14.1.2.0 Cumulative Patch 1 (14.1.2.0.2) • 37606570 - RDA 25.2-2025415 for FMW/WLS 14.1.2 • 37635555 - WEBLOGIC SAMPLES SPU 14.1.2.0.250415 • 37751141 - OWSM BUNDLE PATCH 14.1.2.0.250325 • 37632474 - UMS BUNDLE PATCH 14.1.2.0.250225 opatch version: • 28186730: OPATCH 13.9.4.2.19 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.2.1	April 2025
14.1.1.0.250415	opatch: • 37638712 - WEBLOGIC SAMPLES SPU 14.1.1.0.250415 • 36723262 - OSDT BP FOR WLS 14.1.1.0.0 • 37258703 - JDBC19.25 BUNDLE PATCH 14.1.1.0.241108 • 37658340 - Coherence 14.1.1.0 Cumulative Patch 21 (14.1.1.0.21) • 37506489 - RDA release 25.2-2025415 for FMW 14.1.1.0.0 • 37140343 - OCT 2024 CLONING SPU FOR WLS 14.1.1.0.0 • 37732915 - WLS PATCH SET UPDATE 14.1.1.0.250320 • 35965633 - ADR FOR WEBLOGIC SERVER 14.1.1.0 – SIZE OPTIMIZED FOR JAN 2024 opatch version: • 28186730: OPATCH 13.9.4.2.19 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.2.1	April 2025

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.240416	opatch: • 36426672 - WEBLOGIC SAMPLES SPU 12.2.1.4.240416 • 37258699 - JDBC19.25 BUNDLE PATCH 12.2.1.4.241107. • 36316422 - OPSS Bundle Patch 12.2.1.4.240220. • 34065178 - MERGE REQUEST ON TOP OF 12.2.1.4.0 FOR BUGS 34010500 33903365 • 36789759 - FMW PLATFORM BUNDLE PATCH 12.2.1.4.240812 • 37506516 - RDA release 25.2-2025415 for FMW 12.2.1.4.0 • 36946553 - FMW CONTROL BUNDLE PATCH 12.2.1.4.240814 • 37284722 - WebCenter Core Bundle Patch 12.2.1.4.241114 • 37836334 - ADF BUNDLE PATCH 12.2.1.4.250416 • 34809489 - PS4 : PATCH CONFLICT OBSERVED BETWEEN 34647149 AND 34604561 • 37684007 - OWSM BUNDLE PATCH 12.2.1.4.250309 • 37658278 - Coherence 12.2.1.4 Cumulative Patch 24 (12.2.1.4.25) • 37056593 - OCT 2024 CLONING SPU FOR FMW 12.2.1.4.0 • 37714186 - WLS PATCH SET UPDATE 12.2.1.4.250317 • 37297691 - OSS 19C BUNDLE PATCH 12.2.1.4.241119 • 37710654 - FMW Thirdparty Bundle Patch 12.2.1.4.250314 • 35965629 - ADR FOR WEBLOGIC SERVER 12.2.1.4.0 - SIZE OPTIMIZED FOR JAN 2024 opatch version: • 28186730 - OPATCH 13.9.4.2.19 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.2.1	April 2025
14.1.2.0.250121.02	opatch: • Patch 37439198: WLS PATCH SET UPDATE 14.1.2.0.250102 • Patch 37351934: Coherence Cumulative Patch 14.1.2.0.1 • Patch 28186730: OPATCH 13.9.4.2.18 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.1.1	January 2025 PSUs
14.1.1.0.250121 .03	opatch: • Patch 37458537: WLS PATCH SET UPDATE 14.1.1.0.250108 • Patch 37351880: Coherence 14.1.1.0 Cumulative Patch 20 (14.1.1.0.20) • Patch 37258703: JDBC19.25 BUNDLE PATCH 14.1.1.0.241108 • Patch 37202241: RDA release 25.1-2025121 for FMW 14.1.1.0.0 • Patch 37140343: OCT 2024 CLONING SPU FOR WLS 14.1.1.0.0 • Patch 36723262: OSDT BP FOR WLS 14.1.1.0.0 • Patch 35965633: ADR FOR WEBLOGIC SERVER 14.1.1.0 ? SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) opatch version: • Patch 28186730: OPATCH 13.9.4.2.18 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.1.1	January 2025 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.250121.03	opatch: • Patch 37453807: WLS PATCH SET UPDATE 12.2.1.4.250107 • Patch 37351860: Coherence 12.2.1.4 Cumulative Patch 24 (12.2.1.4.24) • Patch 37388935: ADF BUNDLE PATCH 12.2.1.4.241212 • Patch 37374672: FMW Thirdparty Bundle Patch 12.2.1.4.241210 • Patch 37202255: RDA release 25.1-2025121 for FMW 12.2.1.4.0 • Patch 37258699: JDBC19.25 BUNDLE PATCH 12.2.1.4.241107 • Patch 37297691: OSS 19C BUNDLE PATCH 12.2.1.4.241119 • Patch 37284722: WebCenter Core Bundle Patch 12.2.1.4.241114 • Patch 36789759: FMW PLATFORM BUNDLE PATCH 12.2.1.4.240812 • Patch 37035947: OWSM BUNDLE PATCH 12.2.1.4.240908 • Patch 35965629: ADR FOR WEBLOGIC SERVER 12.2.1.4.0 - SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) • Patch 36316422: OPSS Bundle Patch 12.2.1.4.240220 • Patch 37056593: OCT 2024 CLONING SPU FOR FMW 12.2.1.4.0 • Patch 36946553: FMWCONTROL BUNDLE PATCH 12.2.1.4.240814 • Patch 34065178: MERGE REQUEST ON TOP OF 12.2.1.4.0 FOR BUGS 34010500 33903365 opatch version: • Patch 28186730: OPATCH 13.9.4.2.18 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0, 14.1.1.0.0, 14.1.2.0.0	25.1.1	January 2025 PSUs
14.1.1.0.241015.02	opatch: • 37140343: OCT 2024 CLONING SPU FOR WLS 14.1.1.0.0 • 37087534: WLS PATCH SET UPDATE 14.1.1.0.240922 • 37024687: JDBC19.24 BUNDLE PATCH 14.1.1.0.240904 • 37049917: Coherence 14.1.1.0 Cumulative Patch 19 (14.1.1.0.19) • 36851307: RDA release 24.4-20241015 for FMW 14.1.1.0.0 • 36723262: OSDT BP FOR WLS 14.1.1.0.0 • 35965633: ADR FOR WEBLOGIC SERVER 14.1.1.0 ? SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) opatch version: • 28186730: OPATCH 13.9.4.2.16 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0 AND 14.1.1.0.0	24.4.3	October 2024 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.241015.02	opatch: • 37087476: WLS PATCH SET UPDATE 12.2.1.4.240922 • 37049907: Coherence 12.2.1.4 Cumulative Patch 23 (12.2.1.4.23) • 37028738: ADF BUNDLE PATCH 12.2.1.4.240905 • 37103277: FMW Thirdparty Bundle Patch 12.2.1.4.240925 • 36789759: FMW PLATFORM BUNDLE PATCH 12.2.1.4.240812 • 37035947: OWSM BUNDLE PATCH 12.2.1.4.240908 • 36851321: RDA release 24.4-20241015 for FMW 12.2.1.4.0 • 36926636: JDBC 19.24 BUNDLE PATCH 12.2.1.4.240814 • 37096063: OSS 19C BUNDLE PATCH 12.2.1.4.241001 (Linux x86-64) • 37056593: OCT 2024 CLONING SPU FOR FMW 12.2.1.4.0 • 36946553: FMWCONTROL BUNDLE PATCH 12.2.1.4.240814 • 35965629: ADR FOR WEBLOGIC SERVER 12.2.1.4.0 - SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) • 36316422: OPSS Bundle Patch 12.2.1.4.240220 • 36964687: WEBCENTER CORE BP 12.2.1.4.240819 (Latest Recommended) • 34065178: Required FMW Compatibility Patch for JDK 8u331 (or later) opatch version: • 28186730: OPATCH 13.9.4.2.17 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0 AND 14.1.1.0.0	24.4.3	October 2024 PSUs
14.1.1.0.241015.01	opatch: • 37140343: OCT 2024 CLONING SPU FOR WLS 14.1.1.0.0 • 37087534: WLS PATCH SET UPDATE 14.1.1.0.240922 • 37024687: JDBC19.24 BUNDLE PATCH 14.1.1.0.240904 • 37049917: Coherence 14.1.1.0 Cumulative Patch 19 (14.1.1.0.19) • 36851307: RDA release 24.4-20241015 for FMW 14.1.1.0.0 • 36723262: OSDT BP FOR WLS 14.1.1.0.0 • 35965633: ADR FOR WEBLOGIC SERVER 14.1.1.0 ? SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) opatch version: • 28186730: OPATCH 13.9.4.2.16 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0 AND 14.1.1.0.0	24.4.1	October 2024 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.241015.01	opatch: • 37087476: WLS PATCH SET UPDATE 12.2.1.4.240922 • 37049907: Coherence 12.2.1.4 Cumulative Patch 23 (12.2.1.4.23) • 37028738: ADF BUNDLE PATCH 12.2.1.4.240905 • 37103277: FMW Thirdparty Bundle Patch 12.2.1.4.240925 • 36789759: FMW PLATFORM BUNDLE PATCH 12.2.1.4.240812 • 37035947: OWSM BUNDLE PATCH 12.2.1.4.240908 • 36851321: RDA release 24.4-20241015 for FMW 12.2.1.4.0 • 36926636: JDBC 19.24 BUNDLE PATCH 12.2.1.4.240814 • 37096063: OSS 19C BUNDLE PATCH 12.2.1.4.241001 (Linux x86-64) • 37056593: OCT 2024 CLONING SPU FOR FMW 12.2.1.4.0 • 36946553: FMWCONTROL BUNDLE PATCH 12.2.1.4.240814 • 35965629: ADR FOR WEBLOGIC SERVER 12.2.1.4.0 - SIZE OPTIMIZED FOR JAN 2024 (Linux x86-64) • 36316422: OPSS Bundle Patch 12.2.1.4.240220 • 36964687: WEBCENTER CORE BP 12.2.1.4.240819 (Latest Recommended) • 34065178: Required FMW Compatibility Patch for JDK 8u331 (or later) opatch version: • 28186730: OPATCH 13.9.4.2.17 FOR EM 13.5 AND FMW/WLS 12.2.1.4.0 AND 14.1.1.0.0	24.4.1	October 2024 PSUs
14.1.1.0.240716.01	opatch: • 36725968 - Coherence 14.1.1.0 Cumulative Patch 18 (14.1.1.0.18) • 36781850 - WLS Patch Set Update 14.1.1.0.240628 opatch version: • 28186730 - Opatch 13.9.4.2.16 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	24.3.1	July 2024 PSUs
12.2.1.4.240716.01	opatch: • 36725924 - Coherence 12.2.1.4 Cumulative Patch 22 (12.2.1.4.22) • 36805124 - WLS Patch Set Update 12.2.1.4.240704 • 36700543 - ADF Bundle Patch 12.2.1.4.240605 • 36770738 - FMW Thirdparty Bundle Patch 12.2.1.4.240625 • 36769312 - OWSM Bundle 12.2.1.4.240625 opatch version: • 28186730 - Opatch 13.9.4.2.16 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	24.3.1	July 2024 PSUs
14.1.1.0.240416.02	opatch: • 36410357 - Coherence 14.1.1.0 Cumulative Patch 17 (14.1.1.0.17) • 36454290 - WLS patch set update 14.1.1.0.240328 opatch version: • 28186730 - Opatch 13.9.4.2.15 for EM 13.5 and FMW/WLS 12.2.1.4.0 and 14.1.1.0.0	24.2.1	April 2024 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
14.1.1.0.240328	opatch: • 36410357 - Coherence 14.1.1.0 Cumulative Patch 17 (14.1.1.0.17) • 36454290 - WLS patch set update 14.1.1.0.240328 opatch version: • 28186730 - Opatch 13.9.4.2.15 for EM 13.5 and FMW/WLS 12.2.1.4.0 and 14.1.1.0.0	24.2.1	April 2024 PSUs
12.2.1.4.240401	opatch: • 36348444 - ADF BUNDLE PATCH 12.2.1.4.240228 • 36410345 - Coherence 12.2.1.4 Cumulative Patch 21 (12.2.1.4.21) • 36402397 - OWSM BUNDLE PATCH 12.2.1.4.240313 • 36468190 - FMW Thirdparty Bundle Patch 12.2.1.4.240401 • 36440005 - WLS patch set update 12.2.1.4.240325 • 36316422 - OPSS Bundle Patch 12.2.1.4.240220 opatch version: • 28186730 - Opatch 13.9.4.2.15 for EM 13.5 and FMW/WLS 12.2.1.4.0 and 14.1.1.0.0	24.2.1	April 2024 PSUs
14.1.1.0.240116.01	• 36068072 - Coherence 14.1.1.0 Cumulative Patch 16 (14.1.1.0.16) • 36124787 - WLS Patch Set Update 14.1.1.0.231220 • 35965633 - ADR for Weblogic Server 14.1.1.0 Size Optimized for Jan 2024 opatch version: • 28186730 - Opatch 13.9.4.2.14 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	24.1.1	January 2024 PSUs
12.2.1.4.240116.01	opatch: • 36068046 - Coherence 12.2.1.4 Cumulative Patch 20 (12.2.1.4.20) • 36155700 - WLS Patch Set Update 12.2.1.4.240104 • 36074941 - ADF Bundle Patch 12.2.1.4.231205 • 36086980 - FMW Thirdparty Bundle Patch 12.2.1.4.231207 • 35965629 - ADR for Weblogic Server 12.2.1.4.0 Size Optimized for Jan 2024 • 35868571 - OWSM Bundle Patch 12.2.1.4.231003 opatch version: • 28186730 - Opatch 13.9.4.2.14 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	24.1.1	January 2024 PSUs
14.1.1.0.231017.01	opatch: • 35778872 - Coherence 14.1.1.0 Cumulative Patch 15 (14.1.1.0.15) • 35904051 - WLS Patch Set Update 14.1.1.0.231012 • 35476075 - ADR for Weblogic Server 14.1.1.0 CPU OCT 2023 opatch version: • 28186730 - Opatch 13.9.4.2.14 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.4.1	October 2023 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.231017.01	opatch: • 35778804 - Coherence 12.2.1.4 Cumulative Patch 19 (12.2.1.4.19) • 35893811 - WLS Patch Set Update 12.2.1.4.231010 • 35735469 - ADF Bundle Patch 12.2.1.4.230823 • 35882299 - FMW Thirdparty Bundle Patch 12.2.1.4.231006 • 35476067 - ADR for Weblogic Server 12.2.1.4.0 CPU OCT 2023 • 34302154 - oracle.security.restsec.jwt.JwtToken has reference to jackson 1.x opatch version: • 28186730 - Opatch 13.9.4.2.14 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.4.1	October 2023 PSUs
14.1.1.0.230718.04	opatch: • 35505236 - Coherence 14.1.1.0 Cumulative Patch 14 (14.1.1.0.14) • 35560771 - WLS Patch Set Update 14.1.1.0.230703 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.3	July 2023 PSUs
12.2.1.4.230718.04	opatch: • 35505207 - Coherence 12.2.1.4 Cumulative Patch 18 (12.2.1.4.18) • 35557681 - WLS Patch Set Update 12.2.1.4.230702 • 35503128 - ADF Bundle Patch 12.2.1.4.230615 • 35547646 - FMW Thirdparty Bundle Patch 12.2.1.4.230628 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.3	July 2023 PSUs
14.1.1.0.230718.03	opatch: • 35505236 - Coherence 14.1.1.0 Cumulative Patch 14 (14.1.1.0.14) • 35560771 - WLS Patch Set Update 14.1.1.0.230703 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.2	July 2023 PSUs
12.2.1.4.230718.03	opatch: • 35505207 - Coherence 12.2.1.4 Cumulative Patch 18 (12.2.1.4.18) • 35557681 - WLS Patch Set Update 12.2.1.4.230702 • 35503128 - ADF Bundle Patch 12.2.1.4.230615 • 35547646 - FMW Thirdparty Bundle Patch 12.2.1.4.230628 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.2	July 2023 PSUs
14.1.1.0.230718.02	opatch: • 35505236 - Coherence 14.1.1.0 Cumulative Patch 14 (14.1.1.0.14) • 35560771 - WLS Patch Set Update 14.1.1.0.230703 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.1	July 2023 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.230718.02	opatch: • 35505207 - Coherence 12.2.1.4 Cumulative Patch 18 (12.2.1.4.18) • 35557681 - WLS Patch Set Update 12.2.1.4.230702 • 35503128 - ADF Bundle Patch 12.2.1.4.230615 • 35547646 - FMW Thirdparty Bundle Patch 12.2.1.4.230628 opatch version: • 28186730 - Opatch 13.9.4.2.13 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0	23.3.1	July 2023 PSUs
14.1.1.0.230418.04	opatch: • 35227385 - WLS patch set update 14.1.1.0.230328 • 35122412 - Coherence 14.1.1.0 Cumulative Patch 13 (14.1.1.0.13) opatch version: • 28186730 - Opatch 13.9.4.2.12 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.2.3	April 2023 PSUs
12.2.1.4.230418.04	opatch: • 35226999 - WLS patch set update 12.2.1.4.230328 • 35122398 - Coherence 12.2.1.4 Cumulative Patch 17 (12.2.1.4.17) • 35159582 - OWSM Bundle Patch 12.2.1.4.230308 • 35162846 - FMW Third party Bundle Patch 12.2.1.4.230309 • 35148842 - ADF Bundle Patch 12.2.1.4.230306 • 33093748 - FMW Platform 12.2.1.4.0 SPU for APR CPU 2021 • 30385564 - Oracle XML Developers Kit patch • 33950717 - OPSS Bundle Patch 12.2.1.4.220311 • 31544353 - ADR for WebLogic Server 12.2.1.4.0 July CPU 2020 • 33903365 - Patch for OAM Console login fails after applying 1.80.331 JDK • 34542329 - Merge request on top of 12.2.1.4.0 for bugs 34280277, 26354548, 26629487, 29762601 opatch version: • 28186730 - Opatch 13.9.4.2.12 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.2.3	April 2023 PSUs
14.1.1.0.230418.03	opatch: • 35227385 - WLS patch set update 14.1.1.0.230328 • 35122412 - Coherence 14.1.1.0 Cumulative Patch 13 (14.1.1.0.13) opatch version: • 28186730 - Opatch 13.9.4.2.12 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.2.2	April 2023 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.230418.03	opatch: • 35226999 - WLS patch set update 12.2.1.4.230328 • 35122398 - Coherence 12.2.1.4 Cumulative Patch 17 (12.2.1.4.17) • 35159582 - OWSM Bundle Patch 12.2.1.4.230308 • 35162846 - FMW Third party Bundle Patch 12.2.1.4.230309 • 35148842 - ADF Bundle Patch 12.2.1.4.230306 • 33093748 - FMW Platform 12.2.1.4.0 SPU for APR CPU 2021 • 30385564 - Oracle XML Developers Kit patch • 33950717 - OPSS Bundle Patch 12.2.1.4.220311 • 31544353 - ADR for WebLogic Server 12.2.1.4.0 July CPU 2020 • 33903365 - Patch for OAM Console login fails after applying 1.80.331 JDK • 34542329 - Merge request on top of 12.2.1.4.0 for bugs 34280277, 26354548, 26629487, 29762601 opatch version: • 28186730 - Opatch 13.9.4.2.12 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.2.2	April 2023 PSUs
14.1.1.0.230117.04	opatch: • 34890864 - WLS patch set update 14.1.1.0.221213 • 34845949 - Coherence 14.1.1.0 Cumulative Patch 12 (14.1.1.0.12) opatch version: • 28186730 - Opatch 13.9.4.2.11 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.1.3	January 2023 PSUs
12.2.1.4.230117.04	opatch: • 34883826 - WLS patch set update 12.2.1.4.221210 • 34845927 - Coherence 12.2.1.4 Cumulative Patch 16 (12.2.1.4.16) • 34839859 - OWSM Bundle Patch 12.2.1.4.221128 • 34879707 - FMW Third party Bundle Patch 12.2.1.4.221209 • 34944256 - ADF Bundle Patch 12.2.1.4.230103 • 33093748 - FMW Platform 12.2.1.4.0 SPU for APR CPU 2021 • 30385564 - Oracle XML Developers Kit patch • 33950717 - OPSS Bundle Patch 12.2.1.4.220311 • 31544353 - ADR for WebLogic Server 12.2.1.4.0 July CPU 2020 • 33903365 - Patch for OAM Console login fails after applying 1.80.331 JDK • 34542329 - Merge request on top of 12.2.1.4.0 for bugs 34280277, 26354548, 26629487, 29762601 opatch version: • 28186730 - Opatch 13.9.4.2.11 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	23.1.3	January 2023 PSUs
14.1.1.0.221018.02	opatch: • 34686388 - WLS patch set update 14.1.1.0.221010 • 34545599 - Coherence 14.1.1.0 Cumulative Patch 11 (14.1.1.0.11) New opatch version: • 28186730 - Opatch 13.9.4.2.11 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	22.4.3	October 2022 PSUs

Table 8-1 (Cont.) Patches for the Oracle WebLogic Server for OCI 14.1.2.0.0, 14.1.1.0 and 12.2.1.4 releases

WebLogic Server Version	Patches	WebLogic Server for OCI Release	Patch List
12.2.1.4.221018.02	opatch: • 34566592 - OWSM Bundle Patch 12.2.1.4.220905 • 34604561 - FMW Third party Bundle Patch 12.2.1.4.220915 • 34545596 - Coherence 12.2.1.4 Cumulative Patch 15 (12.2.1.4.15) • 34535558 - ADF Bundle Patch 12.2.1.4.220825 • 34653267 - WLS patch set update 12.2.1.4.220929 • 34542329 - Merge request on top of 12.2.1.4.0 for bugs 34280277, 26354548, 26629487, 29762601 • 33093748 - FMW Platform 12.2.1.4.0 SPU for APR CPU 2021 • 30385564 - Oracle XML Developers Kit patch • 33950717 - OPSS Bundle Patch 12.2.1.4.220311 • 31544353 - ADR for WebLogic Server 12.2.1.4.0 July CPU 2020 • 33903365 - Patch for OAM Console login fails after applying 1.80.331 JDK New opatch version: • 28186730 - Opatch 13.9.4.2.11 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0	22.4.3	October 2022 PSUs

The following table lists the patches for 11.1.1.7.0 and 12.2.1.3 releases. Use your Oracle Support account to locate and download the patch you wish to apply.

Table 8-2   Patches for 11.1.1.7.0 and 12.2.1.3 releases

WebLogic Server Version	Patches
12.2.1.3	opatch: • 35247514 - WLS patch set update 12.2.1.3.230402 • 35122395 - Coherence 12.2.1.3 Cumulative Patch 22 (12.2.1.3.22) • 34883781 - WLS patch set update 12.2.1.3.221210 • 34845919 - Coherence 12.2.1.3 Cumulative Patch 21 (12.2.1.3.21) • 28186730 - Opatch 13.9.4.2.11 for EM 13.4, 13.5 and for FMW/WLS 12.2.1.3.0, 12.2.1.4.0, and 14.1.1.0.0 • 34697822 - WLS patch set update 12.2.1.3.221013 • 34545595 - Coherence 12.2.1.3 Cumulative Patch 20 (12.2.1.3.20) • 34636492 - FMW Third party Bundle Patch 12.2.1.3.220926 • 34667652 - OWSM Bundle Patch 12.2.1.3.221004 • 32982708 - FMW Platform 12.2.1.3.0 SPU for APR CPU 2021 • 34243945 - ADF Bundle Patch 12.2.1.3.220604 • 33903365 - Patch for OAM console login fails after applying 1.80.331 JDK • 1221318 (33902200) - Coherence 12.2.1.3 Cumulative Patch 18 (12.2.1.3.18) • 33949366 - ADF bundle patch 12.2.1.3.220310 • 34010914 - WebLogic patch set update 12.2.1.3.220329 • 33791665 - log4j v1 - CVE-2021-4104, CVE-2022-23302, CVE-2022-23305, CVE-2022-23307 • 33735326 - log4j v2 - CVE-2021-44832 • 33699205 - WebLogic patch set update 12.2.1.3.211222 • 33618953 - OWSM Bundle Patch 12.2.1.3.211129 • 32997257 - ADF bundle patch 12.2.1.3.210614 • 33591009 - Coherence 12.2.1.3 Cumulative Patch 17 (12.2.1.3.17) • 33590225 - ADF bundle patch 12.2.1.3.211119 • 33671996 - WebLogic overlay patch for October 2021 PSU for CVE-2021-44228 and CVE-2021-45046 • 32772477 - FMW Platform 12.2.1.3.0 SPU for April 2021 CPU • 32651962 - FMW common third-party SPU 12.2.1.3.0 for April 2021 CPU • 26394536 - Oracle Process Mgmt and Notification patch • 31544340 - ADR for WebLogic JULY CPU 2020 • 30186876 - SOA composer patch • 30252137 - XML Developers Kit patch • 27263211 - Enterprise Manager patch • 31464643 - Merge request on top of 12.2.1.3.0 for bugs 29011959 and 30385564 • 1221316 (33286132) - Coherence 12.2.1.3 cumulative patch 16 (12.2.1.3.16) • 33313934 - ADF Bundle Patch 12.2.1.3.210903 • 33412599 - WebLogic patch set update 12.2.1.3.210929 • 29840258 - RCU patch • 28659321 - Managed Server logs patch • 32397127 - OPSS patch for April 2021 • 26045997 - JDBC patch • 27608287 - Upgrade Assistant Readiness patch • 24738720 - Enterprise Manager patch • 32917014 - OWSM bundle patch 12.2.1.3.210524 • 18345580 - XML Developers Kit patch • 26355633 - JDBC patch • 26287183 - JDBC patch • 26261906 - Universal Connection Pool patch

Table 8-2 (Cont.) Patches for 11.1.1.7.0 and 12.2.1.3 releases

WebLogic Server Version	Patches
	• 26051289 - JDBC patch
10.3.6 (11.1.1.7.0)	bsu: • 21Y4 - WebLogic patch set update 10.3.6.0.211019 • CW7X - ADR for WebLogic Server • I1EV - WebLogic Server patch • SY38 - WebLogic Server patch • TTGM - Patch to update certgen and related artifacts to support new demo certs. opatch: • 27214515 - ADF patch • 27846936 - OPSS bundle patch • 17617649 - FMW bug fixes • 22577934 - FMW Control patch • 22852289 - FMW bug fixes

To identify the version of WebLogic Server on which your domain is running:

1. Sign in to the WebLogic Server Administration Console for your domain. See Access the WebLogic Server Administration Console.
2. From the Domain Structure panel, expand **Environments**, and then click **Servers**.
3. Click the administration server.
4. Click **Monitoring**.
5. Locate the **Patch List**.

Topics

- About the Patching Utility Tool
- Patch Management Using the Patching Utility Tool

About the Patching Utility Tool

Oracle WebLogic Server for OCI provides the patching utility tool to download the patches for the WebLogic Server instances. This utility can be used if you do not have access to the support portal to download the required patches.

> **Note:**
>
> - The patching utility tool is available only for Oracle WebLogic Server for OCI instances provisioned after release 20.4.3 (December 23, 2020).
> - For instances provisioned prior to release 20.4.3 (before December 23, 2020), download the WebLogic Server patches by using the My Oracle Support website.
> - For UCM only license users, who does not have access to My Oracle Support, open a Support Ticket to get the unique link to download the quarterly Patch Set Updates (PSUs).

From release 21.3.3 onwards, you can use the patching utility tool to download patches for custom images that are created from Oracle WebLogic Server for OCI instance.

For custom images created from existing instances with older version of patching tool, the patching utility does not work. So, before you create custom images from existing instances, Oracle recommends you to upgrade the patching utility to the latest version using the `patch-utils upgrade` command. With the patching tool upgrade, the images use the latest version of patching utility tool with two keys for decryption.

For existing instances created from the Marketplace, you need not upgrade the patching tool. You can use the older version of the patching tool to download the patches.

You can use this patching tool utility on the WebLogic Server compute instance and the bastion instance.

Patch Management Using the Patching Utility Tool

Use the patching utility tool in Oracle WebLogic Server for OCI to list, download, apply, and roll back patches. You can also view the patching tool version and upgrade the patching tool.

> **Note:**
>
> Using this utility, you cannot apply the patch for a bastion instance.

You can perform the following tasks using the patching utility tool:

> **Note:**
>
> If you want to use some of the new features, which were added to the patching utility tool in December 15, 2022, for your existing Oracle WebLogic Server for OCI instances (created before December 15, 2022), then ensure that you upgrade the patching tool. See Upgrade Patching Tool.

- View Patching Tool Version
- Configure Initial Setup
- List Patches
- List Pending Patches
- View Patch Details
- Download Patches
- Apply Patches
- Roll Back Patches
- Upgrade Patching Tool

View Patching Tool Version

Use the patching tool to view the build version along with the Oracle license and copyright information.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Print the build version.

   ```
   patch-utils -v
   ```

   Sample output:

   ```
   Weblogic Cloud Patch-Utils <Patch version number>)
   Copyright (c) 2020, Oracle Corporation and/or its affiliates.Licensed
   under the Universal Permissive License v 1.0 as shown at
   https://oss.oracle.com/licenses/upl.
   ```

Configure Initial Setup

Use the patching tool to configure the region from where to download the patches and create the configuration file in the specified Middleware Home.

The patches are hosted on the object storage locations in the regions supported by the patching tool. Currently, the regions: *us-phoenix-1*, *us-ashburn-1*, *eu-frankfurt-1*, *ap-mumbai-1*, *ap-tokyo-1*, and *sa-saopaulo-1* are supported.

Ensure that you establish a connection from the WebLogic compute instance where you are running `patch-utils` to the object storage endpoints for the regions that are listed in the following table.

Table 8-3 Object Storage Endpoints for the Supported Regions

Region	Object Storage Endpoint
us-phoenix-1	`https://objectstorage.us-phoenix-1.oci.oraclecloud.com`
us-ashburn-1	`https://objectstorage.us-ashburn-1.oraclecloud.com`
eu-frankfurt-1	`https://objectstorage.eu-frankfurt-1.oraclecloud.com`

Table 8-3 (Cont.) Object Storage Endpoints for the Supported Regions

Region	Object Storage Endpoint
ap-mumbai-1	`https://objectstorage.ap-mumbai-1.oraclecloud.com`
ap-tokyo-1	`https://objectstorage.ap-tokyo-1.oraclecloud.com`
sa-saopaulo-1	`https://objectstorage.sa-saopaulo-1.oraclecloud.com`

Note:

You must configure a NAT gateway to use the patching tool functionality if:

- Your WebLogic compute instance is deployed in a region different from those listed in the Table 8-3.
- Your WebLogic compute instance is deployed in a region different than the region where you set up the patching tool configuration.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Set up the configuration.

   ```
   patch-utils setup
   ```

   Sample output:

   ```
   Enter middleware home (default: /u01/app/oracle/middleware):
   Choose oci region for patch download
   ['us-ashburn-1', 'eu-frankfurt-1', 'ap-mumbai-1', 'ap-tokyo-1', 'us-
   phoenix-1', 'sa-saopaulo-1']: us-phoenix-1
   Created config file [/home/opc/.patchutils/config]
   ```

List Patches

Use the patching tool to list all the available patches in the object storage. You can also list current patches and latest patches that are available in the patching tool repository.

> **Note:**
>
> You must set up the configuration file before running the `patch-utils list` command. See Configure Initial Setup.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Run the following commands to list patches:
   - List all patches in the object storage for the applicable WebLogic Server version.

     ```
     patch-utils list
     ```

     Sample output:

     ```
     <Patch number> ADF Bundle Patch for Bug: <Bug number>, WLS version:
     <WLS version number>
     <Patch number> OPSS Patch Bundle Patch for Bug:<Bug number>, WLS
     version: <WLS version number>
     <Patch number> PATCH <Patch number>- OPATCH <OPatch version number>FOR
     FMW/WLS <WLS version number>AND <WLS version number>
     <Patch number> Oracle Coherence Patch Bundle Patch for Bug:<Bug
     number>, WLS version: <WLS version number>
     <Patch number>Weblogic Service Patch Bundle Patch for Bug:<Bug number>,
     WLS version: <WLS version number>
     ```

   - List all the current patches based on OPatch utility for 12c and BSU (BEA Smart Update) for 11g.

     ```
     patch-utils list -a
     ```

     Sample output:

     ```
     Listing current patches
     Oracle Interim Patch Installer version <Patch version number>)
     Copyright (c) 2020, Oracle Corporation. All rights reserved
     Oracle Home : /u01/app/oracle/middleware
     Central Inventory : /u01/app/oraInventory from : /u01/app/oracle/
     middleware/oraInst.loc
     ```

```
OPatch version : <OPatch Version number>
OUI version : <OUI Version number>
Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
<opatchtimestamp>.log
OPatch detects the Middleware Home as "/u01/app/oracle/middleware"
Lsinventory Output file location : /u01/app/oracle/middleware/
cfgtoollogs/opatch/lsinv/<lsinventoryopatchtimestamp>.txt
Local Machine Information:
Hostname: testwls-wls-0.wlssubnet.subnet1.oraclevcn.com
ARU platform id: <ID number>
ARU platform description:: Linux x86-64
Interim patches (1):
Patch <WebLogic 12c version number>: applied on <day month date time>
Unique Patch ID: <Patch ID number>
Patch description: "Bundle patch for Oracle Coherence Version <WebLogic
12c version number>"
Created on <date month year time>
Bugs fixed:<Bug number>
OPatch succeeded.
```

- List the latest patches and other component patches for the relevant WebLogic Server version, from the available patches in catalog.
  For WebLogic Server compute instances, the latest patches are listed for a given Middleware Home. In case of multiple Middleware Homes, you must use the `patch-utils setup` command to change the Middleware Home.

  ```
  patch-utils list -L
  ```

  Sample output on the WebLogic Server compute instance:

  ```
  Patch Id           Description
  ----------
  ------------------------------------------------------------------------
  ---
  <Patch number>    FMW Thirdparty Bundle Patch 12.2.1.4.220915
  <Patch number>    Opatch 13.9.4.2.11 for EM 13.4, 13.5 and FMW/WLS
  12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0
  <Patch number>    WLS Patch Set Update 12.2.1.4.220929
  <Patch number>    Merge Request on Top of 12.2.1.4.0 for Bugs <Bug
  number> <Bug number> <Bug number>  <Bug number>
  <Patch number>    Coherence 12.2.1.4 Cumulative Patch 15 (12.2.1.4.15)
  ```

  Sample output on the bastion instance:

  ```
  Choose wls type ['WLS', 'FusionMiddleware', 'Coherence', 'Forms',
  'Database'] (default: ALL):

  Patch Id           Description
  ----------
  ------------------------------------------------------------------------
  ---
  <Patch number>    FMW Thirdparty Bundle Patch 12.2.1.4.220915
  <Patch number>    Opatch 13.9.4.2.11 for EM 13.4, 13.5 and FMW/WLS
  12.2.1.3.0, 12.2.1.4.0 and 14.1.1.0.0
  ```

```
<Patch number>    WLS Patch Set Update 12.2.1.4.220929
<Patch number>    Merge Request on Top of 12.2.1.4.0 for Bugs <Bug
number> <Bug number> <Bug number> <Bug number>
<Patch number>    Coherence 12.2.1.4 Cumulative Patch 15 (12.2.1.4.15)
```

List Pending Patches

Use the patching tool to list the pending patches in the patching tool repository that need to be applied on a Middleware Home.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Run the following command:

   ```
   patch-utils diff
   ```

   Sample output:

   ```
   Patch Id         Patch Components     Description
   ----------       ------------------
   ---------------------------------------------------------------------------
   <Patch number>     WLS, FMW          Opatch 13.9.4.2.11 for EM 13.4, 13.5
   and FMW/WLS 12.2.1.3.0, 12.2.1.4.0 and
                                         14.1.1.0.0
   <Patch number>     WLS, FMW          Merge Request on Top of 12.2.1.4.0
   for Bugs 34280277 26354548 26629487
                                         29762601
   <Patch number>     WLS, FMW          Coherence 12.2.1.4 Cumulative Patch
   15 (12.2.1.4.15)
   ```

   To list the pending patches for a given Middleware Home, use the following command:

   ```
   patch-utils diff -m <MW_HOME>
   ```

   If you don't use `-m` option, you must first set up the configuration using the provided Middleware Home. See Configure Initial Setup.

   When you apply the latest patches, if the Middleware Home is not present, you receive the following error message:

   ```
   Unable to find middleware home [<MW_HOME>]. Please run on node with
   Middleware home present or to update the middleware home, run - patch-
   utils setup.
   ```

View Patch Details

Use the patching tool to view information of the specified patch.

The WebLogic Server patches include the `readme` file that provides the patch details and other useful information about patching.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. View the information for the selected patch.

   ```
   patch-utils info -i <Patch ID>
   ```

   By default, the first ten lines of the `readme.txt` file is displayed.

   Sample output:

   ```
   Patch Set Update (PSU) for Bug: <Bug number>
   Date: Fri Feb 28 17:33:37 2020
   Platform Patch for : Generic
   Product Patched : ORACLE WEBLOGIC SERVER
   Product Version : <WLS version number>
   This document describes how to install patch for bug # 31985811.It
   includes the following sections:
   Section 1: Known Issues
   .......
   more
   ....
   ```

   You can define the number of lines to be displayed using the `-l` parameter.

   For example, to print 25 lines, run the following command:

   ```
   patch-utils info -i <Patch ID> -n 25
   ```

Download Patches

Use the patching tool to download the patches to the `/tmp` directory.

If your WebLogic compute instance does not have connectivity to the object storage endpoints for the regions that you configured in the initial setup, you must download the patches on the bastion host that you created with Oracle WebLogic Server for OCI and then copy the zip files to the WebLogic VM.

For the list of regions, see Configure Initial Setup.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Run the following commands to download the patches:
   - Download latest patches.

     ```
     patch-utils download -L -p /tmp/<Location to download>
     ```

     Sample output:

     ```
     Successfully downloaded following patches.
     Please copy them to weblogic hosts and apply them locally.['<Patch
     ID_Generic.zip']
     ```

   - Download the patches using the patch ID.

     ```
     patch-utils download -l <Patch ID> -p /tmp/<Location to download>
     ```

     > **Note:**
     >
     > To download multiple patches, specify the patch IDs as comma separated values. Ensure that you download the patches to an accessible location.

     Sample output:

     ```
     Successfully downloaded following patches.
     Please copy them to weblogic hosts and apply them locally.['<Patch
     ID_Generic.zip']
     ```

Apply Patches

Use the patching tool to apply the latest patches on a given Middleware Home.

The `patch utils apply` command downloads the patches from the object storage and then applies the patches to the Middleware Home. The patches that are already installed in the Middleware Home are not applied. You can apply multiple patches by specifying the patch numbers as comma separated lists. The patches are applied in the order they are specified.

If your WebLogic compute instance does not have connectivity to the object storage endpoints for the regions that you configured in the initial set up, you must first download the patches on the bastion host that you created with Oracle WebLogic Server for OCI and then copy the zip files to the WebLogic VM. After download of patches is complete, run `patch-utils apply -f <downloaded zip path>` to apply the patch.

For the list of regions, see Configure Initial Setup.

When you run the apply command, if you receive an error that the Weblogic servers are running on the host, you must stop the servers. See. Start and Stop a Domain.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Run the following commands to apply patches:

   - Apply latest patches.

     ```
     patch-utils apply -L
     ```

     Sample output:

     ```
     Applying latest patches ['<Patch number1>', '<Patch number2>']
     ---------------------------------------------------------

     Thu Nov 25 11:56:25 GMT 2022 - Applying OPATCH upgrade p<Patch
     number1>_122140_1394211_Generic
     ---------------------------------------------------------

     Latest OPATCH version p28186730_122140_1394211_Generic is applied on
     the middleware home

     ------------------------------------------------------------------------
     -------------------

     Thu Nov 25 11:56:26 GMT 2022 - Applying OPATCH patch p<Patch
     number2>_122140_Generic
     ------------------------------------------------------------------------
     --------------------

     Oracle Interim Patch Installer version 13.9.4.2.11
     Copyright (c) 2022, Oracle Corporation.  All rights reserved.


     Oracle Home       : /u01/app/oracle/middleware
     Central Inventory : /u01/app/oraInventory
     from           : /u01/app/oracle/middleware/oraInst.loc
     OPatch version    : 13.9.4.2.11
     OUI version       : 13.9.4.0.0
     Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
     opatch2022-11-25_11-56-26AM_1.log


     OPatch detects the Middleware Home as "/u01/app/oracle/middleware"

     Verifying environment and performing prerequisite checks...
     OPatch continues with these patches:   <Patch number2>
     ```

```
Do you want to proceed? [y|n]
Y (auto-answered by -silent)
User Responded with: Y
All checks passed.

Please shutdown Oracle instances running out of this ORACLE_HOME on the
local system.
(Oracle Home = '/u01/app/oracle/middleware')


Is the local system ready for patching? [y|n]
Y (auto-answered by -silent)
User Responded with: Y
Backing up files...
Applying interim patch '<Patch number2>' to OH '/u01/app/oracle/
middleware'
ApplySession: Optional component(s) [ oracle.sysman.fmw.plugin.soa,
12.2.1.4.0 ] , [ oracle.sysman.fmw.plugin.wc, 12.2.1.4.0 ] ,
[ oracle.sysman.fmw.plugin.wc, 12.2.1.4.0 ] ,
[ oracle.sysman.fmw.plugin.beam, 12.2.1.4.0 ] ,
[ oracle.sysman.fmw.plugin.idm, 12.2.1.4.0 ] ,
[ oracle.sysman.fmw.plugin.idm, 12.2.1.4.0 ] ,
[ oracle.sysman.fmw.plugin.webtier.otd, 12.2.1.4.0 ]  not present in
the Oracle Home or a higher version is found.

Patching component oracle.sysman.fmw.as, 12.2.1.4.0...

Patching component oracle.sysman.fmw.as, 12.2.1.4.0...

Patching component oracle.sysman.fmw.agent, 12.2.1.4.0...

Patching component oracle.sysman.fmw.core, 12.2.1.4.0...
Patch <Patch number2> successfully applied.
Log file location: /u01/app/oracle/middleware/cfgtoollogs/opatch/
opatch2022-11-25_11-56-26AM_1.log

OPatch succeeded.
Oracle Interim Patch Installer version 13.9.4.2.11
Copyright (c) 2022, Oracle Corporation.  All rights reserved.


Oracle Home       : /u01/app/oracle/middleware
Central Inventory : /u01/app/oraInventory
from           : /u01/app/oracle/middleware/oraInst.loc
OPatch version    : 13.9.4.2.11
OUI version       : 13.9.4.0.0
Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
opatch2022-11-25_11-57-30AM_1.log


OPatch detects the Middleware Home as "/u01/app/oracle/middleware"

Invoking utility "cleanup"
OPatch will clean up 'restore.sh,make.txt' files and 'scratch,backup'
directories.
```

```
You will be still able to rollback patches after this cleanup.
Do you want to proceed? [y|n]
Y (auto-answered by -silent)
User Responded with: Y

Backup area for restore has been cleaned up. For a complete list of
files/directories
deleted, Please refer log file.

OPatch succeeded.

Successfully applied patch p<Patch number2>_122140_Generic
```

To apply the patches for a given Middleware Home, use the following command:

```
patch-utils apply -L -m <MW_HOME>
```

If you don't use `-m` option, you must first set up the configuration using the provided Middleware Home. See Configure Initial Setup.

When you apply the latest patches, if the Middleware Home is not present, you receive the following error message:

```
Unable to find middleware home [<MW_HOME>]. Please run on node with
Middleware home present or to update the middleware home, run - patch-
utils setup.
```

- Apply OPatch.
  This command leverages the OPATCH utility found in the Middleware Home to apply individual patches.

  ```
  patch-utils apply -l <Patch ID>
  ```

  Sample output:

  ```
  Applying OPATCH patch <OPatch ID>
  Oracle Interim Patch Installer version <OPatch Version number>
  Copyright (c) 2020, Oracle Corporation. All rights reserved.
  Oracle Home : /u01/app/oracle/middleware
  Central Inventory : /u01/app/oraInventory
  from : /u01/app/oracle/middleware/oraInst.loc
  OPatch version : <OPatch Version number>
  OUI version : <OUI Version number>
  Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
  <opatchtimestamp>.log
  OPatch detects the Middleware Home as "/u01/app/oracle/middleware"
  Verifying environment and performing prerequisite checks...
  OPatch succeeded.
  ```

- Apply local patch.
  This command applies the patch using the local zip file.

> **Note:**
>
> In case of private subnets without a NAT gateway, you must first download the patches on the bastion host that you created with Oracle WebLogic Server for OCI, copy the zip files to the WebLogic VM, and then run the following command.

```
patch-utils apply -f /tmp/<Patch ID>.zip
```

Sample output:

```
Listing opatch inventory before applying new patches
Oracle Interim Patch Installer version <OPatch Version number>
Copyright (c) 2020, Oracle Corporation.  All rights reserved.
Oracle Home : /u01/app/oracle/middleware
Central Inventory : /u01/app/oraInventory
from : /u01/app/oracle/middleware/oraInst.loc
OPatch version : <OPatch Version number>
OUI version : <OUI Version number>
Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
<opatchtimestamp>.log
OPatch detects the Middleware Home as "/u01/app/oracle/middleware"
Local Machine Information:
Hostname: testwls-wls-0.wlssubnet.subnet1.oraclevcn.com
ARU platform id: <ID number>
ARU platform description:: Linux x86-64
Interim patches (1):
Patch <WebLogic 12c version number>: applied on <day month date time>
Unique Patch ID: <Patch ID number>
Patch description: "Bundle patch for Oracle Coherence Version <WebLogic
12c version number>"
Created on <date month year time>
Bugs fixed:<Bug number>
OPatch succeeded
Applying OPATCH patch <_ OPatch ID.zip>
Oracle Interim Patch Installer version <OPatch Version number>
Copyright (c) 2020, Oracle Corporation. All rights reserved.
Oracle Home : /u01/app/oracle/middleware
Central Inventory : /u01/app/oraInventory
from : /u01/app/oracle/middleware/oraInst.loc
OPatch version : <OPatch Version number>
OUI version : <OUI Version number>
Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
<opatchtimestamp>.log
OPatch detects the Middleware Home as "/u01/app/oracle/middleware"
Verifying environment and performing prerequisite checks...
OPatch succeeded.
```

Roll Back Patches

Use the patching tool to roll back the latest applied patches.

The `patch-utils rollback -L` command rolls back all patches that are applied using the `patch-utils apply -L` command.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Run the following command to roll back the applied patches:

   ```
   patch-utils rollback -L
   ```

   Sample output:

   ```
   Rolling back last applied patches <Patch number1>,<Patch number2>
   ---------------------------------------------------------------------------
   -----------------

   Thu Nov 25 11:35:55 GMT 2022 - Rollback OPATCH patch <Patch
   number1>,<Patch number2>
   ---------------------------------------------------------------------------
   -----------------

   Oracle Interim Patch Installer version 13.9.4.2.11
   Copyright (c) 2022, Oracle Corporation.  All rights reserved.


   Oracle Home       : /u01/app/oracle/middleware
   Central Inventory : /u01/app/oraInventory
   from           : /u01/app/oracle/middleware/oraInst.loc
   OPatch version    : 13.9.4.2.11
   OUI version       : 13.9.4.0.0
   Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
   opatch2022-11-25_11-35-55AM_1.log


   OPatch detects the Middleware Home as "/u01/app/oracle/middleware"

   Following patches are not present in the Oracle Home.
   <Patch number2>

   Patches will be rolled back in the following order:
   <Patch number1>
   The following patch(es) will be rolled back: <Patch number1>
   ```

```
Please shutdown Oracle instances running out of this ORACLE_HOME on the
local system.
(Oracle Home = '/u01/app/oracle/middleware')


Is the local system ready for patching? [y|n]
Y (auto-answered by -silent)
User Responded with: Y

Rolling back patch <Patch number>...

RollbackSession rolling back interim patch '<Patch number1>' from OH
'/u01/app/oracle/middleware'

Patching component oracle.sysman.fmw.as, 12.2.1.4.0...

Patching component oracle.sysman.fmw.as, 12.2.1.4.0...

Patching component oracle.sysman.fmw.agent, 12.2.1.4.0...

Patching component oracle.sysman.fmw.core, 12.2.1.4.0...
RollbackSession removing interim patch '<Patch number1>' from inventory
Log file location: /u01/app/oracle/middleware/cfgtoollogs/opatch/
opatch2022-11-25_11-35-55AM_1.log

OPatch succeeded.
Oracle Interim Patch Installer version 13.9.4.2.11
Copyright (c) 2022, Oracle Corporation.  All rights reserved.


Oracle Home       : /u01/app/oracle/middleware
Central Inventory : /u01/app/oraInventory
from            : /u01/app/oracle/middleware/oraInst.loc
OPatch version    : 13.9.4.2.11
OUI version       : 13.9.4.0.0
Log file location : /u01/app/oracle/middleware/cfgtoollogs/opatch/
opatch2022-11-25_11-36-44AM_1.log


OPatch detects the Middleware Home as "/u01/app/oracle/middleware"

Invoking utility "cleanup"
OPatch will clean up 'restore.sh,make.txt' files and 'scratch,backup'
directories.
You will be still able to rollback patches after this cleanup.
Do you want to proceed? [y|n]
Y (auto-answered by -silent)
User Responded with: Y

Backup area for restore has been cleaned up. For a complete list of files/
directories
deleted, Please refer log file.

OPatch succeeded.
```

If the roll back command fails for a particular patch rollback, the original state is restored and a message stating the reason for failure is displayed.

To roll back the patches for a given Middleware Home, use the following command:

```
patch-utils rollback -L -m <MW_HOME>
```

If you don't use `-m` option, you must first set up the configuration using the provided Middleware Home. See Configure Initial Setup.

When you roll back the latest applied patches, if the Middleware Home is not present, you receive the following error message:

```
Unable to find middleware home [<MW_HOME>]. Please run on node with
Middleware home present or to update the middleware home, run - patch-
utils setup.
```

Upgrade Patching Tool

Use the patching tool to upgrade the patching tool to the latest version.

1. Connect to the compute instance or the bastion instance as the `opc` user.

   ```
   ssh -i path_to_private_key opc@node_public_ip
   ```

   Or,

   ```
   ssh -i path_to_private_key -o ProxyCommand="ssh -W %h:%p -i
   path_to_private_key opc@bastion_public_ip" opc@node_private_ip
   ```

2. Upgrade `patch-utils` to the latest version.

   ```
   patch-utils upgrade
   ```

> **Note:**
>
> This command is used to upgrade VMs if the NAT Gateway is enabled on the WebLogic Server subnet.
> Sample output:
>
> ```
> Successfully updated patch-utils to [<Patch Utils version
> number>]. Please rerun patch-utils.
> ```

Patch Management Using the Patching Utility Tool on Windows

Use the patching utility tool in Oracle WebLogic Server for OCI to list, download, and get information about patches. You can also view the patching utility version and upgrade the patching utility.

> **Note:**
>
> For information on connecting to the Compute instance using the Remote Desktop, see Connecting to WebLogic Server for OCI Windows Images Deployed on Private Subnet

View Patching Utility Tool Version

Use the patching utility tool to view the build version along with the Oracle license and copyright information.

1. Connect to the Compute instance as the `opc` user with Remote Desktop.
2. Bring up a command window and print the build version.

   ```
   patch-utils -v
   ```

   Sample output:

   ```
   Weblogic Cloud Patch-Utils <Patch version number>
   Copyright (c) 2020, Oracle Corporation and/or its affiliates.Licensed
   under the Universal Permissive License v 1.0 as shown at
   https://oss.oracle.com/licenses/upl.
   ```

Configure Initial Setup

The patches are hosted on the object storage locations in the regions supported by the patching tool. Currently, the regions: *us-phoenix-1*, *us-ashburn-1*, *eu-frankfurt-1*, *ap-mumbai-1*, *ap-tokyo-1*, and *sa-saopaulo-1* are supported.

Ensure that you establish a connection from the WebLogic compute instance where you are running `patch-utils` to the object storage endpoints for the regions that are listed in the following table.

Table 8-4 Object Storage Endpoints for the Supported Regions

Region	Object Storage Endpoint
us-phoenix-1	`https://objectstorage.us-phoenix-1.oci.oraclecloud.com`
us-ashburn-1	`https://objectstorage.us-ashburn-1.oraclecloud.com`
eu-frankfurt-1	`https://objectstorage.eu-frankfurt-1.oraclecloud.com`

Table 8-4 (Cont.) Object Storage Endpoints for the Supported Regions

Region	Object Storage Endpoint
ap-mumbai-1	`https://objectstorage.ap-mumbai-1.oraclecloud.com`
ap-tokyo-1	`https://objectstorage.ap-tokyo-1.oraclecloud.com`
sa-saopaulo-1	`https://objectstorage.sa-saopaulo-1.oraclecloud.com`

> **Note:**
>
> You must configure a NAT gateway to use the patching tool functionality if:
>
> - Your WebLogic compute instance is deployed in a region different from those listed in the Table 8-3.
> - Your WebLogic compute instance is deployed in a region different than the region where you set up the patching tool configuration.

1. Connect to the Compute instance as the `opc` user with Remote Desktop.
2. Bring up a command window and set up the configuration:

   ```
   patch-utils setup
   ```

   Sample output:

   ```
   Choose wls version ['12.2.1.4','14.1.1.0']: 12.2.1.4
   Choose oci region for patch download ['us-ashburn-1', 'eu-frankfurt-1',
   'ap-mumbai-1', 'ap-tokyo-1', 'us-phoenix-1', 'sa-saopaulo-1']: us-phoenix-1
   Created config file [C:\Users\opc\.patchutils\config]
   ```

List Patches

Use the patching utility to list all the available patches in the object storage. You can also list current patches and latest patches that are available in the patching utility repository.

> **Note:**
>
> You must set up the configuration file before running the `patch-utils list` command as described in **Configure Initial Setup**.

1. Connect to the Compute instance as the `opc` user with Remote Desktop.
2. Bring up a command window and list all patches for the applicable WebLogic Server version.

   ```
   patch-utils list
   ```

   Sample output:

   ```
    Patch Id       Description
   ---------------------------------------------------------------------------
   ----------
   32471832  Generic Platform patch for WebLogic Server. Adds a system
   property to disable file store locks. Patch applicable for Oracle WebLogic
   Server on OKE.
   32905339  Generic Platform Patch for Oracle Web Services Manager
   34604561  FMW Thirdparty Bundle Patch 12.2.1.4.220915
   32880070  Generic Platform Patch for Oracle Fusion Middleware
   33059296  Testing WLS PATCH SET UPDATE 12.2.1.4.210629
   33059296  Generic Platform Patch for Oracle WebLogic Server
   32684757  Generic Platform Patch for Jdeveloper, WLS version:
   12.2.1.4.210325
   34535558  ADF Bundle Patch 12.2.1.4.220825
   34653267  WLS Patch Set Update 12.2.1.4.220929
   34566592  OWSM Bundle Patch 12.2.1.4.220905
   34542329  Merge Request on Top of 12.2.1.4.0for Bugs 34280277 26354548
   26629487 29762601
   34545596  Coherence 12.2.1.4 Cumulative Patch 15(12.2.1.4.15)
   33084721  Testing ADF BUNDLE PATCH 12.2.1.4.210706
   32973297  Oracle Coherence Cummulative Patch
   33084721  Generic Platform Patch for Jdeveloper
   36068046  Coherence 12.2.1.4 Cumulative Patch 20(12.2.1.4.20)
   36155700  WLS Patch Set Update 12.2.1.4.240104
   36074941  ADF Bundle Patch 12.2.1.4.231205
   36086980  FMW Thirdparty Bundle Patch 12.2.1.4.231207
   35965629  ADR for Weblogic Server 12.2.1.4.0 Size Optimized for Jan 2024
   35868571  OWSM Bundle Patch 12.2.1.4.231003
   35778804  Coherence 12.2.1.4 Cumulative Patch 19(12.2.1.4.19)
   28186730  Opatch 13.9.4.2.14 for EM 13.4, 13.5 and FMW/WLS 12.2.1.3.0,
   12.2.1.4.0 and 14.1.1.0.0
   ```

3. List the latest patches and other component patches for the relevant WebLogic Server version, from the available patches in catalog.

   ```
   patch-utils list -L
   ```

Sample output:

```
Choose the  component ['WLS', 'FusionMiddleware','Coherence', 'Forms',
'Database'] (default: ALL):
Patch Id  Patch Components        Description
---------------------------------------------------------------------------
---------------
36068046  WLS, FMW, COH       Coherence 12.2.1.4Cumulative Patch
20(12.2.1.4.20)
36155700  WLS, FMW            WLS Patch Set Update 12.2.1.4.240104
36074941  FMW                 ADF Bundle Patch 12.2.1.4.231205
36086980  WLS, FMW            FMW Thirdparty Bundle Patch 12.2.1.4.231207
35965629  WLS, FMW            ADR forWeblogic Server 12.2.1.4.0Size
Optimized forJan 2024
35868571  WLS, FMW            OWSM Bundle Patch 12.2.1.4.231003
```

View Patch Details

Use the patching utility to view information of the specified patch.

The WebLogic Server patches include the `readme` file that provides the patch details and other useful information about patching.

1. Connect to the Compute instance as the `opc` user with Remote Desktop.
2. Bring up a command window and list all patches for the applicable WebLogic Server version.

   ```
   patch-utils info -i <Patch ID>
   ```

   The first ten lines of the `readme.txt` file are displayed.

   Sample output:

   ```
   Oracle Coherence 12.2.1.4.19 Release
   Released: September, 2023

   Oracle Coherence 12.2.1.4.19 README
   ===================================
   This README provides information about how to apply the patch update to
   OracleCoherence 12.2.1.4.0 using OPatch.
   OPatch and Oracle Fusion Middleware Installation Documentation
   --------------------------------------------------------------
   .........more....
   ```

Download Patches

Use the patching utility to download the patches to a directory.

If your WebLogic compute instance does not have connectivity to the object storage endpoints for the regions that you configured in the initial setup, you must download the patches on the bastion host that you created with Oracle WebLogic Server for OCI and then copy the zip files to the WebLogic VM.

1. Connect to the Compute instance as the `opc` user with Remote Desktop.

2. Bring up a command window and download the latest patches for the applicable WebLogic Server version.

   ```
   patch-utils download -L -p <Location to download>
   ```

   Sample output:

   ```
   Choose the component ['WLS','FusionMiddleware','Coherence', 'Forms',
   'Database'] (default: ALL):
   Successfully downloaded following patch(es). Please copy them to weblogic
   hosts and apply them using WebLogic opatch utility.
   ['p36068046_122140_Generic.zip', 'p36155700_122140_Generic.zip',
   'p36074941_122140_Generic.zip', 'p36086980_122140_Generic.zip',
   'p35965629_122140_MSWIN-x86-64.zip', 'p35868571_122140_Generic.zip',
   'p28186730_122140_1394214_Generic.zip']
   ```

3. You can also download a patch using the patch ID.

   ```
   patch-utils download -l <Patch ID> -p <Location to download>
   ```

   Sample output:

   ```
   Successfully downloaded following patch(es).
   Please copy them to weblogic hosts and apply them using WebLogic opatch
   utility.
   ['p35778804_122140_Generic.zip']
   ```

Upgrade Patching Utility

Use the patching tool to upgrade the patching tool to the latest version.

> **Note:**
>
> Upgrading the patching utility requires elevated privileges. The command window must be opened with "Run as Administrator".

1. Connect to the Compute instance as the `opc` user with Remote Desktop.
2. Bring up a command window from the Start menu, right-click and select "Run as Administrator", and print the build version.

   ```
   patch-utils upgrade
   ```

   Sample output:

   ```
   Successfully updated patch-utils to [<Patch Utils version number>]. Please
   rerun patch-utils.
   ```

A
License Information

Learn about the licensed third-party technology associated with Oracle WebLogic Server for OCI.

Open Source or Other Separately Licensed Software

Required notices for open source or other separately licensed software products or components distributed in Oracle WebLogic Server for OCI are identified in the following table along with the applicable licensing information. Additional notices and/or licenses may be found in the included documentation or `readme` files of the individual third party software.

Provider	Component(s)	Licensing Information
Simon Kelly	dnsmasq	GNU General Public License Version 3

B
Configure SSL for a Domain

Secure Socket Layer (SSL) is the most commonly-used method of securing data sent across the internet. For domains created before June2020, you can configure SSL between clients and the load balancer used to access your Oracle WebLogic Server for OCI domain.

> **Note:**
>
> This procedure applies only to domains that were created before June 29, 2020.
>
> To set up custom SSL for Oracle WebLogic Server for OCI instances, see Overview of Configuring SSL in WebLogic Server.

In this configuration, SSL connections (the HTTPS protocol) terminate at the load balancer. Connections from the load balancer to the compute instances running Oracle WebLogic Server do not use SSL; they use the HTTP protocol.

If you selected the **Prepare Load Balancer for HTTPS** option when creating the domain, then you only need to perform these tasks:

- Add a Certificate to the Load Balancer

If you did not select this option when creating the domain, then you must perform all of the tasks:

- Create an HTTPS Listener for the Load Balancer
- Add a Certificate to the Load Balancer
- Update the App Gateway for HTTPS (if the domain uses Oracle Identity Cloud Service)

Create an HTTPS Listener for the Load Balancer

Update the load balancer for your domain. Create a listener for the HTTPS port, and then configure the SSL request headers for Oracle WebLogic Server.

> **Note:**
>
> This procedure applies only to domains that were created before June 2020. The steps are required only if you did *not* select the **Prepare Load Balancer for HTTPS** option when creating the domain.

The SSL request headers instruct WebLogic Server to use the HTTPS protocol in external URLs that it generates, such as in web application links.

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Networking**, and then click **Load Balancers**.

3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack was initially created, this might be the same compartment that contains the compute instances for the domain.

4. Click the load balancer that was provisioned as part of your stack, `*prefix*-lb`.
5. Click **Rule Sets**.
6. Click **Create Rule Set**
7. For **Name**, enter `SSLHeaders`.
8. Click **Specify Request Header Rules**.
9. Specify these header parameters.
   - **Action**: Add Request Header
   - **Header**: `WL-Proxy-SSL`
   - **Value**: `true`
10. Click **Another Request Header Rule**.
11. Specify these header parameters.
    - **Action**: Add Request Header
    - **Header**: `is_ssl`
    - **Value**: `ssl`
12. Click **Create**, and then click **Close**.
13. Click **Listeners**.
14. Click **Create Listener**
15. For **Name**, enter `https`.
16. For **Port**, enter `443`.
17. For **Backend Set**, select the backend resource created for the load balancer.
18. Click **Create**, and then click **Close**.
19. After the `https` listener is created, edit it.
20. Click **Additional Rule Set**, and then select `SSLHeaders`.
21. Click **Save Changes**, and then click **Close**.
22. If you provisioned a new load balancer subnet as part of your stack, update the security list for this subnet and permit access to the HTTPS port.
    a. From the navigation menu, select **Networking**, and then click **Virtual Cloud Networks**.
    b. Click the virtual cloud network (VCN) used by your domain.
    c. Click **Security Lists**.
    d. Click the load balancer security list that was provisioned for your stack.
       - *prefix*`-lb-security-list`, if you created a single regional subnet
       - *prefix*`-wls-lb-security-list-1` and *prefix*`-wls-lb-security-list-2`, if you created subnets for specific availability domains
    e. Edit the existing ingress rule for port 80.

f. Change the **Destination Port Range** to 443.

g. Click **Save Changes**.

If you want to delete this stack at a later time, you will not be able to destroy the stack using Resource Manager. Because of the changes to the load balancer resources, you will have to manually delete the load balancer.

See these topics in the Oracle Cloud Infrastructure documentation:

- Managing Listeners
- Managing Rule Sets
- Security Lists

Add a Certificate to the Load Balancer

Upload your SSL certificate, and then associate the certificate with the HTTPS listener.

> **Note:**
>
> This procedure applies only to domains that were created before June 2020.

You can use a custom, self-signed SSL certificate, or a certificate that you've obtained from a Certificate Authority (CA). For production WebLogic Server environments, Oracle recommends that you use a CA-issued SSL certificate, which reduces the chances of experiencing a man-in-the-middle attack.

1. Access the Oracle Cloud Infrastructure console.
2. From the navigation menu, select **Networking**, and then click **Load Balancers**.
3. Select the **Compartment** in which the network resources for your domain were created.

   Depending on how the stack was initially created, this might be the same compartment that contains the compute instances for the domain.
4. Click the load balancer that was provisioned as part of your stack, `*prefix*-lb`.
5. Click **Certificates**.
6. Click **Add Certificate**.
7. Enter a name for your certificate.
8. Either upload the certificate file, or paste its contents into the text area.
9. If applicable, specify a CA certificate or a private key file.

   For example, if you are using a self-signed certificate, upload the corresponding private key file. See Managing SSL Certificates in the Oracle Cloud Infrastructure documentation.
10. Click **Add Certificate**, and then click **Close**.
11. After the certificate was successfully added, click **Listeners**.
12. Edit the `https` listener.
13. Click **Use SSL**, and then select your new certificate.
14. Click **Save Changes**, and then click **Close**.
15. If a listener exists named `http`, delete this listener.

This is the default load balancer listener if you did not select the **Prepare Load Balancer for HTTPS** option when creating the domain.

You cannot modify an existing load balancer certificate. You must add a new certificate, and then associate the listener with the new certificate.

Update the App Gateway for HTTPS

If your Oracle WebLogic Server domain uses Oracle Identity Cloud Service for authentication, update and restart the App Gateway on each compute instance in the domain.

> **Note:**
>
> This procedure applies only to domains that were created before June 2020. The steps are required only if both of these are true:
>
> - This procedure applies only to domains that were created before June 2020.
> - You did *not* select the **Prepare Load Balancer for HTTPS** option when creating the domain.
> - You selected the **Enable Authentication Using Identity Cloud Service** option when creating the domain.

1. Open an SSH connection to the first compute instance in the domain, as the `opc` user.

   Example:

   ```
   ssh -i mykey opc@203.0.113.13
   ```

2. Create a backup of the folder `/u01/data/cloudgate_config`.

   ```
   sudo cp -avr /u01/data/cloudgate_config /u01/data/cloudgate_config_bak
   ```

3. Edit the file `/u01/data/cloudgate_config/appgateway-env`.
4. Edit the variable named `CG_CALLBACK_PREFIX`. Replace `http` with `https`.

   ```
   CG_CALLBACK_PREFIX=https://%hostid%
   ```

5. Stop and remove the App Gateway container.

   ```
   sudo podman container stop appgateway


   sudo podman container rm appgateway
   ```

6. Delete the contents of the folder `/u01/data/cloudgate_config`, *except* for the following files.
   - `appgateway-env`
   - `cwallet.sso`
   - `origin_conf`

7. Start the App Gateway container.

   ```
   sudo /opt/scripts/idcs/run_cloudgate.sh
   ```

8. Repeat from **step 1** for all remaining compute instances in this domain.

C
Script Files

This section list all the required script files required for Oracle WebLogic Server for OCI.

Topics:

- Script File to Create Policies for Autoscaling Functions Post Provisioning
- Script File to Delete Resources

Script File to Create Policies for Autoscaling Functions Post Provisioning

Use the script file to create policies for autoscaling functions after creating the Oracle WebLogic Server for OCI domain.

If the **OCI Policies** check box is not selected during domain creation (`create_policies` set to `false`), you can use the script file to create function's dynamic group and policies after creating the domain. The script verifies if the stack has `create_policies` set to `false`, in which case it returns an error that policies would be automatically created via stack. However, you can use the `--force` option to override and create dynamic group and policies when `create_policies == true`.

Copy the following scripts in Cloud Shell to create polices for autoscaling. For example, copy the scripts and save the file as `create_autoscaling_policies.py`

> **Note:**
>
> For instructions on how to use the script to create function's dynamic group and policies, see Dynamic Group Policies for Autoscaling.

After you run the script, you must enable alarms from the Oracle Cloud Infrastructure console. See To enable an alarm in Oracle Cloud Infrastructure documentation.

```
"""
#
# Copyright (c) 2022, Oracle Corporation and/or its affiliates.
# Licensed under the Universal Permissive License v 1.0 as shown at https://
oss.oracle.com/licenses/upl.
"""
import argparse
import logging
import oci
import os
import sys
import textwrap
from oci.identity import models as identity_models
```

```
logger = logging.getLogger()
logging.basicConfig()
REGION = ''


def get_config():
    if REGION != '':
        return {"region": REGION}
    return {}


def get_variable(variables, name, default=None):
    """
    First read from stack variables
    If not found and default provided, return default
    :param variables:
    :param name:
    :param default:
    :return:
    """
    ret = None
    found = True

    if name in variables:
        ret = variables.get(name)
    else:
        if default is not None:
            return default
        else:
            logger.info("ERROR: value for the variable [%s] could not be 
found!" % (name))
            found = False

    if found:
        logger.debug("Value of variable [%s] is [%s]" % (name, ret))

    return ret


def inspect_stack(signer, stack_ocid):
    client = oci.resource_manager.ResourceManagerClient(config=get_config(), 
signer=signer)
    try:
        stack = client.get_stack(stack_id=stack_ocid)
    except(Exception, ValueError) as ex:
        logger.exception("ERROR: accessing resource manager stack failed")
        raise
    resp = stack.data

    return resp


def create_functions_dynamic_group(signer, stack_compartment_id, tenancy_id, 
service_name):
    """
    Creates functions dynamic group in root compartment.
    :param signer:
    :param stack_compartment_id:
    :param tenancy_id:
    :param service_name:
    :return:
    """
    client = oci.identity.IdentityClient(config={}, signer=signer)
    client_composite_ops =
oci.identity.IdentityClientCompositeOperations(client)
    dg_name = "{0}-functions-dynamic-group-manual".format(service_name)
    try:
        create_dynamicgrp_resp =
client_composite_ops.create_dynamic_group_and_wait_for_state(

create_dynamic_group_details=identity_models.CreateDynamicGroupDetails(
                compartment_id=tenancy_id,
                description="Autoscaling Functions Dynamic Group of stack:
{0}".format(service_name),
                matching_rule="All {resource.type = 'fnfunc', "
                              "resource.compartment.id='%s'}" %
(stack_compartment_id),
                name=dg_name
            ),

wait_for_states=[identity_models.DynamicGroup.LIFECYCLE_STATE_ACTIVE]
        )
    except(Exception, ValueError) as ex:
        logger.exception("Failed to create functions dynamic group
{0}".format(dg_name))
        raise
    return create_dynamicgrp_resp.data


def create_policies_for_functions_dynamic_group(signer, stack_compartment_id,
tenancy_id, service_name,
                                                network_compartment_id,

apm_domain_compartment_id=None,
                                                atp_db_compartment_id=None,
                                                ocidb_compartment_id=None,

app_atp_db_compartment_id=None,
                                                app_ocidb_compartment_id=None,
                                                fss_compartment_id=None):
    """
    Creates policies for functions dynamic group.

    :param signer:
    :param stack_compartment_id:
    :param tenancy_id:
    :param service_name:
    :param network_compartment_id:
    :param apm_domain_compartment_id:
    :param atp_db_compartment_id:
    :param ocidb_compartment_id:
```

```
    :param app_atp_db_compartment_id:
    :param app_ocidb_compartment_id:
    :param fss_compartment_id:
    :return:
    """
    client = oci.identity.IdentityClient(config={}, signer=signer)
    client_composite_ops =
oci.identity.IdentityClientCompositeOperations(client)
    dg_name = "{0}-functions-dynamic-group-manual".format(service_name)
    try:
        policies = [
            "Allow dynamic-group {0} to inspect tenancies in
tenancy".format(dg_name),
            "Allow dynamic-group {0} to inspect limits in
tenancy".format(dg_name),
            "Allow dynamic-group {0} to manage volume-family in compartment
id {1}".format(dg_name,

             stack_compartment_id),
            "Allow dynamic-group {0} to manage instance-family in compartment
id {1}".format(dg_name,

               stack_compartment_id),
            "Allow dynamic-group {0} to use app-catalog-listing in
compartment id {1}".format(dg_name,

                stack_compartment_id),
            "Allow dynamic-group {0} to manage virtual-network-family in
compartment id {1}".format(dg_name,

                      network_compartment_id),
            "Allow dynamic-group {0} to manage load-balancers in compartment
id {1}".format(dg_name,

              network_compartment_id),
            "Allow dynamic-group {0} to manage orm-family in compartment id
{1}".format(dg_name, stack_compartment_id),
            "Allow dynamic-group {0} to read metrics in compartment id
{1}".format(dg_name, stack_compartment_id),
            "Allow dynamic-group {0} to manage repos in
tenancy".format(dg_name),
            "Allow dynamic-group {0} to manage functions-family in
compartment id {1}".format(dg_name,

                stack_compartment_id),
            "Allow dynamic-group {0} to manage ons-topics in compartment id
{1}".format(dg_name, stack_compartment_id),
            "Allow dynamic-group {0} to manage instance-agent-command-family
in compartment id {1}".format(dg_name,

                             stack_compartment_id),
            "Allow dynamic-group {0} to manage alarms in compartment id
{1}".format(dg_name, stack_compartment_id),
            "Allow dynamic-group {0} to manage log-groups in compartment id
{1}".format(dg_name, stack_compartment_id),
            "Allow dynamic-group {0} to manage unified-configuration in
```

```
compartment id {1}".format(dg_name,

                  stack_compartment_id),
            "Allow dynamic-group {0} to inspect dynamic-groups in
tenancy".format(dg_name),
            "Allow dynamic-group {0} to use tag-namespaces in
tenancy".format(dg_name)
        ]
        if apm_domain_compartment_id is not None:
            policies.append("Allow dynamic-group {0} to use apm-domains in
compartment id {1}".format(dg_name,

                        apm_domain_compartment_id))
        if atp_db_compartment_id is not None:
            policies.append(
                "Allow dynamic-group {0} to inspect autonomous-transaction-
processing-family in compartment id {1}".format(
                    dg_name,
                    atp_db_compartment_id))
        if ocidb_compartment_id is not None:
            policies.append("Allow dynamic-group {0} to inspect database-
family in compartment id {1}".format(dg_name,

                                ocidb_compartment_id))
        if app_ocidb_compartment_id is not None:
            policies.append("Allow dynamic-group {0} to inspect database-
family in compartment id {1}".format(dg_name,

                                app_ocidb_compartment_id))
        if app_atp_db_compartment_id is not None:
            policies.append(
                "Allow dynamic-group {0} to use autonomous-transaction-
processing-family in compartment id {1}".format(
                    dg_name,
                    app_atp_db_compartment_id))
        if fss_compartment_id is not None:
            policies.append("Allow dynamic-group {0} to manage mount-targets
in compartment id {1}".format(dg_name,

                             fss_compartment_id))
            policies.append("Allow dynamic-group {0} to manage file-systems
in compartment id {1}".format(dg_name,

                            fss_compartment_id))
            policies.append("Allow dynamic-group {0} to manage export-sets in
compartment id {1}".format(dg_name,

                           fss_compartment_id))
        create_policy_resp =
client_composite_ops.create_policy_and_wait_for_state(
            create_policy_details=identity_models.CreatePolicyDetails(
                compartment_id=tenancy_id,
                description="Policy for Autoscaling Functions Dynamic Group
of stack: {0}".format(service_name),
                statements=policies,
                name="{0}-wlsc-policy-manual".format(service_name)
            ),
            wait_for_states=[identity_models.Policy.LIFECYCLE_STATE_ACTIVE]
        )
    except(Exception, ValueError) as ex:
        logger.exception("Failed to create policies for functions dynamic 
group {0}".format(dg_name))
        raise
    return create_policy_resp.data


def get_apm_compartment_id(signer, apm_domain_id):
    client = oci.apm_control_plane.ApmDomainClient(config=get_config(), 
signer=signer)
    try:
        resp = client.get_apm_domain(apm_domain_id)
    except(Exception, ValueError) as ex:
        logger.exception("Failed to get compartment for APM domain ID 
[{0}]".format(apm_domain_id))
        raise
    apm_domain = resp.data
    return apm_domain.compartment_id


def main():
    signer = None
    # delegate token should be present at /etc/oci/delegation_token in cloud 
shell
    if os.path.exists('/etc/oci/delegation_token'):
        with open('/etc/oci/delegation_token', 'r') as file:
            delegation_token = file.read()
        signer = 
oci.auth.signers.InstancePrincipalsDelegationTokenSigner(delegation_token=dele
gation_token)
    else:
        logger.error("In the Cloud shell the delegation token does not exist 
at location /etc/oci/delegation_token. "
                     "Run the script from the Cloud shell, where you need to 
delete the resources.")
        sys.exit(1)

    if signer is None:
        logger.error('Instance principal signer is not initialized')
        sys.exit(1)

    parser = argparse.ArgumentParser(description=textwrap.dedent('''
        This script creates Functions Dynamic Group and Policies for 
Autoscaling.
    '''), formatter_class=argparse.RawDescriptionHelpFormatter)
    parser.add_argument('stack_ocid', help='Stack OCID')
    parser.add_argument('-r', '--region', default='', help='Region other than 
home region must be specified.')
    parser.add_argument('-d', '--debug', action='store_const', const=True, 
help='Enable Debug')
    parser.add_argument('-f', '--force', action='store_const', const=True, 
help='Force creation of dynamic group and '
```

```
  'policies')

    args = parser.parse_args()

    stack_ocid = args.stack_ocid
    global REGION

    REGION = args.region
    debug = args.debug
    if debug:
        logger.setLevel(logging.DEBUG)
    else:
        logger.setLevel(logging.INFO)
    force = args.force

    # Get stack variables
    resp = inspect_stack(signer, stack_ocid)
    logger.debug(resp)
    variables = resp.variables
    create_policies = get_variable(variables, 'create_policies', "true")

    if create_policies == 'true':
        if not force:
            logger.error('Stack has create_policies enabled so manual
creation of autoscaling functions dynamic group '
                         'and policies is not required. Bailing out!')
            return
        else:
            logger.warning('Stack has create_policies enabled so manual
creation of autoscaling functions dynamic '
                           'group and policies is not required.')

    stack_compartment_id = get_variable(variables, 'compartment_ocid')
    tenancy_id = get_variable(variables, 'tenancy_ocid')
    service_name = get_variable(variables, 'service_name')

    # create functions dynamic group
    create_functions_dynamic_group(signer, stack_compartment_id, tenancy_id,
service_name)

    # create policies for the functions dynamic group
    apm_domain_id = get_variable(variables, 'apm_domain_id', "")
    atp_db_id = get_variable(variables, "atp_db_id", "")
    ocidb_dbsystem_id = get_variable(variables, "ocidb_dbsystem_id", "")
    app_atp_db_id = get_variable(variables, "app_atp_db_id", "")
    appdb_dbsystem_id = get_variable(variables, "appdb_dbsystem_id", "")
    add_fss = get_variable(variables, "add_fss", "false")

    network_compartment_id = get_variable(variables,
'network_compartment_id', stack_compartment_id)
    apm_domain_compartment_id = get_apm_compartment_id(signer, apm_domain_id)
if apm_domain_id != '' else None
    atp_db_compartment_id = get_variable(variables, 'atp_db_compartment_id',
                                         stack_compartment_id) if atp_db_id !
= '' else None
    ocidb_compartment_id = get_variable(variables, 'ocidb_compartment_id',
                                        stack_compartment_id) if
ocidb_dbsystem_id != '' else None
    app_atp_db_compartment_id = get_variable(variables,
'app_atp_db_compartment_id',
                                             stack_compartment_id) if
app_atp_db_id != '' else None
    app_ocidb_compartment_id = get_variable(variables,
'app_ocidb_compartment_id',
                                            stack_compartment_id) if
appdb_dbsystem_id != '' else None
    fss_compartment_id = get_variable(variables, 'fss_compartment_id',
                                      stack_compartment_id) if add_fss ==
'true' else None

    create_policies_for_functions_dynamic_group(signer, stack_compartment_id,
tenancy_id, service_name,
                                                network_compartment_id,
apm_domain_compartment_id,
                                                atp_db_compartment_id,
                                                ocidb_compartment_id,
                                                app_atp_db_compartment_id,
                                                app_ocidb_compartment_id,
                                                fss_compartment_id)


if __name__ == '__main__':
    main()
```

Script File to Delete Resources

You must create a script file to remove the autoscaling resources before you destroy the stack in Oracle WebLogic Server for OCI.

You can copy the following scripts to the script file, `remove_resources.py` file in Cloud Shell.

```
"""
#
# Copyright (c) 2021, 2022, Oracle Corporation and/or its affiliates.
# Licensed under the Universal Permissive License v 1.0 as shown at https://
oss.oracle.com/licenses/upl.
"""

import os
import sys
import oci
from oci.events import models as event_models
from oci.functions import models as fn_models
from oci.monitoring import models as monitoring_models
from oci.artifacts import models as artifacts_models
from oci.ons import models as ons_models
from oci import pagination
import argparse
import textwrap

REGION = ''
```

```
def get_config():
    if REGION != '':
        return {"region": REGION}
    return {}


"""
Lists and deletes the resources like instances, policies, volumes, VCN
related resources, logs and tags etc..
"""


class CleanUpResources:

    def __init__(self, service_name_prefix):
        self.service_name_prefix = service_name_prefix
        # delegate token should be present at /etc/oci/delegation_token in
cloud shell
        if os.path.exists('/etc/oci/delegation_token'):
            with open('/etc/oci/delegation_token', 'r') as file:
                delegation_token = file.read()
            self.signer =
oci.auth.signers.InstancePrincipalsDelegationTokenSigner(delegation_token=dele
gation_token)
        else:
            print("ERROR: In the Cloud shell the delegation token does not
exist at location /etc/oci/delegation_token."
                  "Run the script from the Cloud shell, where you need to
delete the resources.")
            sys.exit(1)
        self.vcn_client = oci.core.VirtualNetworkClient(config=get_config(),
signer=self.signer)
        self.virtual_network_composite_operations =
oci.core.VirtualNetworkClientCompositeOperations(self.vcn_client)
        self.log_client =
oci.logging.LoggingManagementClient(config=get_config(), signer=self.signer)
        self.log_composite_operations =
oci.logging.LoggingManagementClientCompositeOperations(self.log_client)
        self.identity_client = oci.identity.IdentityClient(config={},
signer=self.signer)
        self.identity_client_composite_operations =
oci.identity.IdentityClientCompositeOperations(self.identity_client)
        self.events_client = oci.events.EventsClient(config=get_config(),
signer=self.signer)
        self.events_client_composite_operations =
oci.events.EventsClientCompositeOperations(self.events_client)
        self.fn_client =
oci.functions.FunctionsManagementClient(config=get_config(),
signer=self.signer)
        self.fn_composite_operations =
oci.functions.FunctionsManagementClientCompositeOperations(client=self.fn_clie
nt)
        self.monitoring_client =
oci.monitoring.MonitoringClient(config=get_config(), signer=self.signer)
```

```
        self.monitoring_composite_operations =
oci.monitoring.MonitoringClientCompositeOperations(
            client=self.monitoring_client)
        self.artifacts_client =
oci.artifacts.ArtifactsClient(config=get_config(), signer=self.signer)
        self.artifacts_composite_operations =
oci.artifacts.ArtifactsClientCompositeOperations(
            client=self.artifacts_client)
        self.ons_control_plane_client =
oci.ons.NotificationControlPlaneClient(config=get_config(),
signer=self.signer)

    # Lists all the resources based on the service name prefix
    def list_all_resources(self):
        search_client =
oci.resource_search.ResourceSearchClient(config=get_config(),
signer=self.signer)
        running_resources = ["RUNNING", "Running", "AVAILABLE", "STOPPED",
"Stopped", "ACTIVE", "CREATED", "INACTIVE"]
        resource_not_required = ["PrivateIp", "Vnic"]
        # https://docs.oracle.com/en-us/iaas/Content/Search/Concepts/
queryoverview.htm#resourcetypes
        structured_search =
oci.resource_search.models.StructuredSearchDetails(
            query="query all resources where displayname =~
'{}'".format(self.service_name_prefix),
            type='Structured',

matching_context_type=oci.resource_search.models.SearchDetails.MATCHING_CONTEX
T_TYPE_NONE)

        resources = search_client.search_resources(structured_search)
        resources_details = []
        no_of_resources = 0
        # Tags and default route table doesn't start with service prefix
        tagname_resource = "wlsoci-" + self.service_name_prefix
        default_rt = "Default Route Table for " + self.service_name_prefix
        # Logs and unified agent config resources use service_prefix with
underscore instead of hyphen
        log_resource = self.service_name_prefix[:-1] + "_"
        print(
            "Resource Name                              Resource
Type                        Resource Lifecycle State
OCID         DOC")
        print(

"=============================================================================
====================================================================")
        for resource in resources.data.items:
            resource_name = resource.display_name
            if (resource_name.startswith(
                    self.service_name_prefix) or tagname_resource in
resource_name or default_rt in resource_name or log_resource in
resource_name) and (
                    resource.lifecycle_state in running_resources) and (
                    resource.resource_type not in resource_not_required):
```

```
                resources_details.append(resource)
                no_of_resources = no_of_resources + 1
                print("{}             {}          {}          {}
{}".format(resource.display_name,

        resource.resource_type,

        resource.lifecycle_state,

        resource.identifier,

        resource.time_created))
        print(

"=============================================================================
===================================================================")
        print("Total number of resources {}".format(len(resources_details)))
        return resources_details

    # Removes all resources based on the service name prefix
    def cleanup_resources(self, delete_list):
        print("Deleting the resources")
        self.delete_all_autoscaling_resources(delete_list)
        self.delete_policies(delete_list)
        self.delete_instance(delete_list)
        self.delete_block_volumes(delete_list)
        self.delete_load_balancer(delete_list)
        self.delete_subnet(delete_list)
        self.delete_sec_list(delete_list)
        self.delete_route_table(delete_list)
        self.delete_dhcp_options(delete_list)
        self.delete_internet_gateway(delete_list)
        self.delete_service_gateway(delete_list)
        self.delete_local_peering_gateway(delete_list)
        self.delete_nat_gateway(delete_list)
        self.delete_vcn_resources(delete_list)
        self.delete_unified_agent_configuration(delete_list)
        self.delete_log(delete_list)
        self.delete_log_group(delete_list)
        self.delete_mount_targets(delete_list)
        self.delete_fss(delete_list)
        self.delete_tag_namespace(delete_list)
        self.delete_boot_volumes(delete_list)

    # Delete Policies
    def delete_policies(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Policy":
                policy_ocid = resource.identifier
                print("Deleting policy: {0}, with ocid:
{1}".format(resource.display_name, policy_ocid))
                try:

self.identity_client_composite_operations.delete_policy_and_wait_for_state(
                        policy_ocid,
wait_for_states=[oci.identity.models.Policy.LIFECYCLE_STATE_DELETED])
                    print("Deleted policy successfully!")
                except Exception as e:
                    print("Error while deleting the policy {0}, policy id
{1}, Error message {2}".format(
                        resource.display_name, policy_ocid, str(e)))

    # Delete Dynamic Group
    def delete_dynamic_group(self):
        tenancy = os.environ['OCI_TENANCY']
        dynamic_group_list =
self.identity_client.list_dynamic_groups(tenancy).data
        for d_group in dynamic_group_list:
            if self.service_name_prefix in d_group.name:
                print("Deleting the dynamic group: {0}, with ocid:
{1}".format(d_group.name, d_group.id))
                try:

self.identity_client_composite_operations.delete_dynamic_group_and_wait_for_st
ate(
                        d_group.id,
wait_for_states=[oci.identity.models.DynamicGroup.LIFECYCLE_STATE_DELETED])
                    print("Deleted the dynamic group successfully!")
                except Exception as e:
                    print("Error while deleting the dynamic group name {},
ocid {}, Error message {}".format(
                        d_group.name, d_group.id, str(e)))

    # Delete Block Volumes
    def delete_block_volumes(self, delete_list):
        bv_client = oci.core.BlockstorageClient(config=get_config(),
signer=self.signer)
        bv_composite_operations =
oci.core.BlockstorageClientCompositeOperations(bv_client)
        for resource in delete_list:
            if resource.resource_type == "Volume":
                bv_ocid = resource.identifier
                try:
                    print(
                        "Deleting the block volume: {0}, with ocid
{1}".format(resource.display_name, bv_ocid))
                    bv_composite_operations.delete_volume_and_wait_for_state(
                        bv_ocid,
wait_for_states=[oci.core.models.Volume.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the block volume successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the block volume {0}, ocid {1},
Error message {2}".format(
                            resource.display_name, bv_ocid, str(e)))

    # Delete all compute instances
    def delete_instance(self, delete_list):
        compute_client = oci.core.ComputeClient(config=get_config(),
signer=self.signer)
        compute_composite_operations =
oci.core.ComputeClientCompositeOperations(compute_client)
        for resource in delete_list:
            if resource.resource_type == "Instance":
                instance_ocid = resource.identifier
                instance_name = resource.display_name
                print("Deleting the compute instance: {0}, with ocid
{1}".format(instance_name, instance_ocid))
                try:

compute_composite_operations.terminate_instance_and_wait_for_state(
                        instance_ocid,
wait_for_states=[oci.core.models.Instance.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the compute instance successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the instance {0}, ocid {1},
Error message {2}".format(
                            instance_name, instance_ocid, str(e)))

    # Delete all Subnets in the VCN
    def delete_subnet(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Subnet":
                subnet_ocid = resource.identifier
                print(
                    "Deleting subnet: {0}, with ocid
{1}".format(resource.display_name, resource.identifier))
                try:

self.virtual_network_composite_operations.delete_subnet_and_wait_for_state(
                        subnet_ocid,

wait_for_states=[oci.core.models.Subnet.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted subnet successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the subnet {0}, ocid {1}, Error
message {2}".format(resource.display_name,

                    subnet_ocid, str(e)))

    # Delete Security lists
    def delete_sec_list(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "SecurityList":
                sec_list_name = resource.display_name
                sec_list_ocid = resource.identifier
                if not ("Default" in sec_list_name):
                    print(
                        "Deleting the security list: {0}, with ocid
{1}".format(resource.display_name,

  resource.identifier))
                    try:

self.virtual_network_composite_operations.delete_security_list_and_wait_for_st
```

```
ate(
                            sec_list_ocid,

wait_for_states=[oci.core.models.SecurityList.LIFECYCLE_STATE_TERMINATED])
                        print("Deleted the security list successfully!")
                    except Exception as e:
                        print(
                            "Error while deleting the security list {0}, ocid
{1}, Error message {2}".format(
                                resource.display_name, sec_list_ocid, str(e)))

    # Delete Load balancers
    def delete_load_balancer(self, delete_list):
        lb_client = oci.load_balancer.LoadBalancerClient(config=get_config(),
signer=self.signer)
        lb_composite_operations =
oci.load_balancer.LoadBalancerClientCompositeOperations(lb_client)
        for resource in delete_list:
            if resource.resource_type == "LoadBalancer":
                lb_name = resource.display_name
                lb_ocid = resource.identifier
                print("Deleting Load balancer {0} with ocid
{1}".format(lb_name, lb_ocid))
                try:

lb_composite_operations.delete_load_balancer_and_wait_for_state(
                        lb_ocid,

wait_for_states=[oci.load_balancer.models.WorkRequest.LIFECYCLE_STATE_SUCCEEDE
D])
                    print("Load balancer deleted successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the loadbalancer {0}, ocid {1},
Error message {2}".format(
                            lb_name, lb_ocid, str(e)))

    # Delete Route tables
    def delete_route_table(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "RouteTable":
                route_table_name = resource.display_name
                route_table_ocid = resource.identifier
                # Removing the route rules from the tables
                rt_details = oci.core.models.UpdateRouteTableDetails()
                rt_details.route_rules = []

self.virtual_network_composite_operations.update_route_table_and_wait_for_stat
e(
                    route_table_ocid, rt_details,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_AVAILABLE])
                self.vcn_client.update_route_table(route_table_ocid,
rt_details)
                # Default route table can't be deleted from VCN
                if not ("Default" in route_table_name):
                    print(
                        "Deleting the route table: {0}, with ocid
{1}".format(resource.display_name,

resource.identifier))
                    try:

self.virtual_network_composite_operations.delete_route_table_and_wait_for_stat
e(
                            route_table_ocid,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_TERMINATED])

                        print("Deleted the route table successfully!")
                    except Exception as e:
                        print("Error while deleting the route table {0}, ocid
{1}, Error message {2}".format(
                            resource.display_name, route_table_ocid, str(e)))
                        if "associated with Subnet" in str(e):
                            try:

self.delete_subnet_route_table_association(route_table_ocid)
                                # After removing the association again
retrying the removal of route table
                                # This is for Db subnet route table

self.virtual_network_composite_operations.delete_route_table_and_wait_for_stat
e(
                                    route_table_ocid,

wait_for_states=[oci.core.models.RouteTable.LIFECYCLE_STATE_TERMINATED])
                                print("Deleted the route table successfully!")
                            except Exception as e:
                                print("Error while deleting the route table
after removing the association "
                                      "{0}, ocid {1}, Error message
{2}".format
                                      (resource.display_name,
route_table_ocid, str(e)))

    # Delete Subnet and route table association to remove route table
    def delete_subnet_route_table_association(self, route_table_ocid):
        default_rt_id_in_vcn = ""
        print("Route table is associated with a subnet. Removing the
association between the subnet and route table")
        rt_res = self.vcn_client.get_route_table(route_table_ocid).data
        vcn_id = rt_res.vcn_id
        compartment_id = rt_res.compartment_id
        list_route_rables_vcn =
self.vcn_client.list_route_tables(compartment_id=compartment_id,

vcn_id=vcn_id).data
        for rt in list_route_rables_vcn:
            if "Default Route" in rt.display_name:
                default_rt_id_in_vcn = rt.id
        list_subnets =
self.vcn_client.list_subnets(compartment_id=compartment_id,
vcn_id=vcn_id).data
        for subnet in list_subnets:
            subnet_ocid = subnet.id
            if subnet.route_table_id == route_table_ocid:
                subnet_details = oci.core.models.UpdateSubnetDetails()
                subnet_details.route_table_id = default_rt_id_in_vcn
                try:

self.virtual_network_composite_operations.update_subnet_and_wait_for_state(
                        subnet_ocid, subnet_details,

wait_for_states=[oci.core.models.Subnet.LIFECYCLE_STATE_AVAILABLE])
                    print("Removed the association between the subnet and
route table.")
                except Exception as e:
                    print("Error while removing the association between the
subnet and route table {}".format(str(e)))

    # Delete DHCP Options
    def delete_dhcp_options(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "DHCPOptions":
                dhcp_name = resource.display_name
                dhcp_ocid = resource.identifier
                if not ("Default" in dhcp_name):
                    print(
                        "Deleting the DHCP options: {0}, with ocid
{1}".format(resource.display_name, dhcp_ocid))
                    try:

self.virtual_network_composite_operations.delete_dhcp_options_and_wait_for_sta
te(dhcp_ocid,

                           wait_for_states=[


oci.core.models.DhcpOptions.LIFECYCLE_STATE_TERMINATED])
                        print("Deleted the DHCP options successfully!")
                    except Exception as e:
                        print(
                            "Error while deleting the DHCP options {0}, ocid
{1}, Error message {2} ".format(
                                resource.display_name, dhcp_ocid, str(e)))

    # Delete Internet Gateway
    def delete_internet_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "InternetGateway":
                ig_ocid = resource.identifier
                print("Deleting the Internet Gateway: {0}, with ocid
{1}".format(resource.display_name,

   ig_ocid))
                try:
```

```
self.virtual_network_composite_operations.delete_internet_gateway_and_wait_for
_state(ig_ocid,

                           wait_for_states=[


oci.core.models.InternetGateway.LIFECYCLE_STATE_TERMINATED])

                    print("Deleted the Internet Gateway successfully!")
                except Exception as e:
                    print("Error while deleting the Internet Gateway {0},
ocid {1}, Error message {2}".format(
                        resource.display_name,
                        ig_ocid, str(e)))

    # Delete Service Gateway
    def delete_service_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "ServiceGateway":
                svc_gateway_ocid = resource.identifier
                print("Deleting the service gateway: {0}, with ocid
{1}".format(resource.display_name,

  svc_gateway_ocid))
                try:

self.virtual_network_composite_operations.delete_service_gateway_and_wait_for_
state(
                        svc_gateway_ocid,
wait_for_states=[oci.core.models.ServiceGateway.LIFECYCLE_STATE_TERMINATED])

                    print("Deleted the service gateway successfully!")
                except Exception as e:
                    print("Error while deleting the service gateway {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name,
                        svc_gateway_ocid, str(e)))

    # Delete Local Peering Gateway
    def delete_local_peering_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "LocalPeeringGateway":
                lpg_ocid = resource.identifier
                print("Deleting the local peering gateway: {0}, with ocid
{1}".format(resource.display_name,

        lpg_ocid))
                try:

self.virtual_network_composite_operations.delete_local_peering_gateway_and_wai
t_for_state(
                        lpg_ocid,
wait_for_states=[oci.core.models.LocalPeeringGateway.LIFECYCLE_STATE_TERMINATE
D])

                    print("Deleted local peering gateway successfully!")
                except Exception as e:
                    print("Error while deleting the local peering gateway
{0}, ocid {1}, Error message {2}".format(
                        resource.display_name, lpg_ocid, str(e)))

    # Delete Nat Gateway
    def delete_nat_gateway(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "NatGateway":
                nat_ocid = resource.identifier
                print("Deleting the NAT gateway: {0}, with ocid
{1}".format(resource.display_name,

nat_ocid))
                try:

self.virtual_network_composite_operations.delete_nat_gateway_and_wait_for_stat
e(
                        nat_gateway_id=nat_ocid,

wait_for_states=[oci.core.models.NatGateway.LIFECYCLE_STATE_TERMINATED]
                    )
                    print("Deleted the NAT gateway successfully!")
                except Exception as e:
                    print("Error while deleting the NAT gateway {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, nat_ocid, str(e)))

    # Delete VCN
    def delete_vcn_resources(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "Vcn":
                vcn_ocid = resource.identifier
                vcn_name = resource.display_name
                print("Deleting the VCN: {0}, with ocid {1}".format(vcn_name,
vcn_ocid))
                try:

self.virtual_network_composite_operations.delete_vcn_and_wait_for_state(vcn_oc
id,

              oci.core.models.Vcn.LIFECYCLE_STATE_TERMINATED)
                    print("Deleted the VCN successfully!")
                except Exception as e:
                    print("Error while deleting the VCN {0}, VCN id {1},
Error message {2}".format(vcn_name, vcn_ocid,

                     str(e)))

    # Deleting the Unified Agent Configuration
    def delete_unified_agent_configuration(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "UnifiedAgentConfiguration":
                uac_ocid = resource.identifier
                print("Deleting the unified agent configuration: {}, with
ocid {}".format(resource.display_name,
            uac_ocid))
                try:

self.log_composite_operations.delete_unified_agent_configuration_and_wait_for_
state(
                        uac_ocid,

wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])
                    print("Deleted the unified agent configuration
successfully!")
                except Exception as e:
                    print(
                        "Error while deleting the unified agent configuration
name {0}, ocid {1} - Error message {2}".format(
                            resource.display_name, uac_ocid, str(e)))

    # Delete logs in a Log groups
    def delete_log(self, delete_list, name=None):
        """
        Delete log resources for the service.

        :param delete_list:
        :param name: if name is set, only delete the named log resource,
otherwise delete all log resources in the delete_list
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "LogGroup":
                log_group_ocid = resource.identifier
                list_logs = self.log_client.list_logs(log_group_ocid).data
                for log in list_logs:
                    to_delete = True

                    if name is not None:
                        if log.display_name != name:
                            to_delete = False
                    if to_delete:
                        print("Deleting the log name {0}, with log ocid
{1}".format(log.display_name, log.id))
                        try:

self.log_composite_operations.delete_log_and_wait_for_state(
                                log_group_ocid, log.id,

wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])
                            print("Deleted the log {}
successfully!".format(log.display_name))

                        except Exception as e:
                            print("Error while deleting the log name {}, log
ocid {}, Error message {}".format(
                                log.display_name, log.id, str(e)))

    # Delete Log Group
    def delete_log_group(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "LogGroup":
                log_group_ocid = resource.identifier
                print("Deleting the log group: {0}, with ocid
{1}".format(resource.display_name,

log_group_ocid))
                try:

self.log_composite_operations.delete_log_group_and_wait_for_state(
                        log_group_ocid,
wait_for_states=[oci.logging.models.WorkRequest.STATUS_SUCCEEDED])

                    print("Deleted log group successfully!")
                except Exception as e:
                    print("Error while deleting the log group {0}, ocid {1},
Error message {2}".format(
                        resource.display_name, log_group_ocid, str(e)))

    # Delete the Mount targets
    def delete_mount_targets(self, delete_list):
        mt_client = oci.file_storage.FileStorageClient(config=get_config(),
signer=self.signer)
        mt_composite_operations =
oci.file_storage.FileStorageClientCompositeOperations(mt_client)
        for resource in delete_list:
            if resource.resource_type == "MountTarget":
                mt_ocid = resource.identifier
                print("Deleting the mount target {0}, with ocid
{1}".format(resource.display_name, mt_ocid))
                try:

mt_composite_operations.delete_mount_target_and_wait_for_state(
                        mt_ocid,
wait_for_states=[oci.file_storage.models.MountTarget.LIFECYCLE_STATE_DELETED])
                    print("Deleted the mount target successfully!")
                except Exception as e:
                    print("Error while deleting the mount target {0}, ocid
{1}, Error message {2}".format(
                        resource.display_name, mt_ocid, str(e)))

    # Delete FSS
    def delete_fss(self, delete_list):
        fss_client = oci.file_storage.FileStorageClient(config=get_config(),
signer=self.signer)
        fss_composite_operations =
oci.file_storage.FileStorageClientCompositeOperations(fss_client)
        for resource in delete_list:
            if resource.resource_type == "FileSystem":
                fss_ocid = resource.identifier
                try:
                    # Get the list of exports to delete
                    list_exports =
fss_client.list_exports(file_system_id=fss_ocid).data
                    for export in list_exports:
                        export_ocid = export.id
                        print("Deleting the export id {}".format(export_ocid))

fss_composite_operations.delete_export_and_wait_for_state(
                            export_id=export_ocid,

wait_for_states=[oci.file_storage.models.Export.LIFECYCLE_STATE_DELETED])
                        print("Deleted the exports successfully!")
                except Exception as e:
                    print("Error while deleting the export, Error message
{}".format(str(e)))
                try:
                    print("Deleting the FSS: {0}, with ocid
{1}".format(resource.display_name, fss_ocid))

fss_composite_operations.delete_file_system_and_wait_for_state(
                        fss_ocid,
wait_for_states=[oci.file_storage.models.FileSystem.LIFECYCLE_STATE_DELETED])
                    print("Deleted the FSS successfully!")
                except Exception as e:
                    print("Error while deleting the FSS name {0}, ocid {1},
Error message {2}".format(
                        resource.display_name, fss_ocid, str(e)))

    # Deletion of TagNamespace
    def delete_tag_namespace(self, delete_list):
        for resource in delete_list:
            if resource.resource_type == "TagNamespace":
                tag_ns_name = resource.display_name
                tag_ns_ocid = resource.identifier
                print("Deleting the tag namespace {0}, with ocid
{1}".format(tag_ns_name, tag_ns_ocid))
                try:
                    # Retiring the tag namespace
                    tag_status =
self.identity_client.get_tag_namespace(tag_namespace_id=tag_ns_ocid).data
                    print("Tag namespace: {} and isRetired:
{}".format(tag_ns_name, tag_status.is_retired))

                    if not tag_status.is_retired:
                        print("Retiring the tag namespace
{}".format(tag_ns_name))
                        tag_ns_details =
oci.identity.models.UpdateTagNamespaceDetails()
                        tag_ns_details.is_retired = True

self.identity_client_composite_operations.update_tag_namespace_and_wait_for_st
ate(
                            tag_namespace_id=tag_ns_ocid,
                            update_tag_namespace_details=tag_ns_details,
                            wait_for_states=[

oci.identity.models.TagNamespace.LIFECYCLE_STATE_INACTIVE])
                        tag_status =
self.identity_client.get_tag_namespace(tag_namespace_id=tag_ns_ocid).data
                        print("Tag status before deleting
{}".format(tag_status.is_retired))
```

```
                    print("Deleting the tag namespace {}".format(tag_ns_name))
                    # Tag namespace deletion is taking too long time. So not 
waiting for the completion.

self.identity_client.cascade_delete_tag_namespace(tag_namespace_id=tag_ns_ocid
)
                    print("Asynchronous deletion of Tag namespaces is 
enabled."
                          "Check the deletion status manually. Tag name {0} 
with ocid {1}".format(tag_ns_name,

                    tag_ns_ocid))
                except Exception as e:
                    print("Error while deleting the Tag namespace {0}, ocid 
{1}, Error message {2} "
                          .format(tag_ns_name, tag_ns_ocid, str(e)))

    # Deleting the unattached boot volumes
    def delete_boot_volumes(self, delete_list):
        bv_client = oci.core.BlockstorageClient(config=get_config(), 
signer=self.signer)
        bv_composite_operations = 
oci.core.BlockstorageClientCompositeOperations(bv_client)
        for resource in delete_list:
            if resource.resource_type == "BootVolume" and 
resource.lifecycle_state == "AVAILABLE":
                bv_ocid = resource.identifier
                bv_name = resource.display_name
                print("Deleting the boot volume {}, with ocid {}
".format(bv_name, bv_ocid))
                try:

bv_composite_operations.delete_boot_volume_and_wait_for_state(
                        boot_volume_id=bv_ocid,

wait_for_states=[oci.core.models.BootVolume.LIFECYCLE_STATE_TERMINATED])
                    print("Deleted the boot volume successfully!")
                except Exception as e:
                    print("Error while deleting the boot volume name {}, ocid 
{}, Error message {}".format(bv_name,

                             bv_ocid,

                             str(e)))

    def delete_functions(self, delete_list):
        """
        Deletes OCI functions and function application for the service.

        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "FunctionsApplication":
                fn_app_id = resource.identifier
                fn_app_name = resource.display_name
                try:
                    result = pagination.list_call_get_all_results(
                        self.fn_client.list_functions,
                        fn_app_id
                    )

                    # Delete all functions within function application
                    print("Deleting functions within function application 
{}".format(fn_app_name))
                    for fn in result.data:
                        
self.fn_composite_operations.delete_function_and_wait_for_state(
                            function_id=fn.id,
                            
wait_for_states=[fn_models.Function.LIFECYCLE_STATE_DELETED]
                        )
                        print("Deleted the function {} 
successfully!".format(fn.display_name))

                except Exception as e:
                    print("Error while deleting the functions within 
application id {}, Error message {}".format(fn_app_id, str(e)))


    def delete_functions_app(self, delete_list):
        """
        Deletes function application for the service.

        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "FunctionsApplication":
                fn_app_id = resource.identifier
                try:
                    # Delete the function application
                    
self.fn_composite_operations.delete_application_and_wait_for_state(
                        application_id = fn_app_id,
                        wait_for_states = 
[fn_models.Application.LIFECYCLE_STATE_DELETED]
                    )
                except Exception as e:
                    print("Error while deleting the Functions application 
with id {}, Error message {}".format(fn_app_id, str(e)))

    def delete_event_rules(self, delete_list):
        """
        Deletes the event rules for the service.
        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "EventRule":
                event_rule_id = resource.identifier
                event_rule_name = resource.display_name
                try:
                    print("Deleting event rule {}".format(event_rule_name))

self.events_client_composite_operations.delete_rule_and_wait_for_state(
                        rule_id=event_rule_id,

wait_for_states=[event_models.Rule.LIFECYCLE_STATE_DELETED]
                    )
                    print("Deleted the event rule {}
successfully!".format(event_rule_name))
                except Exception as e:
                    print("Error while deleting the event rule id {}, Error
message {}".format(event_rule_id, str(e)))


    def delete_autoscaling_logs(self, delete_list):
        """
        Deletes the event rule invoke log for the service.

        :param delete_list:
        :return:
        """
        # trim the extra hyphen char if present from the service_name_prefix
        service_name = self.service_name_prefix[:-1] if
self.service_name_prefix[
                                                            -1] == '-' else
self.service_name_prefix
        self.delete_log(delete_list,
name="{0}_event_rule_invoke_log".format(service_name))
        self.delete_log(delete_list,
name="{0}_autoscaling_log".format(service_name))

    def predestroy_autoscaling_resources(self, delete_list):
        """
        Deletes pre-destroy resources associated with autoscaling.
        This is to be invoked when user needs to delete only the resources
created via API during provisioning outside
        of terraform preferably prior to running terraform destroy action.

        :param delete_list:
        :return:
        """
        print("Executing pre-destroy of resources created for autoscaling
feature")
        self.delete_functions(delete_list)
        self.delete_event_rules(delete_list)
        self.delete_autoscaling_logs(delete_list)

    def delete_alarms(self, delete_list):
        """
        Delete all alarms created for autoscaling for the service.

        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "Alarm":
                alarm_id = resource.identifier
                alarm_name = resource.display_name
                try:
                    print("Deleting alarm {}".format(alarm_name))

self.monitoring_composite_operations.delete_alarm_and_wait_for_state(
                        alarm_id=alarm_id,

wait_for_states=[monitoring_models.Alarm.LIFECYCLE_STATE_DELETED]
                    )
                    print("Deleted the alarm {}
successfully!".format(alarm_name))
                except Exception as e:
                    print("Error while deleting the alarm id {}, Error
message {}".format(alarm_id, str(e)))

    def delete_container_repos(self, delete_list):
        """
        Deletes container repos created for autoscaling for the service.
        Deleting the repos also deletes the container images in those repos.

        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "ContainerRepo":
                repo_id = resource.identifier
                repo_name = resource.display_name
                try:
                    print("Deleting container repo {}".format(repo_name))

self.artifacts_composite_operations.delete_container_repository_and_wait_for_s
tate(
                        repository_id=repo_id,

wait_for_states=[artifacts_models.ContainerRepository.LIFECYCLE_STATE_DELETED]
                    )
                    print("Deleted the container repo {}
successfully!".format(repo_name))
                except Exception as e:
                    print("Error while deleting the container repo id {},
Error message {}".format(repo_id, str(e)))

    def delete_notification_topics(self, delete_list):
        """
        Deletes the notification topics created for autoscaling for the
service.
        Deleting the notification topics also deletes the associated
subscriptions for those topics.

        :param delete_list:
        :return:
        """
        for resource in delete_list:
            if resource.resource_type == "OnsTopic":
                topic_id = resource.identifier
                topic_name = resource.display_name
                try:
                    print("Deleting notification topic {}".format(topic_name))
                    self.ons_control_plane_client.delete_topic(
                        topic_id=topic_id
                    )
                    oci.wait_until(self.ons_control_plane_client,
self.ons_control_plane_client.get_topic(topic_id),
                                   'lifecycle_state',
ons_models.NotificationTopic.LIFECYCLE_STATE_DELETING)
                    print("Deleted the notification topic {}
successfully!".format(topic_name))
                except Exception as e:
                    print(
                        "Error while deleting the notification topic id {},
Error message {}".format(topic_id, str(e)))

    def delete_all_autoscaling_resources(self, delete_list):
        """
        Deletes all resources created when autoscaling feature is enabled.

        :param delete_list:
        :return:
        """
        self.predestroy_autoscaling_resources(delete_list)
        self.delete_container_repos(delete_list)
        self.delete_alarms(delete_list)
        self.delete_notification_topics(delete_list)
        self.delete_functions_app(delete_list)


def main():
    parser = argparse.ArgumentParser(description=textwrap.dedent('''
        This script is used for:
        - pre-destroying autoscaling resources (prior to running destroy job
on the stack from OCI Console)
        - delete all infra resources created for the stack (identified by its
service name prefix)
    '''), formatter_class=argparse.RawDescriptionHelpFormatter)
    # Required position params
    parser.add_argument('command', choices=['delete', 'list', 'pre-destroy'],
help='Command')
    parser.add_argument('service_name_prefix', help='Stack service name')

    # Optional params
    parser.add_argument('-r', '--region', default='', help='Region other than
home region must be specified.')
    parser.add_argument('-f', '--feature', default='autoscaling',
help='Feature to run pre-destroy for.')

    args = parser.parse_args()

    command = args.command
    service_prefix = args.service_name_prefix
    global REGION

    REGION = args.region
    feature = args.feature

    print("Service prefix name:" + service_prefix)

    if len(service_prefix) >= 16:
        service_prefix = service_prefix[0:16]
    service_prefix = service_prefix + "-"
    cleanup_util = CleanUpResources(service_prefix)

    if command == 'list':
        print("Listing all resources with service prefix name " + 
service_prefix)
        cleanup_resources = cleanup_util.list_all_resources()
    elif command == 'delete':
        print("Deleting all resources with service prefix name " + 
service_prefix)
        cleanup_resources = cleanup_util.list_all_resources()
        cleanup_util.cleanup_resources(cleanup_resources)
        cleanup_util.delete_dynamic_group()
    elif command == 'pre-destroy' and feature == 'autoscaling':
        print('Deleting pre-destroy autoscaling resources with service prefix 
name ' + service_prefix)
        cleanup_resources = cleanup_util.list_all_resources()
        cleanup_util.predestroy_autoscaling_resources(cleanup_resources)


if __name__ == '__main__':
    main()
```

D
Migrate Terraform Scripts

You can access the Terraform Scripts of an Oracle WebLogic Server for OCI and modify it as required. See Terraform Scripts in Oracle WebLogic Server for OCI.

Support ended for Terraform v0.11.x. So, you would not be able to scale out or destroy the stacks created with Terraform v0.11.x. However, you can configure the stacks in your local environment by downloading the terraform configuration, variables, and terraform state file from your Oracle Resource Manager (ORM) stacks. You can continue using terraform v0.11.x with the downloaded scripts and manage their stacks outside ORM.

Complete the following steps:

1. Download the previous versions of the terraform from https://releases.hashicorp.com/terraform/.
2. Install the terraform on the host, where you want to run the scripts.
3. Download the configuration and terraform state file from the last apply job:
   a. Click the navigation menu ☰, and select **Developer Services**. Under the **Resource Manager** group, click **Stacks**.
   b. Select the **Compartment** that contains your stack.
   c. Click the name of your stack.
   d. In the **Jobs** section, select the latest apply job.
   e. Click **Download Terraform Configuration** and save the file to the preferred location.
   f. Click **Download Terraform State** and save the file to the preferred location
4. Extract variables from the stack.
   Run the following OCI CLI command to generate the variables in json format:

   ```
   #ensure that oci cli and jq are installed in cloud shell.
   <user>@cloudshell:~ (<domain>-1)$ oci resource-manager stack get --stack-
   id <OCID> | jq -r .data.variables
   ```

5. Copy the variables into the `.json` file.
   You can edit the file to reapply parameters, like, node count, VM shape, SSH keys, and so on. For information about the variables, see Variables in Terraform Scripts.
6. Unzip the terraform configuration files to the working folder (`~/wlsoci/`).
7. Copy the terraform state and the varaibles.json fles to the working folder (`~/wlsoci/`).
8. As required, by using reapply, scale out or destroy the stacks. See Invoke Terraform Scripts.