Oracle® Fusion Middleware

High Availability Guide for Oracle Identity and Access Management

11*g* Release 2 (11.1.2.2)

E28391-08

May 2014

Documentation for administrators, developers, and others that describes high availability concepts as well as administration and configuration procedures to deploy and manage Oracle Fusion Middleware with high availability requirements.

Oracle Fusion Middleware High Availability Guide for Oracle Identity and Access Management 11*g* Release 2 (11.1.2.2)

E28391-08

Primary Author: Christine Ford

Contributing Author: Fermin Castro, Michael Blevins, Pradeep Bhat, Michael Rhys, Samrat Ray, Jeni Ferns, Richard Delval, Bharath K. Reddy, Susan Kornberg, Joe Paul, Ajay Keni, Amit Sharma, Vasuki Ashok, Olaf Stullich, Jingjing Wei, Olfat Aly, Eileen He, Ramaprakash Sathyanarayan, Apurv Chandra, Atika Jain, Buddhika Kottahachchi, Arun Theebaprakasam, Rajiv Jaisankar, Vinay Kalra, Daniel Shih, Pratima Gogineni, Russ Hodgson, Christopher Johnson, Pramodini Gattu, Gururaj BS, Ben Gelernter, Peter Jacobsen.

Contributor: Van Sioung Ng Yan Tun, Eric Cloney, Fiona Zheng, Ruchica Behl, David J. Jones, Ellen Desmond, Don Biasotti, Vinaye Misra, Dhaval Shah, Herbert Stiel, Albert Tam, Pushkar Kapasi, Shail Goel, Rahul Menezes, Satheesh Amilineni, Ratheesh Pai, Jan Carlin.

Contents

3 High Availability for WebLogic Server

4 Considerations for High Availability Oracle Database Access

5 Configuring High Availability for Oracle Identity Manager Components

6 Configuring High Availability for Oracle Access Management Access Manager Components

8 Configuring High Availability for Oracle Access Management Security Token Service

9 Configuring High Availability for Identity Federation Components

10 Configuring High Availability for Oracle Entitlements Server

11 Configuring High Availability for Mobile and Social

12 Configuring High Availability for Oracle Privileged Account Manager Components

13 Configuring High Availability for Oracle Identity Navigator

14 Oracle Unified Directory

15 Active-Passive Topologies for Oracle Fusion Middleware High Availability

Preface

This preface contains these sections:

- Intended Audience
- Documentation Accessibility
- Related Documentation
- Conventions

Intended Audience

The *Oracle Fusion Middleware High Availability Guide for Oracle Identity and Access Management* is intended for administrators, developers, and others whose role is to deploy and manage Oracle Fusion Middleware with high availability requirements.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documentation

For more information, see these Oracle resources:

- *Oracle Fusion Middleware Concepts*
- *Oracle Fusion Middleware Administrator's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction to High Availability

A high availability architecture is one of the key requirements for any Enterprise Deployment. Oracle Fusion Middleware has an extensive set of high availability features, which protect its components and applications from unplanned down time and minimize planned downtime.

The solutions and procedures that this book describes are designed to eliminate single points of failure for Oracle Fusion Middleware components with no or minimal down time. These solutions help ensure that applications deployed with Oracle Fusion Middleware meet the required availability to achieve your business goals.

Oracle Identity and Access Management enables enterprises to manage the end-to-end lifecycle of user identities across all enterprise resources. You can deploy applications faster, apply the most granular protection to enterprise resources, and much more. This guide describes how to configure Identity and Access Management (IAM) products for high availability in an active-active configuration.

This chapter explains high availability and its importance from the perspective of Oracle Fusion Middleware. This chapter includes the following sections:

- Section 1.1, "What is High Availability"
- Section 1.2, "How To Use This Guide"
- Section 1.3, "High Availability Information in Other Documentation"

1.1 What is High Availability

High availability refers to the ability of users to access a system without loss of service. Deploying a high availability system minimizes the time when the system is down, or unavailable and maximizes the time when it is running, or available. This section provides an overview of high availability from a problem-solution perspective. This section includes the following topics:

- Section 1.1.1, "High Availability Problems"
- Section 1.1.2, "High Availability Solutions"

1.1.1 High Availability Problems

Mission critical computer systems need to be available 24 hours a day, 7 days a week, and 365 days a year. However, part or all of the system may be down during planned or unplanned downtime. A system's availability is measured by the percentage of time that it is providing service in the total time since it is deployed. Table 1–1 provides an example.

Table 1–1 Availability Percentages and Corresponding Downtime Values

Availability Percentage	Approximate Downtime Per Year
95%	18 days
99%	4 days
99.9%	9 hours
99.99%	1 hour
99.999%	5 minutes

System downtime may be categorized as planned or unplanned. Unplanned downtime is any sort of unexpected failure. Planned downtime refers to scheduled operations that are known in advance and that render the system unavailable. The effect of planned downtime on end users is typically minimized by scheduling operational windows when system traffic is slow. Unplanned downtime may have a larger effect because it can happen at peak hours, causing a greater impact on system users.

These two types of downtimes (planned and unplanned) are usually considered separately when designing a system's availability requirements. A system's needs may be very restrictive regarding its unplanned downtimes, but very flexible for planned downtimes. This is the typical case for applications with high peak loads during working hours, but that remain practically inactive at night and during weekends. You may choose different high availability features depending on the type of failure is being addressed.

1.1.2 High Availability Solutions

High availability solutions can be categorized into local high availability solutions that provide high availability in a single data center deployment, and disaster recovery solutions, which are usually geographically distributed deployments that protect your applications from disasters such as floods or regional network outages.

Among possible types of failures, process, node, and media failures as well as human errors can be protected by local high availability solutions. Local physical disasters that affect an entire data center can be protected by geographically distributed disaster recovery solutions.

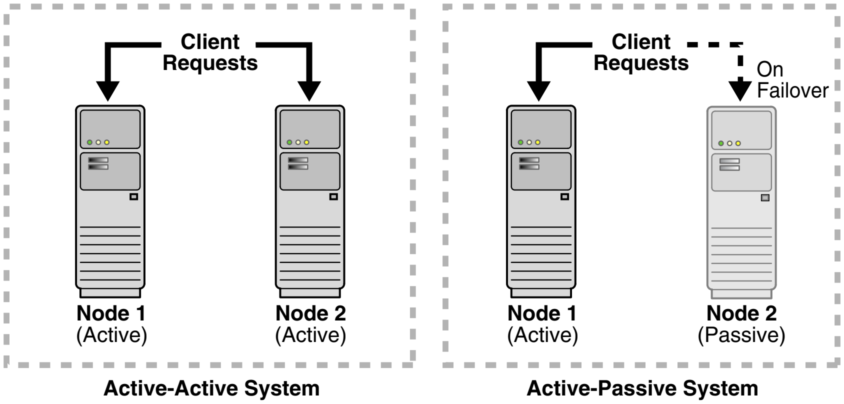
To solve the high availability problem, a number of technologies and best practices are needed. The most important mechanism is redundancy. High availability comes from redundant systems and components. You can categorize local high availability solutions by their level of redundancy, into active-active solutions and active-passive solutions (see Figure 1–1):

- **Active-active solutions** deploy two or more active system instances and can be used to improve scalability and provide high availability. In active-active deployments, all instances handle requests concurrently.
- **Active-passive solutions** deploy an active instance that handles requests and a passive instance that is on standby. In addition, a heartbeat mechanism is set up between these two instances. This mechanism is provided and managed through operating system vendor-specific clusterware. Generally, vendor-specific cluster agents are also available to automatically monitor and failover between cluster nodes, so that when the active instance fails, an agent shuts down the active instance completely, brings up the passive instance, and application services can successfully resume processing. As a result, the active-passive roles are now switched. The same procedure can be done manually for planned or unplanned

downtime. Active-passive solutions are also generally referred to as cold failover clusters.

You can use Oracle Cluster Ready Services (CRS) to manage the Fusion Middleware Active-Passive (CFC) solutions.

Figure 1–1 Active-Active and Active-Passive High Availability Solutions



In addition to architectural redundancies, the following local high availability technologies are also necessary in a comprehensive high availability system:

- **Process death detection and automatic restart**

  Processes may die unexpectedly due to configuration or software problems. A proper process monitoring and restart system should monitor all system processes constantly and restart them should problems appear.

  A system process should also maintain the number of restarts within a specified time interval. This is also important since continually restarting within short time periods may lead to additional faults or failures. Therefore a maximum number of restarts or retries within a specified time interval should also be designed as well.

- **Clustering**

  Clustering components of a system together enables the components to be viewed functionally as a single entity from the perspective of a client for runtime processing and manageability. A cluster is a set of processes running on single or multiple computers that share the same workload. There is a close correlation between clustering and redundancy. A cluster provides redundancy for a system.

  If failover occurs during a transaction in a clustered environment, the session data is retained as long as there is at least one surviving instance available in the cluster.

- **State replication and routing**

  For stateful applications, client state can be replicated to enable stateful failover of requests in the event that processes servicing these requests fail.

- **Failover**

  With a load-balancing mechanism in place, the instances are redundant. If any of the instances fail, requests to the failed instance can be sent to the surviving instances.

- **Server load balancing**

When multiple instances of identical server components are available, client requests to these components can be load balanced to ensure that the instances have roughly the same workload.

- **Server Migration**

  Some services can only have one instance running at any given point of time. If the active instance becomes unavailable, the service is automatically started on a different cluster member. Alternatively, the whole server process can be automatically started on a different system in the cluster.

- **Integrated High Availability**

  Components depend on other components to provide services. The component should be able to recover from dependent component failures without any service interruption.

- **Rolling Patching**

  Patching product binaries often requires down time. Patching a running cluster in a rolling fashion can avoid downtime. Patches can be uninstalled in a rolling fashion as well.

- **Configuration management**

  A clustered group of similar components often need to share common configuration. Proper configuration management ensures that components provide the same reply to the same incoming request, enables these components to synchronize their configurations, and provides high availability configuration management for less administration downtime.

- **Backup and Recovery**

  User errors may cause a system to malfunction. In certain circumstances, a component or system failure may not be repairable. A backup and recovery facility should be available to back up the system at certain intervals and restore a backup when an unrepairable failure occurs.

1.2 How To Use This Guide

This guide discusses the architecture, interaction, and dependencies of Oracle Fusion Middleware components in 11*g* Release 2 (11.1.2.2), and explains how you can deploy them in a high availability architecture.

This section describes changes in this guide and where to find component documentation for 11*g* Releases 1 and 2.

- Section 1.2.1, "What's New In this Guide"
- Section 1.2.2, "Oracle Fusion Middleware Documentation Libraries"

1.2.1 What's New In this Guide

Previous versions of the High Availability Guide covered components that are not Identity and Access Management (IAM) components, such as Oracle SOA Suite. This version of the guide describes only Oracle Identity and Access Management 11*g* Release 2 components. See the following tables for more information.

Note: There is no high availability guide for Oracle Forms and Reports 11*g* Release 2. Oracle recommends that you refer to the chapter "Configuring High Availability for Oracle Portal, Forms, Reports, and Discoverer" in the 11*g* Release 1 High Availability Guide.

- Table 1–2 lists ***11g* R2 Components** components that this guide includes.
- Table 1–3 lists the ***11g* R2 Components that are not in this guide.**
- Table 1–4 lists components that the Release 1 *Oracle Fusion Middleware High Availability Guide for Oracle Identity and Access Management* includes.

Table 1–2 11g R2 Components in the 11g R2 High Availability Guide

11*g* Release 2 (11.1.2.2) Components	**See...**
Oracle Identity Manager (OIM)	Chapter 5, "Configuring High Availability for Oracle Identity Manager Components"
Oracle Access Management Access Manager (OAM)	Chapter 6, "Configuring High Availability for Oracle Access Management Access Manager Components"
Oracle Adaptive Access Manager (OAAM)	Chapter 7, "Configuring High Availability for Oracle Adaptive Access Manager Components"
Oracle Access Management Security Token Service (STS)	Chapter 8, "Configuring High Availability for Oracle Access Management Security Token Service"
Oracle Access Management Identity Federation (IF)	Chapter 9, "Configuring High Availability for Identity Federation Components"
Oracle Entitlements Server (OES)	Chapter 10, "Configuring High Availability for Oracle Entitlements Server"
Oracle Access Management Mobile and Social	Chapter 11, "Configuring High Availability for Mobile and Social"
Oracle Privileged Account Manager (OPAM)	Chapter 12, "Configuring High Availability for Oracle Privileged Account Manager Components"
Oracle Identity Navigator (OIN)	Chapter 13, "Configuring High Availability for Oracle Identity Navigator"
Oracle Unified Directory (OUD)	Chapter 14, "Oracle Unified Directory"

Table 1–3 11g R2 Components not in the 11g R2 High Availability Guide

11*g* Release 2 (11.1.2.2) Component	**See this chapter in the *Oracle Fusion Middleware High Availability Guide***, 11g Release 1
Oracle Forms and Reports	"Oracle Portal, Forms, Reports, and Discoverer"

For all remaining Oracle Fusion Middleware products not released with 11*g* Release 2 (11.1.2.2), use the latest Oracle Fusion Middleware High Availability Guide, 11g Release 1. Table 1–4 lists components that 11*g* Release 1 includes:

Table 1–4 Components in the 11g Release 1 High Availability Guide

Components	**See this chapter in the *Oracle Fusion Middleware High Availability Guide***, 11g Release 1
Oracle SOA Suite, including: Oracle SOA Service Infrastructure, Oracle BPEL Process Manager, Oracle BPM Suite, Oracle Mediator, Oracle JCA Adapters, Oracle B2B, Oracle Web Services Manager (WSM), Oracle User Messaging Service (UMS)	"Configuring High Availability for Oracle SOA Suite"
Oracle ADF and Oracle WebCenter Portal	"Configuring High Availability for Oracle ADF and Oracle WebCenter Portal"
Oracle Data Integrator (ODI)	"Configuring High Availability for Oracle Data Integrator (ODI)"
Oracle Business Intelligence (BI) and EPM	"Configuring High Availability for Oracle Business Intelligence and EPM"
Oracle Web Tier	"Configuring High Availability for Web Tier Components"
Oracle WebCenter Content	"Configuring High Availability for WebCenter Content"
Oracle Portal and Discoverer	"Configuring High Availability for Oracle Portal, Forms, Reports, and Discoverer"

1.2.2 Oracle Fusion Middleware Documentation Libraries

This section provides links to the documentation library for the Oracle Fusion Middleware release(s) that you are running.

Table 1–5 Oracle Fusion Middleware Documentation Libraries

If you are running...	**See this library for documentation...**
Oracle Fusion Middleware 11g Release 1	Release 1, 11.1.1.8
Oracle Fusion Middleware 11*g* Release 2 (11.1.2.2)	Release 2, 11.1.2.2

1.3 High Availability Information in Other Documentation

Table 1–6 lists Oracle Fusion Middleware guides (other than this guide) that contain high availability information. This information pertains to high availability of various Oracle Fusion Middleware components.

Table 1–6 High Availability Information in Oracle Fusion Middleware Documentation

Component	Location of Information
Oracle SOA Suite	*Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite* *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*
Oracle WebCenter Portal	*Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* *Oracle Fusion Middleware Installation Guide for Oracle WebCenter* *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter Portal*
Oracle ADF	*Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*
Oracle Data Integrator	*Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator* *Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator* *Oracle Fusion Middleware Knowledge Module Developer's Guide for Oracle Data Integrator*
Oracle WebLogic Server Clusters	*Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*
Oracle Fusion Middleware Backup and Recovery	*Oracle Fusion Middleware Administrator's Guide*
Oracle Web Cache	*Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*
Oracle Identity Management	*Oracle Fusion Middleware Installation Guide for Oracle Identity Management* *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management*
Oracle Virtual Directory	*Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*
Oracle HTTP Server	*Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server*
Oracle Internet Directory	*Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*
Oracle Access Manager	*Oracle Fusion Middleware Administrator's Guide for Oracle Access Manager with Oracle Security Token Service*
Oracle Authorization Policy Manager	*Oracle Fusion Middleware Authorization Policy Manager Administrator's Guide*
Oracle Identity Manager	*Oracle Fusion Middleware System Administrator's Guide for Oracle Identity Manager*
Oracle Adaptive Access Manager	*Oracle Fusion Middleware Administrator's Guide for Oracle Adaptive Access Manager*
Oracle Real Application Clusters (Oracle RAC)	*Oracle Real Application Clusters Installation Guide*
Oracle WebCenter Content	*Oracle Fusion Middleware Overview Guide for Oracle Enterprise Content Management*
Oracle WebCenter Content: Imaging	*Oracle WebCenter Content Administrator's Guide for Imaging*
Oracle WebCenter Content	*Oracle WebCenter Content System Administrator's Guide for Content Server*
Oracle WebCenter Content: Records	*Oracle Fusion Middleware Administrator's Guide for Universal Records Management*
Oracle Repository Creation Utility (RCU)	*Oracle Fusion Middleware Repository Creation Utility User's Guide*

Table 1–6 (Cont.) High Availability Information in Oracle Fusion Middleware Documentation

Component	Location of Information
Oracle Portal	*Oracle Fusion Middleware Administrator's Guide for Oracle Portal*
Oracle Forms	*Oracle Fusion Middleware Forms Services Deployment Guide*
Oracle Reports	*Oracle Fusion Middleware Oracle Reports User's Guide to Building Reports*
Oracle Business Intelligence Discoverer	*Oracle Fusion Middleware Administrator's Guide for Oracle Business Intelligence Discoverer*
Oracle Business Intelligence Enterprise Edition	*Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*
Oracle Real-Time Decisions	*Oracle Fusion Middleware Administrator's Guide for Oracle Real-Time Decisions*

Part I

High Availability Features and Capabilities

Part I provides information on Oracle Fusion Middleware high availability features, capabilities, and database access.

This part contains the following chapters:

- Chapter 2, "Oracle Fusion Middleware High Availability Framework"
- Chapter 3, "High Availability for WebLogic Server"
- Chapter 4, "Considerations for High Availability Oracle Database Access"

2 Oracle Fusion Middleware High Availability Framework

This chapter describes the Oracle Fusion Middleware features that are important in high availability topologies. It contains the following topics:

2.1 Key Oracle Fusion Middleware Concepts

Oracle Fusion Middleware provides two types of components:

- Java components: A Java component is a peer of a system component but is deployed as one or more Java EE applications and a set of resources. Java components are deployed to an WebLogic Server domain as part of a domain template. Examples of Java components are Oracle SOA Suite and Oracle WebCenter Portal: Spaces.
- System components: A system component is a manageable process that is not deployed as a Java application. Instead, a system component is managed by the Oracle Process Manager and Notification (OPMN). Examples of system components include Oracle HTTP Server and Oracle Internet Directory.

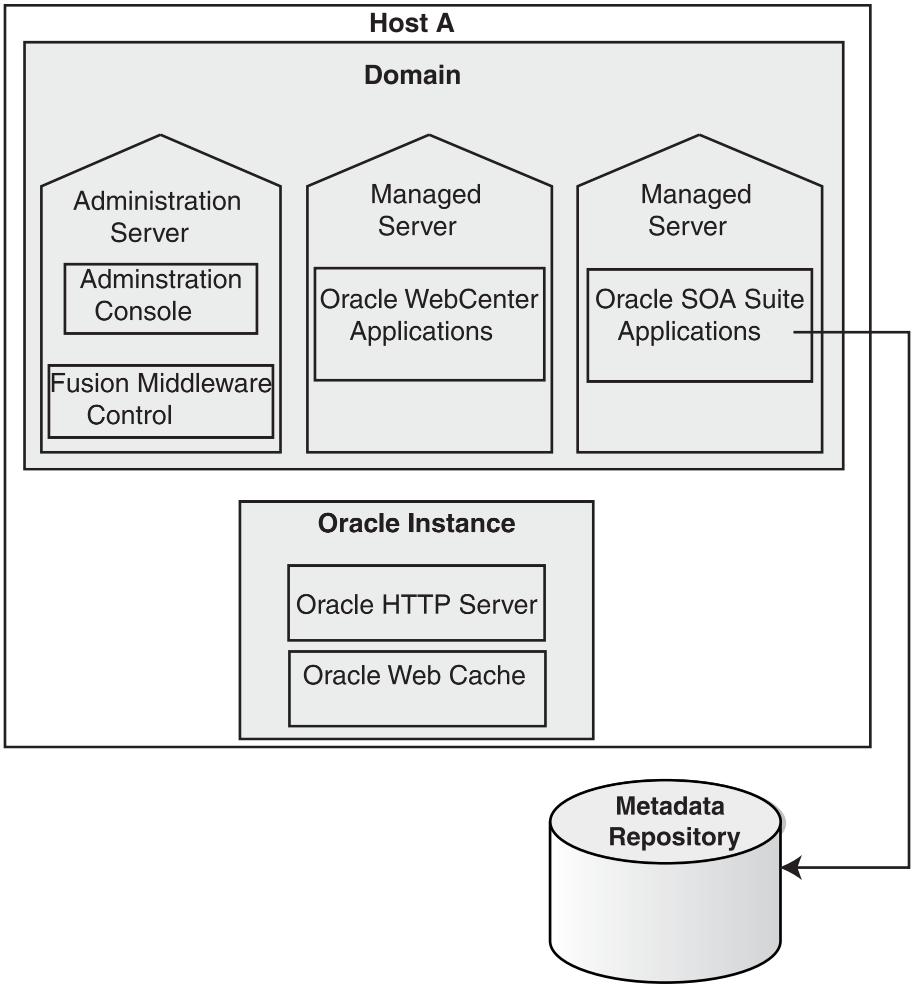
A Java component and a system component are peers.

After you install and configure Oracle Fusion Middleware, your Oracle Fusion Middleware environment contains the following:

- An WebLogic Server domain, which contains one Administration Server and one or more managed servers.
- If your environment includes system components, one or more system component domains.
- An Oracle Metadata Repository, if the components you installed require one.

Figure 2–1 shows an Oracle Fusion Middleware environment with an Oracle WebLogic Server domain with an Administration Server and two managed servers, a system component domain, and an Oracle Metadata Repository.

Figure 2–1 Oracle Fusion Middleware Environment



Your environment also includes a Middleware home, which consists of the Oracle WebLogic Server home, and, optionally, one or more Oracle homes.

2.1.1 What is a WebLogic Server Domain?

A WebLogic Server administration **domain** is a logically related group of Java components. A domain includes a special WebLogic Server instance called the **Administration Server,** which is the central point from which you configure and manage all resources in the domain. Usually, you configure a domain to include additional WebLogic Server instances called *managed servers*. You deploy Java components, such as Web applications, EJBs, and Web services, and other resources to the managed servers and use the Administration Server for configuration and management purposes only.

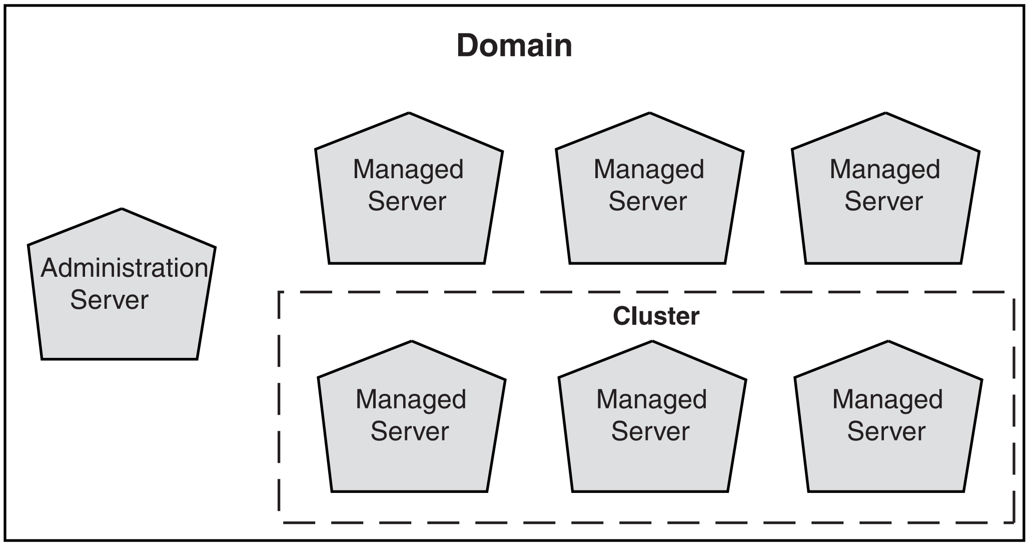
Managed servers in a domain can be grouped together into a cluster.

An WebLogic Server Domain is a peer of a system component domain. Both contain specific configurations outside of their Oracle homes.

The directory structure of an WebLogic Server domain is separate from the directory structure of the WebLogic Server Home. It can reside anywhere; it need not be within the Middleware home directory.

Figure 2–2 shows a Oracle WebLogic Server domain with an Administration Server, three standalone managed servers, and three managed servers in a cluster.

Figure 2–2 Oracle WebLogic Server Domain



> **See Also:** *Oracle Fusion Middleware Understanding Domain Configuration for Oracle WebLogic Server* for more information about domain configuration

The following topics describe entities in the domain:

- What Is the Administration Server?
- About Managed Servers and Managed Server Clusters
- What Is Node Manager?

2.1.1.1 What Is the Administration Server?

The **Administration Server** operates as the central control entity for the configuration of the entire domain. It maintains the domain's configuration documents and distributes changes in the configuration documents to managed servers. You can use the Administration Server as a central location from which to monitor all resources in a domain.

Each WebLogic Server domain must have one server instance that acts as the Administration Server.

To interact with the Administration Server, you can use the Oracle WebLogic Server Administration Console, Oracle WebLogic Scripting Tool (WLST), or create your own JMX client. In addition, you can use Oracle Enterprise Manager Fusion Middleware Control for some tasks.

Fusion Middleware Control and the WebLogic Administration Console run in the Administration Server. Fusion Middleware Control is a Web-based administration console used to manage Oracle Fusion Middleware, including components such as Oracle HTTP Server, Oracle SOA Suite and Oracle WebCenter Portal, Oracle Portal, Forms, Reports, and Discoverer, and the Oracle Identity Management components. Oracle WebLogic Server Administration Console is the Web-based administration console used to manage the resources in an Oracle WebLogic Server domain, including the Administration Server and managed servers in the domain.

2.1.1.2 About Managed Servers and Managed Server Clusters

Managed servers host business applications, application components, Web services, and their associated resources. To optimize performance, managed servers maintain a read-only copy of the domain's configuration document. When a managed server starts up, it connects to the domain's Administration Server to synchronize its configuration document with the document that the Administration Server maintains.

When you create a domain, you create it using a particular domain template. That template supports a particular component or group of components, such as the Oracle SOA Suite. The Managed Servers in the domain are created specifically to host those particular Oracle Fusion Middleware system components.

Java-based Oracle Fusion Middleware system components (such as Oracle SOA Suite, Oracle WebCenter Portal, and some Identity Management components) and customer-developed applications are deployed to Managed Servers in the domain.

If you want to add other components, such as Oracle WebCenter Portal, to a domain that was created using a template that supports another component, you can extend the domain by creating additional Managed Servers in the domain, using a domain template for the component which you want to add.

For production environments that require increased application performance, throughput, or high availability, you can configure two or more Managed Servers to operate as a cluster. A **cluster** is a collection of multiple WebLogic Server server instances running simultaneously and working together to provide increased scalability and reliability. In a cluster, most resources and services are deployed identically to each Managed Server (as opposed to a single Managed Server), enabling failover and load balancing. A single domain can contain multiple WebLogic Server clusters and multiple Managed Servers that are not configured as clusters. The key difference between clustered and non-clustered Managed Servers is support for failover and load balancing. These features are available only in a cluster of Managed Servers.

> **See Also:** Understanding WebLogic Server Clustering" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*

2.1.1.3 What Is Node Manager?

Node Manager is a Java utility that runs as separate process from WebLogic Server and enables you to perform common operations for a Managed Server, regardless of its location with respect to its Administration Server. While use of Node Manager is optional, it provides valuable benefits if your WebLogic Server environment hosts applications with high-availability requirements.

If you run Node Manager on a system that hosts Managed Servers, you can start and stop the Managed Servers remotely using the Administration Console or the command line. Node Manager can also automatically restart a Managed Server after an unexpected failure.

> **See Also:** *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*

2.1.2 What Is a System Component Domain?

A **system component domain** contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. The system components in a system component domain must reside on the same system. A system component domain directory contains files that can be updated, such as configuration files, log files, and temporary files.

A system component domain is a peer of a WebLogic Server domain. Both contain specific configurations outside of their Oracle homes.

The directory structure of a system component domain is separate from the directory structure of the Oracle home. It can reside anywhere; it need not be within the Middleware home directory.

2.1.3 What Is a Middleware Home?

A **Middleware home** consists of the Oracle WebLogic Server home, and, optionally, one or more Oracle homes.

A Middleware home can reside on a local file system or on a remote shared disk that is accessible through NFS.

See Section 2.1.4, "What Is an Oracle Home?" for information about Oracle homes. See Section 2.1.1, "What is a WebLogic Server Domain?" for information about Oracle WebLogic Server homes.

In a high availability installation where two or more hosts are clustered, the Middleware Home binaries must be installed into directories with the exact same directory paths on all the hosts in the cluster.

2.1.4 What Is an Oracle Home?

An **Oracle home** contains installed files necessary to host a specific product. For example, the SOA Oracle home contains a directory that contains binary and library files for Oracle SOA Suite.

An Oracle home resides within the directory structure of the Middleware home. Each Oracle home can be associated with multiple system component domains or Oracle WebLogic Server domains.

In a high availability installation where two or more hosts are clustered, the Oracle Home binaries must be installed into directories with the exact same directory paths on all the hosts in the cluster.

2.1.4.1 What Is an Oracle Common Home?

The **Oracle Common home** contains the binary and library files required for the Oracle Enterprise Manager Fusion Middleware Control and Java Required Files (JRF). There can be only one Oracle Common home within each Middleware home.

2.1.5 What Is a WebLogic Server Home?

A WebLogic Server home contains installed files necessary to host a WebLogic Server. The WebLogic Server home directory is a peer of Oracle home directories and resides within the directory structure of the Middleware home.

2.2 Oracle Fusion Middleware High Availability Terminology

The definitions of terms listed in this section are useful in helping to understand the concepts presented in this book:

- **failover**: When a member of a high availability system fails unexpectedly (unplanned downtime), the system undergoes a failover operation to continue offering services to its consumers. For an *active-active* system, the load balancer entity serving requests to the active members performs the failover. If an active member fails, the load balancer detects the failure and automatically redirects requests for the failed member to surviving active members.

  For an *active-passive* system, the passive member activates during the failover operation and consumers are directed to it. You can perform the failover process or automate it by setting up hardware cluster services to detect failures and move cluster resources from the failed node to the standby node.

- **failback**: A planned operation after unplanned downtime. After a system undergoes a successful failover operation, you can repair the original failed member and re-introduce into the system as a standby member. You can initiate a failback process to activate this member and deactivate the other. This process reverts the system back to its pre-failure configuration.

- **shared storage**: Although each node in a cluster is a standalone server that runs its own set of processes, there are some file system based data and configuration elements which need uniform access from all nodes in a cluster. Shared storage refers to the ability of the cluster to be able to access the same storage, usually disks, from any node in the cluster.

  For SAN based deployments, a clustered file system, such as OCFS, may also be needed. Some examples of this usage are, JMS file based persistence store, transaction logs persistence store, domain configuration in case of cold failover cluster setup.

- **primary node**: The node that is actively running Oracle Fusion Middleware at any given time in a cluster. If this node fails, Oracle Fusion Middleware fails over to the secondary node. Because the primary node runs the active Oracle Fusion Middleware installation(s), if this node fails, Oracle Fusion Middleware fails over to the secondary node. See the definition for secondary node in this section.

- **secondary node**: When the primary node that runs the active Oracle Fusion Middleware installation(s) fails, Oracle Fusion Middleware fails over to this node. See the definition for primary node in this section.

- **network hostname**: A name assigned to an IP address either through the `/etc/hosts` file or through DNS resolution. This name is visible in the network that the system to which it refers to is connected. Often, the network hostname and physical hostname are identical. However, each system has only one physical hostname but may have multiple network hostnames. Thus, a system's network hostname may not always be its physical hostname.

- **physical hostname**: This guide differentiates between the terms physical hostname and network hostname. This guide uses physical hostname to refer to the internal name of the current system. On UNIX, this is the name returned by the `hostname` command.

- **switchover and switchback**: Planned operations. During normal operation, active members of a system may require maintenance or upgrading. You can start a switchover process to enable a substitute member to take over the workload performed by the member that requires maintenance, upgrading, or any planned downtime. The switchover operation ensures continued service to system consumers.

When the maintenance or upgrade is complete, you can perform a switchback operation to activate the upgraded member and bring the system back to the pre-switchover configuration.

- **virtual IP**: To present a single system view of a cluster to network clients, a virtual IP serves as an entry point IP address to the group of servers that are cluster members. You can assign a virtual IP to a server load balancer or a hardware cluster.

  A load balancer also uses a virtual IP as the entry point to a set of servers. These servers tend to be active at the same time. This virtual IP address is not assigned to any individual server but to the load balancer which acts as a proxy between servers and their clients.

- **virtual hostname**: In a cluster, a network hostname assigned to virtual IP bound to one of the nodes in the cluster at any given time.

> **Note:** When this document uses the term *virtual hostname*, it is assumed to be associated with a virtual IP address. In cases where just the IP address is needed or used, it is explicitly stated.

- **hardware cluster**: A collection of computers that provides a single view of network services (such as an IP address) or application services (such as databases, Web servers) to clients of these services. Each node in a hardware cluster is a standalone server that runs its own processes. These processes can communicate with one another to form what looks like a single system that cooperatively provides applications, system resources, and data to users.

  A hardware cluster achieves high availability and scalability through the use of specialized hardware (cluster interconnect, shared storage) and software (health monitors, resource monitors). (The cluster interconnect is a private link that the hardware cluster uses for heartbeat information to detect node death.) Due to the need for specialized hardware and software, hardware clusters are commonly provided by hardware vendors such as Sun, HP, IBM, and Dell. While the number of nodes that can be configured in a hardware cluster is vendor dependent, for the purpose of Oracle Fusion Middleware high availability, only two nodes are required. Hence, this document assumes a two-node hardware cluster for high availability solutions employing a hardware cluster.

- **cluster agent**: The software that runs on a node member of a hardware cluster that coordinates availability and performance operations with other nodes. Clusterware provides resource grouping, monitoring, and the ability to move services. A cluster agent can automate the service failover.

- **clusterware**: A software that manages the operations of cluster members as a system. It enables you to define a set of resources and services to monitor using a heartbeat mechanism between cluster members and to move these resources and services to a different member in the cluster as efficiently and transparently as possible.

2.3 Oracle Fusion Middleware High Availability Solutions

This section describes local high availability concepts, and Oracle Fusion Middleware high availability technologies. This section includes the following topics:

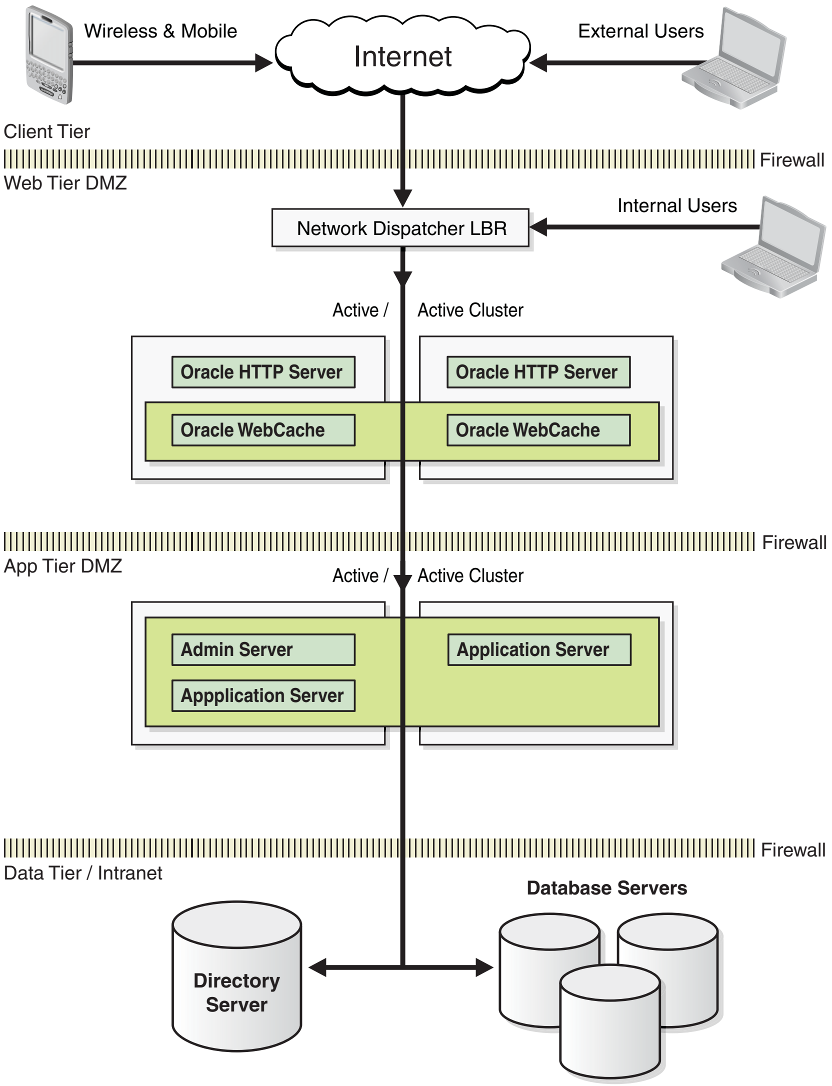
- Section 2.3.1, "Local High Availability"
- Section 2.3.2, "Oracle Fusion Middleware High Availability Technologies"
- Section 2.3.3, "Active-Passive Deployment"
- Section 2.3.4, "About Active-Active and Active-Passive Solutions"
- Section 2.3.5, "Disaster Recovery"

2.3.1 Local High Availability

Local high availability solutions can be categorized as either active-active or active-passive solutions. Oracle Fusion Middleware supports both active-active deployments and active-passive deployments.

Figure 2–3 shows an Oracle Fusion Middleware high availability active-active deployment topology.

Figure 2–3 Oracle Fusion Middleware Enterprise Deployment Architecture



As shown in Figure 2–3, this topology represents a multi-tiered architecture. Users access the system from the client tier. Requests go through a hardware load balancer, which then routes them to a Web server cluster that is running Oracle HTTP Server and Oracle Web Cache in the web tier. Web servers use Proxy Plug-in (`mod_wl_ohs.conf`) to route the requests to the WebLogic cluster in the application tier. Applications running on the WebLogic cluster in the application tier then interact with the database cluster in the data tier to service the request.

There is no single point of failure in the entire architecture. WebLogic Administration Server is configured in Cold Failover Cluster mode, as described in Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster," and is protected using external clusterware.

2.3.2 Oracle Fusion Middleware High Availability Technologies

The Oracle Fusion Middleware infrastructure has these high availability features:

- **Process death detection and automatic restart**

  For Java EE components running on WebLogic Server, Node Manager monitors the Managed Servers. If a Managed Server goes down, Node Manager tries to restart it for a configured number of times.

  For system components, OPMN monitors the processes. If a system component process goes down, OPMN attempts to restart it for a configurable number of times.

- **Clustering**

  Oracle Fusion Middleware Java EE components leverage underlying powerful WebLogic Server clustering capabilities to provide clustering. Oracle Fusion Middleware uses WebLogic clustering capabilities, such as redundancy, failover, session state replication, cluster-wide JNDI services, Whole Server Migration, and cluster wide configuration.

  These capabilities provide for seamless failover of all Java EE Oracle Fusion Middleware system components transparent to the client preserving session and transaction data as well as ensuring data consistency. For further description of these features, see Chapter 3, "High Availability for WebLogic Server."

  System components can also be deployed in a run time cluster. They are typically front-ended by a load balancer to route traffic.

- **State replication and routing**

  Oracle WebLogic Server can be configured for replicating the state of stateful applications. It does so by maintaining a replica of the state information on a different Managed Server, which is a cluster member. Oracle Fusion Middleware components, such as ADF and WebCenter, which are stateful, leverage this feature to ensure seamless failover to other members of the cluster.

  System components, such as Oracle Internet Directory, Oracle HTTP Server, Oracle Web Cache are stateless.

  Some Oracle Fusion Middleware components, which have part of the functionality implemented in C, such as Oracle Forms and Oracle Reports, are stateful and do not have state replication capabilities. Please refer to following paragraph for information about failover of these components.

- **Failover**

  Typically, a Managed Server running Oracle Fusion Middleware Java EE components has a Web server, such as Oracle HTTP Server, clustered in front of it. The Web server proxy plug-in (`mod_wl_ohs.conf`) is aware of the run time availability of different Managed Servers and the location of the Managed Server on which the state replica is maintained. If the primary Managed Server becomes unavailable, the plug-in routes the request to the server where the application is available. If stateful, applications such as Oracle ADF and WebCenter Portal, the location of the replica is also taken into account while routing to the new Managed Server.

For stateless system components, their multiple instances deploy as a runtime cluster behind a load balancer. The load balancer is configured to do a periodic health check of the component instances. If an instance is unavailable, the load balancer routes the subsequent requests to anther available instance and the failover is seamless.

For stateful components, which have parts based on C, and do not have state replication, sticky routing ensures that the subsequent requests go to the cluster member where the state was initially established. This is ensured by a Web server proxy plug-in and the Java EE parts of the components. If the component instance fails, subsequent requests route to another available member in the cluster. In this situation, the state information is lost and the user must recreate the session.

Some of the internal implementation of components use EJBs. EJB failover is seamlessly handled by replica aware WebLogic Server stubs.

Where needed, components are JTA compliant and data consistency is preserved in case of failover.

Singleton services leverage built-in failover capabilities, such as singleton SOA adapters, or use the underlying WebLogic Server infrastructure, such as Whole Server Migration.

- **Server Migration**

  Oracle Fusion Middleware components, such as SOA, which uses pinned services, such as JMS and JTA, leverage WebLogic Server capabilities to provide failover an automatic restart on a different cluster member.

- **Integrated High Availability**

  Oracle Fusion Middleware has a comprehensive feature set around load balancing and failover to leverage availability and scalability of Oracle RAC databases. All Oracle Fusion Middleware components have built-in protection against loss of service, data or transactions as a result of Oracle RAC instance unavailability due to planned or unplanned downtime. This is achieved by using WebLogic Server multi data sources. Additionally, components have proper exception handling and configurable retry logic for seamless failover of in-flight transactions at the time of failure.

  For XA compliant applications, such as Oracle SOA components, WebLogic server acts as a Transaction coordinator and ensures that all branches of a transaction are pinned to one of the Oracle RAC instances.

  In case of a Managed Server failure the transaction service is automatically migrated over to another node in the cluster and performs the transaction recovery.

  For communication between Web servers and application servers, the proxy plug-in has a built-in load balancing and failover capability to seamlessly reroute client requests to an available cluster member.

- **Rolling Patching**

Oracle WebLogic Server allows for rolling patching where a minor maintenance patch can be applied to the product binaries in a rolling fashion without having to shut down the entire cluster.

During the rolling patching of a cluster, each server in the cluster is individually patched and restarted while the other servers in the cluster continue to host your application. You can also uninstall a patch, maintenance pack, or minor release in a rolling fashion.

- **Configuration Management**

  Most of the Oracle Fusion Middleware component configuration can done at the cluster level. Oracle Fusion Middleware uses WebLogic Server's cluster wide-configuration capabilities for server configuration, such as data sources, EJBs, and JMS, as well as component application artifacts, and ADF and WebCenter custom applications.

- **Backup and Recovery**

  Oracle Fusion Middleware backup and recovery is a simple solution based on file system copy for Middle-tier components. RMAN is used for Oracle databases. There is also support for online backups. With Oracle Fusion Middleware, you can integrate with existing backup and recovery tools, or use scheduled backup tasks through oracle Fusion Middleware Enterprise Manager or cron jobs.

2.3.2.1 Server Load Balancing

Typically, Oracle Fusion Middleware high availability deployments are front ended by a load balancer which can be configured to distributed incoming requests using various algorithms.

Oracle Fusion Middleware also has built-in load balancing capabilities for intra component interaction. For example, Web server to application server, or application server to database server.

Oracle Fusion Middleware 11*g* does not provide external load balancers. To ensure that your external load balancer is compatible with Oracle Fusion Middleware, check that your external load balancer meets the requirements listed below:

- Virtual servers and port configuration: The load balancer should have the ability to configure virtual server names and ports on your external load balancer. The virtual server names and ports must meet the following requirements.

  - The load balancer should enable configuration of multiple virtual servers. For each virtual server, the load balancer should enable configuration of traffic management on more than one port. For example, for Oracle Fusion Middleware Identity Management, the load balancer needs to be configured with a virtual server and port for HTTP / HTTPS traffic, and separate virtual servers and ports for LDAP and LDAPS traffic.

  - The virtual server names must be associated with IP addresses and be part of your DNS. Clients must be able to access the external load balancer through the virtual server names.

- Persistence/stickiness: Some Oracle Fusion Middleware components use persistence or stickiness in an external load balancer. If your external load balancer does not allow you to set cookie persistence at the URI level, set the cookie persistence for all HTTP traffic. In either case, set the cookie to expire when the browser session expires. See your external load balancer documentation for details.

The recommended architecture for Oracle Fusion Middleware is a load balancer fronting Oracle HTTP Servers in the web tier, with Oracle WebLogic Server behind the Oracle HTTP Servers in the application tier.

If WebLogic Server is deployed directly behind a load balancer in the web tier, then review the information in the "Load Balancers and the WebLogic Session Cookie" in the *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*. Note that this is not a recommended deployment architecture for Oracle Fusion Middleware.

- Resource monitoring/port monitoring/process failure detection: Configure the external load balancer to detect service and node failures (through notification or some other means) and to stop directing traffic to the failed node. Your external load balancer may have the ability to automatically detect failures.

  For example, for Oracle Fusion Middleware Identity Management, the external load balancer should monitor Oracle Internet Directory, Oracle Fusion Middleware Single Sign-On, and Oracle Delegated Administration Services. To monitor these components, set up monitors for the following protocols:

  - LDAP and LDAPS listen ports
  - HTTP and HTTPS listen ports (depending on the deployment type)

  These monitors use the respective protocols to monitor the services, meaning they use LDAP for the LDAP port, LDAP over SSL for the LDAP SSL port, and HTTP/HTTPS for the Oracle HTTP Server port. If your external load balancer does not offer these monitors, consult your external load balancer documentation for the best method of configuring it to automatically stop routing incoming requests to a service that is unavailable.

- Network Address Translation (NAT): The load balancer should have the capability to perform network address translation (NAT) for traffic being routed from clients to the Oracle Fusion Middleware nodes.

- Port translation configuration: The load balancer should have the ability to perform port translation, where it enables incoming requests received on one port to be routed to a server process running on a different port. For example, a request received on port 80 can be routed to port 7777.

- Protocol translation: The load balancer should support protocol translation between systems running different protocols. It enables users on one network to access hosts on another network, despite differences in the native protocol stacks associated with the originating device and the targeted host. For example, incoming requests can be HTTPS, and outgoing requests can be HTTP.

  This feature is recommended but not required.

- SSL acceleration: SSL acceleration is a method of offloading the processor-intensive public key encryption algorithms involved in SSL transactions to a hardware accelerator.

  This feature is recommended but not required.

- Fault tolerant mode: Oracle highly recommends configuring the load balancer to be in fault-tolerant mode, otherwise the load balancer becomes a single point of failure for the system. This rules out most software load balancers that are based on a single process/interceptor as reliable solutions.

- Ability to preserve the client IP addresses: The load balancer must have the capability to insert the original client IP address of a request in an

X-Forwarded-For HTTP header or a similar feature to preserve the client IP address.

- Other: Oracle highly recommends configuring the load balancer virtual server to return immediately to the calling client when the back-end services to which it forwards traffic are unavailable. This configuration is preferred over the client disconnecting on its own after a timeout, based on the TCP/IP settings on the client system.

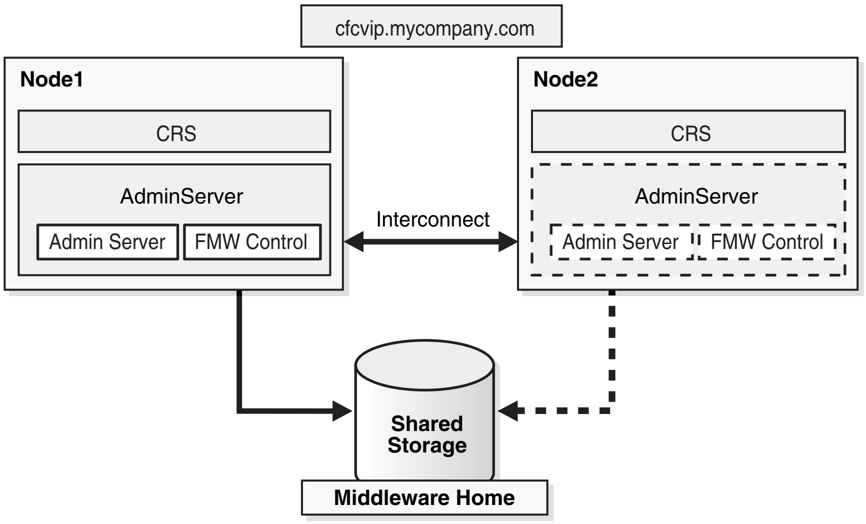
You may not need to meet all of the requirements in the previous listed. The requirements for external load balancers depend on the topology you are considering, and on the Oracle Fusion Middleware components you are load balancing.

2.3.3 Active-Passive Deployment

Oracle Fusion Middleware provides an active-passive model for all its components using Oracle Fusion Middleware Cold Failover Clusters. In an Oracle Fusion Middleware Cold Failover Cluster configuration, two or more application server instances are configured to serve the same application workload but only one is active at any particular time.

Figure 2–4 shows an example active-passive deployment.

Figure 2–4 Example Active-Passive Cold Failover Cluster Deployment



In Figure 2–4, the Administration Server runs on a two-node hardware cluster: Node 1 and Node 2. The Administration Server is listening on the Virtual IP or hostname. The Middleware Home and the domain directory is on a shared disk that is mounted on Node 1 or Node 2 at any given point. Both the Middleware home and the domain directory should be on the same shared disk or shared disks that can fail over together. If an enterprise has multiple Fusion Middleware domains for multiple applications or environments, this topology is well suited for Administration Server high availability. A single hardware cluster can be deployed to host these multiple Administration Servers. Each Administration Server can use its own virtual IP and set of shared disks to provide high availability of domain services.

For details about active-passive concepts, and configuration procedures for Oracle Cold Failover Clusters, see Chapter 15, "Active-Passive Topologies for Oracle Fusion Middleware High Availability."

2.3.4 About Active-Active and Active-Passive Solutions

Oracle recommends using active-active solutions when possible. This is the primary recommendation for maximum availability. Active-active solutions provide faster failover, scalability, and protection against node, instance and component failures. In addition, active-active solutions also offer transparent failover and the easy addition of resources for scaling up vertically and horizontally.

Scalability requirements are an important consideration when designing an Oracle Fusion Middleware high availability solution. Active-active solutions scale up (vertically) by adding more instances or components inside the same node.

Adding multiple redundant services in the same node improves the availability of a system, but only against instance failures and not against node failures. In addition, as described in Section 2.3.2, "Oracle Fusion Middleware High Availability Technologies," Oracle Fusion Middleware provides death detection and automatic restart of components. Active-active high availability solutions include multiple active instances installed on different nodes. As a result, when a node is completely lost other instances are available to keep the system going, uninterrupted. Using multiple instances in different nodes provides what is known as horizontal scalability, or scaling out.

Active-active solutions require logic to load balance and failover requests among the active Oracle Fusion Middleware instances. Load balancing is provided by distributing the incoming requests to different service providers. Failover is achieved by detecting any failures in this service providers and re-routing the request to other available service providers. This logic is implemented in different ways:

- Direct implementation: The logic is implemented directly by the client making a request to the system. For example, a JDBC client implements load balancing and failover logic to connect to multiple instances of an Oracle database (Real Application Cluster). It can be implemented by an external hardware load balancer.
- Third party implementation: The logic is provided by third party components that intercept the client requests and distribute the load to the multiple Oracle Instances. When several Oracle Instances are grouped to work together, they present themselves as a single virtual entry point to the system, which hides the multiple instance configuration. External load balancers can send requests to any application server instance in a cluster, as any instance can service any request.

Unlike the scalability properties of an active-active configuration, in active-passive configurations the passive component is used only when the active component fails. In active-active solutions all instances handle requests concurrently. As a result, active-active systems provide higher transparency and have greater scalability than an active-passive system.

Active-passive solutions are limited to vertical scalability, with just one node remaining active. Active-passive solutions also have an implicit failover time when failure occurs. This failover time is usually determined by the time it takes to restart the components in the node that becomes active post-failure. However, the operational and licensing costs of an active-passive model are lower than that of an active-active deployment.

There are situations where active-passive solutions are appropriate. Oracle recommends using hardware-cluster based active-passive solutions in the following scenarios:

- The licensing, management, and the total cost of ownership of a load balancer excludes an active-active solution, particularly if there is a hardware cluster available. Hardware clusters require two nodes, a switch for connecting the nodes, and shared storage that can be reached from both hardware nodes.

- You may have concurrency issues with Singleton services. With Singleton services, only one active instance can exit at runtime. Singleton services may be important in relation to other components. They typically provide basic services to multiple components, so if they are not available, then many other services or processes may not be available. Here are some issues to consider when protecting Singleton services:
  - Recovery Time: Singleton services or components can not run in active-active configurations. Client requests are not transparently load balanced to multiple instances of the service. This implies that, in case of a failure, there is an implicit recovery time. This recovery time varies depending on the type of Singleton protection model you adopt.
  - Reliability in failure detection: The system must prevent false positives. Most singleton services access data repositories that may or may not be designed for concurrent access. If a singleton service is reported as 'dead' and the system decides to start a new instance, a 'split brain' scenario could arise. The dead service must be analyzed for implications of this concurrency and how likely a false positive is to happen based on the failure detection mechanism.
  - Consistency in service across restarts: Singleton components must provide consistent service after a failover. These components must maintain the same behavior after recovering from a failure. The configuration and persistent repositories used by the service must be available during failures. Also, start dependencies must be accounted for upon failover. For example, if the singleton service needs to be restarted it may have start dependencies on other services and these must be preserved.
  - Cost (hardware/software resources required): Different protection mechanisms may require a pure software based solution, or a hardware based solution with the implicit costs.
  - Installation/Configuration/Management: The different protection mechanisms for singleton services should not add complexity to the system
  - Maintenance (patches, upgrades): Protection models for singleton services should enable easily and allow minimum downtime for applying patches and upgrades.

Based on these criteria, different solutions may be used for Oracle Fusion Middleware Singleton Components depending on the pertaining requirements to the specific singleton service:

- Cold Failover Cluster Solution: This solution requires a shared storage and a connection to detect hardware failures. The re-routing of requests by migration of the VHN through the failover procedure does not need intelligence on clients or load balancers.
- Whole Server Migration- This is the process of moving a clustered WebLogic Server instance or a component running on a clustered instance elsewhere if a failure occurs. In the case of whole server migration, the server instance is migrated to a different physical system upon failure. In the case of service-level migration, the services are moved to a different server instance within the cluster.
- Custom active-passive models based on software blocking mechanism: This logic is included in a component to prevent other instances of the same component from becoming active at the same time. Typical solutions use locks in a database, or custom in-memory active notifications that prevent concurrency.

In many cases, reliability of failure detection is an important factor for adopting one solution over another. This is especially true when concurrency can cause corruption of resources that are used by the singleton service. Typically, files may be written concurrently by different active instances.

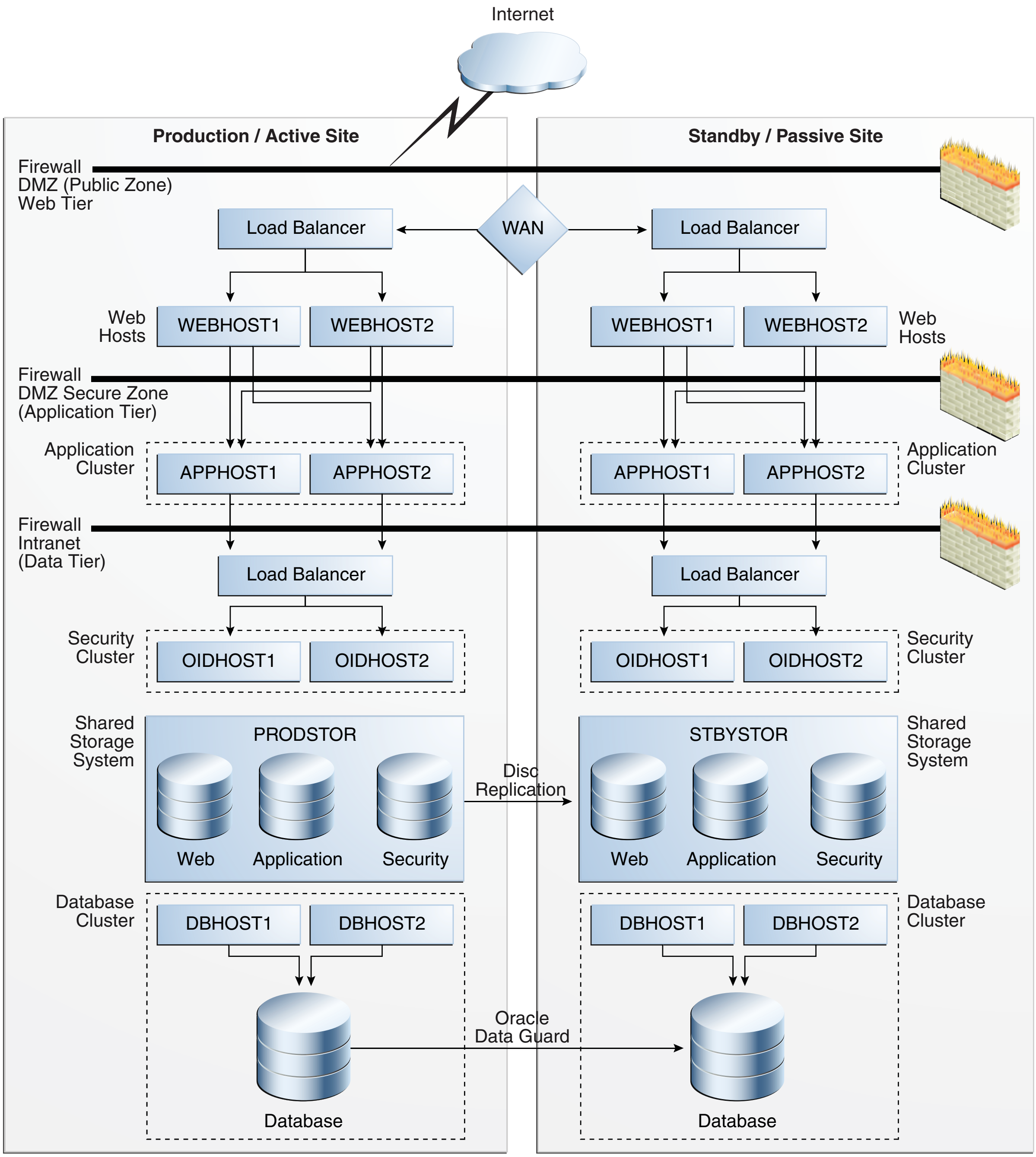
You may adopt other solutions for different components for the issues explained in this section.

2.3.5 Disaster Recovery

Figure 2–5 shows an Oracle Fusion Middleware architecture configured for Disaster Recovery. For Oracle Fusion Middleware product binaries, configuration files, and metadata files, the disk replication-based solution involves deploying Oracle Fusion Middleware on NAS/SAN devices. Product binaries and configuration data, stored in Oracle Homes, are stored on NAS/SAN devices using mounted locations from host systems. In addition, disk replication technologies are used to replicate product binaries and configuration from a production site shared storage system to a standby site shared storage system on a periodic basis. Standby site servers are also mounted to the disks on the standby site. If a failure or planned outage of the production (active) site occurs, replication to the standby (passive) site is stopped. The services and applications are subsequently started on the standby site. The network traffic is then be routed to the standby site.

For detailed information about disaster recovery for Oracle Fusion Middleware components, refer to *Oracle Fusion Middleware Disaster Recovery Guide*.

Figure 2–5 Production and Standby Site for Oracle Fusion Middleware Disaster Recovery Topology



2.4 Protection from Planned and Unplanned Down Time

The following tables list possible planned and unplanned downtime and suggested solutions for these downtime possibilities. Table 2–1 describes planned downtime:

Table 2–1 Planned Down Time Solutions

Operations	**Solutions**
Deploying and redeploying applications	Hot Deployment

Table 2–1 (Cont.) Planned Down Time Solutions

Operations	Solutions
Patching	Rolling Patching
Configuration Changes	Online configuration Changes Change Notification Batching of changes Deferred Activation
Scalability and Topology Extensions	Cluster Scale-Out

Table 2–2 describes unplanned downtime:

Table 2–2 Unplanned Down Time Solutions

Failures	Solutions
Software Failure	Death Detection and restart using Node Manager for Java EE and OPMN for system components. Server Clusters & Load Balancing Cold Failover Clusters Server Migration Service Migration State Replication and Replica aware Stubs
Hardware Failure	Server Clusters & Load Balancing Server Migration Clusterware Integration
Data Failure	Backup and Recovery
Human Error	
Site Disaster	Oracle Fusion Middleware Disaster Recovery Solution

Note: The architectures and deployment procedures defined in this guide enable simple clustered deployments. The procedures described in these chapters can be used as a building block to enable this and other similar high availability topologies for these Fusion Middleware components. It is also expected that production deployments will use other required procedures, such as associating security policies with a centralized LDAP server. For complete details of secured, multi-tiered architecture, and deployment procedures, please refer to the Enterprise Deployment Guide for the component you are configuring.

3

High Availability for WebLogic Server

This chapter describes the Oracle WebLogic Server high availability capabilities that provide Oracle Fusion Middleware high availability.

For complete documentation of WebLogic Server clustering, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.1 What Is a WebLogic Server Cluster?

A WebLogic Server cluster consists of multiple WebLogic Server server instances running simultaneously and working together to provide increased scalability and reliability. A cluster appears to clients to be a single WebLogic Server instance. The server instances that constitute a cluster can run on the same system, or be located on different systems. You can increase a cluster's capacity by adding additional server instances to the cluster on an existing system, or you can add systems to the cluster to host the incremental server instances. Each server instance in a cluster must run the same version of WebLogic Server.

3.2 WebLogic Server Clusters and WebLogic Server Domains

A cluster is part of a particular WebLogic Server domain. A *domain* is an interrelated set of WebLogic Server resources that are managed as a unit. A domain includes one or more WebLogic Server instances, which can be clustered, non-clustered, or a combination of clustered and non-clustered instances. A domain can include multiple clusters. A domain also contains the application components deployed in the domain and the resources and services required by those application components and the server instances in the domain. Examples of the resources and services used by applications and server instances include system definitions, optional network channels, connectors, and startup classes.

In each domain, one WebLogic Server instance acts as the Administration Server—the server instance which configures, manages, and monitors all other server instances and resources in the domain. Each Administration Server manages one domain only. If a domain contains multiple clusters, each cluster in the domain has the same Administration Server.

All server instances in a cluster must reside in the same domain; you cannot split a cluster over multiple domains. Similarly, you cannot share a configured resource or subsystem between domains. For example, if you create a JDBC connection pool in one domain, you cannot use it with a server instance or cluster in another domain. Instead, you must create a similar connection pool in the second domain.

Clustered WebLogic Server instances behave similarly to non-clustered instances, except that they provide failover and load balancing. The process and tools you use to configure clustered WebLogic Server instances are the same as those to configure non-clustered instances. However, to achieve the load balancing and failover benefits that clustering enables, you must adhere to certain guidelines for cluster configuration.

3.3 Benefits of Clustering

A WebLogic Server cluster provides these benefits:

- Scalability

  The capacity of an application deployed on a WebLogic Server cluster can be increased dynamically to meet demand. You can add server instances to a cluster without interruption of service—the application continues to run without affecting clients and end users.

- High Availability

  In a WebLogic Server cluster, application processing can continue when a server instance fails. You cluster application components by deploying them on multiple server instances in the cluster—so, if a server instance on which a component is running fails, another server instance on which that component is deployed can continue application processing.

The choice to cluster WebLogic Server instances is transparent to application developers and clients. However, understanding the technical infrastructure that enables clustering helps programmers and administrators maximize the scalability and availability of their applications.

3.4 Key Capabilities of a Cluster

This section defines key clustering capabilities that enable scalability and high availability. This section includes the following topics:

- Section 3.4.1, "Application Failover"
- Section 3.4.2, "Server Migration"
- Section 3.4.3, "Load Balancing"

3.4.1 Application Failover

Failover means that when an application component, typically referred to as an object in the following sections, doing a particular job—some set of processing tasks—becomes unavailable for any reason, a copy of the failed object finishes the job.

For the new object to be able to take over for the failed object, there must be:

- A copy of the failed object available to take over the job.
- Information, available to other objects and the program that manages failover, defining the location and operational status of all objects—so that it can be determined that the first object failed before finishing its job.
- Information, available to other objects and the program that manages failover, about the progress of jobs in process—so that an object taking over an interrupted job knows how much of the job was completed before the first object failed. For example: what data changed and what steps in the process were completed.

WebLogic Server uses standards-based communication techniques and facilities—including IP sockets and the Java Naming and Directory Interface (JNDI)—to share and maintain information about the availability of objects in a cluster. These techniques allow WebLogic Server to determine that an object stopped before finishing its job, and where there is a copy of the object to complete the interrupted job.

Information about what has been done on a job is called *state*. WebLogic Server maintains state information using techniques called *session replication* and *replica-aware stubs*. When an object unexpectedly stops doing its job, replication techniques enable a copy of the object to pick up where the failed object stopped and finish the job.

3.4.2 Server Migration

WebLogic Server supports automatic and manual migration of a clustered server instance from one system to another. The server migration process relocates a Managed Server in its entirety, including IP addresses and hosted applications, to one of a predefined set of available host systems. This feature is designed for environments with requirements for high availability. Server migration is useful for:

- Ensuring uninterrupted availability of *singleton services*—services that must run on only a single server instance at any given time, when the hosting server instance fails. A Managed Server configured for automatic migration automatically migrates to another system if a failure occurs.
- Facilitating the process of relocating a Managed Server and all services it hosts, as part of a planned system administration process. You initiate the Managed Server migration from the Administration Console or command line.

3.4.3 Load Balancing

Load balancing is the even distribution of jobs and associated communications across the computing and networking resources in your environment. Load balancing requires:

- Multiple copies of an object that can do a particular job.

- Information about the location and operational status of all objects.

  WebLogic Server allows objects to be clustered—deployed on multiple server instances—so that there are alternative objects to do the same job. WebLogic Server shares and maintains the availability and location of deployed objects using unicast, IP sockets, and JNDI.

3.5 Types of Objects That Can Be Clustered

A clustered application or application component is one that is available on multiple WebLogic Server instances in a cluster. If an object is clustered, failover and load balancing for that object is available. Deploy objects homogeneously—to every server instance in your cluster—to simplify cluster administration, maintenance, and troubleshooting.

Web applications can consist of different types of objects, including Enterprise Java Beans (EJBs), servlets, and Java Server Pages (JSPs). Each object type has a unique set of behaviors related to control, invocation, and how it functions within an application. For this reason, the methods that WebLogic Server uses to support clustering—and hence to provide load balancing and failover—can vary for different types of objects. In a WebLogic Server deployment, you can cluster these object types:

- Servlets
- JSPs
- EJBs
- Remote Method Invocation (RMI) objects
- Java Messaging Service (JMS) destinations

Different object types can have certain common behaviors. When this is the case, the clustering support and implementation considerations for similar object types may be same. The following sections combine explanations and instructions for these objects:

- Servlets and JSPs
- EJBs and RMI objects

3.6 Communications in a Cluster

WebLogic Server instances in a cluster communicate with one another using two basic network technologies:

- IP sockets, which are the conduits for peer-to-peer communication between clustered server instances.
- IP unicast or multicast, which server instances use to broadcast availability of services and heartbeats that indicate continued availability. When creating a new cluster, Oracle recommends that you use unicast for messaging within a cluster. For backward compatibility with previous versions of WebLogic Server, you must use multicast for communications between clusters.

Note: When using the unicast protocol for a WebLogic Server cluster, servers that are part of the cluster must specify a listen address. The servers cannot be listening on ANY, which is equivalent to leaving the **Listen Address** field in the Administration Console blank.

3.7 Cluster-Wide JNDI Naming Service

Clients of a non-clustered WebLogic Server server instance access objects and services by using a JNDI-compliant naming service. The JNDI naming service contains a list of the public services that the server instance offers, organized in a tree structure. A WebLogic Server instance offers a new service by binding into the JNDI tree a name that represents the service. Clients obtain the service by connecting to the server instance and looking up the bound name of the service.

Server instances in a cluster use a cluster-wide JNDI tree. A cluster-wide JNDI tree is similar to a single server instance JNDI tree, insofar as the tree contains a list of available services. In addition to storing the names of local services, however, the cluster-wide JNDI tree stores the services offered by clustered objects (EJBs and RMI classes) from other server instances in the cluster.

Each WebLogic Server instance in a cluster creates and maintains a local copy of the logical cluster-wide JNDI tree. Creation of a cluster-wide JNDI tree begins with the local JNDI tree bindings of each server instance. As a server instance boots (or as new services are dynamically deployed to a running server instance), the server instance first binds the implementations of those services to the local JNDI tree. The implementation is bound into the JNDI tree only if no other service of the same name exists.

Once the server instance successfully binds a service into the local JNDI tree, additional steps are performed for clustered objects that use replica-aware stubs. After binding the clustered object's implementation into the local JNDI tree, the server instance sends the object's stub to other members of the cluster. Other members of the cluster monitor the multicast or unicast address to detect when remote server instances offer new services.

3.8 Failover and Replication in a Cluster

For a cluster to provide high availability it must be able to recover from service failures. WebLogic Server instances in a cluster detect failures of their peer server instances by monitoring:

- Socket connections to a peer server

  WebLogic Server instances monitor the use of IP sockets between peer server instances as an immediate method of detecting failures. If a server connects to one of its peers in a cluster and begins transmitting data over a socket, an unexpected closure of that socket causes the peer server to be marked as "failed," and its associated services are removed from the JNDI naming tree.

- Regular server heartbeat messages

  If clustered server instances do not have opened sockets for peer-to-peer communication, failed servers may also be detected via the WebLogic Server heartbeat. All server instances in a cluster use multicast or unicast to broadcast regular server heartbeat messages to other members of the cluster.

  Each heartbeat message contains data that uniquely identifies the server that sends the message. Servers broadcast their heartbeat messages at regular intervals of 10 seconds. In turn, each server in a cluster monitors the multicast or unicast address to ensure that all peer servers' heartbeat messages are being sent.

  If a server monitoring the multicast or unicast address misses three heartbeats from a peer server (i.e., if it does not receive a heartbeat from the server for 30 seconds or longer), the monitoring server marks the peer server as "failed." It then updates its local JNDI tree, if necessary, to retract the services that were hosted on the failed server.

3.8.1 Session Replication

User session data can be stored in two standard ways in a Java EE application: stateful session EJBs or HTTP sessions. By themselves, they rarely impact cluster scalability. However, when coupled with a session replication mechanism required to provide high-availability, bottlenecks are introduced. If a Java EE application has Web and EJB components, you should store user session data in HTTP sessions:

- HTTP session management provides more options for handling fail-over, such as replication, a shared database or file.
- Superior scalability.
- Replication of the HTTP session state occurs outside of any transactions. Stateful session bean replication occurs in a transaction which is more resource intensive.
- The HTTP session replication mechanism is more sophisticated and provides optimizations a wider variety of situations than stateful session bean replication.

3.9 Whole Server Migration

In a WebLogic Server cluster, most services are deployed homogeneously on all server instances in the cluster, enabling transparent failover from one server to another. In contrast, pinned services such as JMS and the JTA transaction recovery system are targeted at individual server instances within a cluster—for these services, WebLogic Server supports failure recovery with migration as opposed to failover.

Migration in WebLogic Server is the process of moving a clustered WebLogic Server instance or a component running on a clustered instance elsewhere if failure occurs. In the case of whole server migration, the server instance is migrated to a different physical system upon failure. In the case of service-level migration, the services are moved to a different server instance within the cluster.

WebLogic Server provides a feature for making JMS and the JTA transaction system highly available: migratable servers. Migratable servers provide for both automatic and manual migration at the server-level, rather than the service level.

When a migratable server becomes unavailable for any reason, for example, if it hangs, loses network connectivity, or its host system fails—migration is automatic. Upon failure, a migratable server automatically restarts on the same system if possible. If the migratable server cannot restart on the system it failed on, it is migrated to another system. In addition, an administrator can manually initiate migration of a server instance.

3.9.1 Node Manager's Role in Whole Server Migration

Server migration requires Node Manager—it must run on each system that hosts, or is intended to host.

Node Manager supports server migration in these ways:

- You must use Node Manager for the initial startup of migratable servers.

  When you initiate the startup of a Managed Server from the Administration Console, the Administration Server uses Node Manager to start the server instance. You can also invoke Node Manager to start the server instance using the stand-alone Node Manager client; however, the Administration Server must be available so that the Managed Server can obtain its configuration.

> **Note:** Migration of a server instance that was not initially started with Node Manager will fail.

- You must use Node Manager to suspend, shutdown, or force shutdown of migratable servers.
- Node Manager tries to restart a migratable server with an expired lease on the system where it was running at the time of failure.

  Node Manager performs the steps in the server migrate process by running customizable shell scripts, provided with WebLogic Server, that start, restart and stop servers; migrate IP addresses; and mount and unmount disks. The scripts are available for Solaris and Linux.

  – In an automatic migration, the cluster master invokes Node Manager to perform the migration.

  – In a manual migration, the Administration Server invokes Node Manager to perform the migration.

3.9.2 Server Migration Processes and Communications

The following sections describe key processes in a cluster that contains migratable servers:

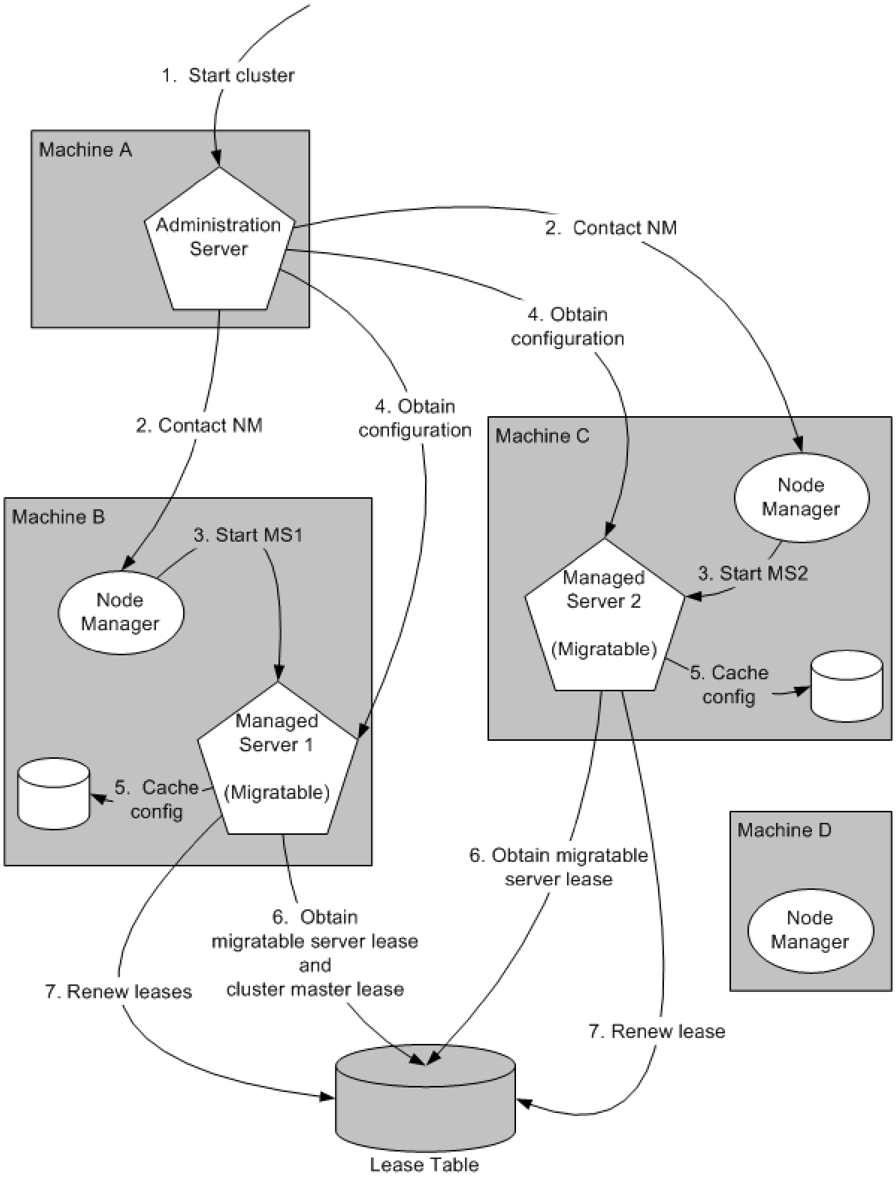
- Section 3.9.2.1, "Startup Process in a Cluster with Migratable Servers"
- Section 3.9.2.2, "Automatic Whole Server Migration Process"
- Section 3.9.2.3, "Manual Whole Server Migration Process"
- Section 3.9.2.4, "Administration Server's Role in Whole Server Migration"
- Section 3.9.2.5, "Migratable Server Behavior in a Cluster"
- Section 3.9.2.6, "Cluster Master's Role in Whole Server Migration"

3.9.2.1 Startup Process in a Cluster with Migratable Servers

Figure 3–1 shows the processing and communications that occur during startup of a cluster that contains migratable servers.

The example cluster contains two Managed Servers, both of which are migratable. The Administration Server and the two Managed Servers each run on different machines. A fourth machine is available as a backup—in the event that one of the migratable servers fails. Node Manager is running on the backup machine and on each machine with a running migratable server.

Figure 3–1 Startup of Cluster with Migratable Servers



These are the key steps that occur during startup of the cluster illustrated in Figure 3–1:

1. The administrator starts up the cluster.
2. The Administration Server invokes Node Manager on Machines B and C to start Managed Servers 1 and 2, respectively. See Section 3.9.2.4, "Administration Server's Role in Whole Server Migration."
3. The Node Manager on each machine starts up the Managed Server that runs there. See Section 3.9.1, "Node Manager's Role in Whole Server Migration."
4. Managed Servers 1 and 2 contact the Administration Server for their configuration. See Section 3.9.2.5, "Migratable Server Behavior in a Cluster."
5. Managed Servers 1 and 2 cache the configuration they started up.
6. Managed Servers 1 and 2 each obtain a migratable server lease in the lease table. Because Managed Server 1 starts up first, it also obtains a cluster master lease. See

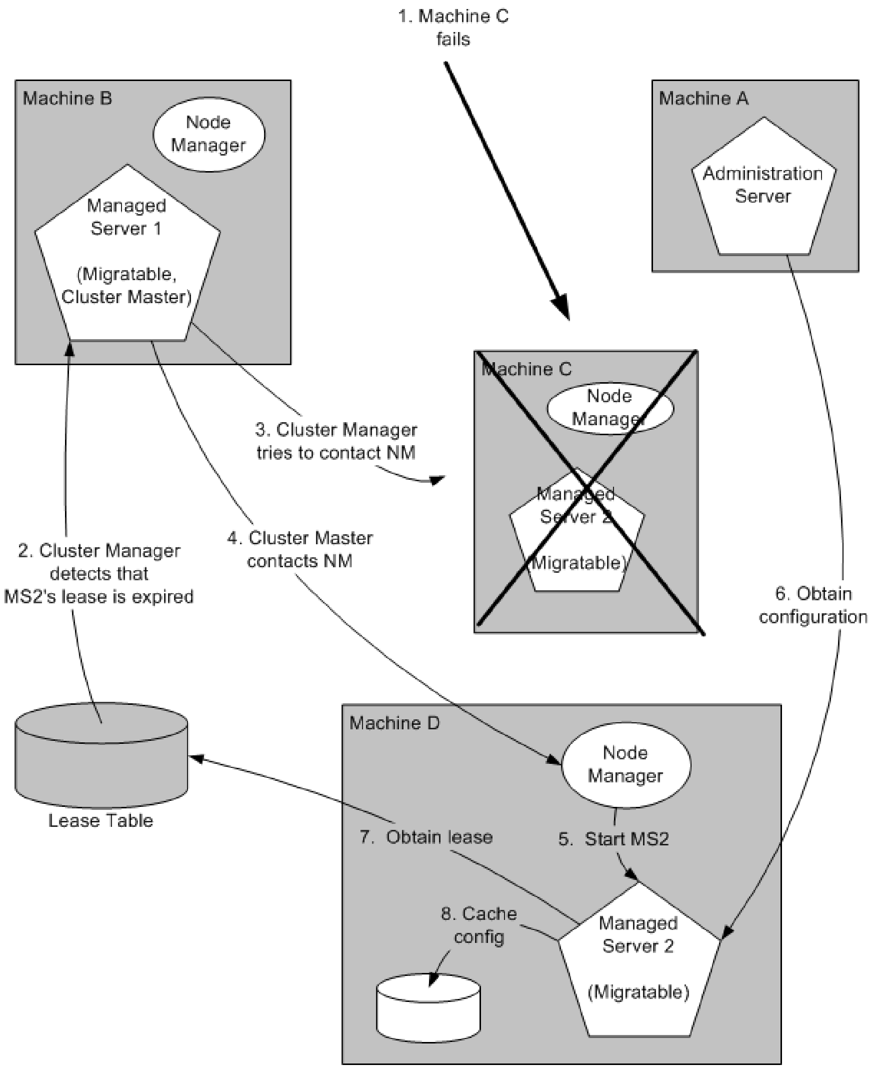
Section 3.9.2.6, "Cluster Master's Role in Whole Server Migration."

7. Managed Server 1 and 2 periodically renew their leases in the lease table, proving their health and liveness.

3.9.2.2 Automatic Whole Server Migration Process

Figure 3–2 shows the automatic migration process after the failure of the machine hosting Managed Server 2.

Figure 3–2 Automatic Migration of a Failed Server



1. Machine C, which hosts Managed Server 2, fails.

2. Upon its next periodic review of the lease table, the cluster master detects that Managed Server 2's lease has expired. See Section 3.9.2.6, "Cluster Master's Role in Whole Server Migration."

3. The cluster master tries to contact Node Manager on Machine C to restart Managed Server 2, but fails, because Machine C is unreachable.

> **Note:** If the Managed Server 2's lease had expired because it was hung, and Machine C was reachable, the cluster master would use Node Manager to restart Managed Server 2 on Machine C.

4. The cluster master contacts Node Manager on Machine D, which is configured as an available host for migratable servers in the cluster.
5. Node Manager on Machine D starts Managed Server 2. See Section 3.9.1, "Node Manager's Role in Whole Server Migration."
6. Managed Server 2 starts up and contacts the Administration Server to obtain its configuration.
7. Managed Server 2 caches the configuration it started up with.
8. Managed Server 2 obtains a migratable server lease.

During migration, the clients of the migrating Managed Server may have a brief interruption in service; it may be necessary to reconnect. On Solaris and Linux operating systems, you reconnect using `ifconfig` command. The clients of a migrated server do not need to know the particular machine the server migrates to.

When a machine that previously hosted a server instance that was migrated becomes available again, the reversal of the migration process—migrating the server instance back to its original host machine—is known as *failback*. WebLogic Server does not automate the failback process. You can accomplish failback by manually restoring the server instance to its original host.

The general procedures to restore a server to its original host are:

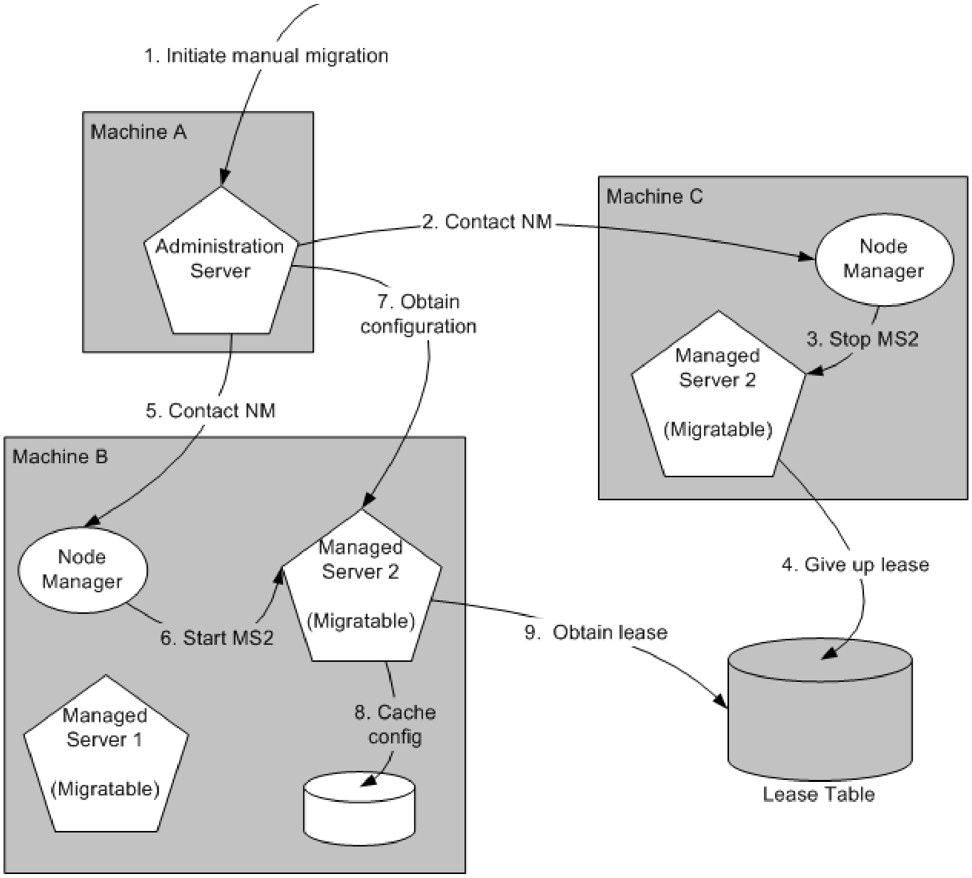
- Gracefully shutdown the new instance of the server.
- After you restart the failed machine, restart Node Manager and the managed server.

The exact procedures you follow depend on your server and network environment.

3.9.2.3 Manual Whole Server Migration Process

Figure 3–3 shows what happens when an administrator manually migrates a migratable server.

Figure 3–3 Manual Whole Server Migration



1. An administrator uses the Administration Console to initiate the migration of Managed Server 2 from Machine C to Machine B.
2. The Administration Server contacts Node Manager on Machine C. See Section 3.9.2.4, "Administration Server's Role in Whole Server Migration."
3. Node Manager on Machine C stops Managed Server 2.
4. Managed Server 2 removes its row from the lease table.
5. The Administration Server invokes Node Manager on Machine B.
6. Node Manager on Machine B starts Managed Server 2.
7. Managed Server 2 obtains its configuration from the Administration Server.
8. Managed Server 2 caches the configuration it started up with.
9. Managed Server 2 adds a row to the lease table.

3.9.2.4 Administration Server's Role in Whole Server Migration

In a cluster that contains migratable servers, the Administration Server invokes Node Manager on each system:

- That hosts cluster members to start the migratable servers. This is a prerequisite for server migratability—if Node Manager did not initially start a server instance, you cannot migrate the server.
- Involved in a manual migration process to stop and start the migratable server.

- That hosts cluster members to stop server instances during a normal shutdown. This is a prerequisite for server migratability—if a server instance is shut down directly without using Node Manager, when the cluster master detects that the server instance is not running, it will call Node Manager to restart it.

The Administration Server also provides its regular domain management functionality, persisting configuration updates issued by an administrator and providing a run-time view of the domain, including the migratable servers it contains.

3.9.2.5 Migratable Server Behavior in a Cluster

A migratable server is a clustered Managed Server that is configured as migratable. These are the key behaviors of a migratable server:

- If you are using a database to manage leasing information, during startup and restart by Node Manager, a migratable server adds a row to the lease table. The row for a migratable server contains a timestamp, and the machine where it is running.
- When using a database to manage leasing information, a migratable server adds a row to the database as a result of startup, it tries to take on the role of cluster master, and succeeds if it is the first server instance to join the cluster.
- Periodically, the server renews its "lease" by updating the timestamp in the lease table.

  By default a migratable server renews its lease every 30,000 milliseconds—the product of two configurable `ServerMBean` properties:

  - `HealthCheckIntervalMillis`, which by default is 10,000.
  - `HealthCheckPeriodsUntilFencing`, which by default is 3.
- If a migratable server fails to reach the lease table and renew its lease before the lease expires, it terminates as quickly as possible using a Java `System.exit`—in this case, the lease table still contains a row for that server instance. For information about how this relates to automatic migration, see Section 3.9.2.6, "Cluster Master's Role in Whole Server Migration."
- During operation, a migratable server listens for heartbeats from the cluster master. When it detects that the cluster master is not sending heartbeats, it attempts to take over the role of cluster master, and succeeds if no other server instance has claimed that role.

3.9.2.6 Cluster Master's Role in Whole Server Migration

In a cluster that contains migratable servers, one server instance acts as the cluster master. Its role is to orchestrate the server migration process. Any server instance in the cluster can serve as the cluster master. When you start a cluster that contains migratable servers, the first server to join the cluster becomes the cluster master and starts up the cluster manager service. If a cluster does not include at least one migratable server, it does not require a cluster master and the cluster master service does not start up. In the absence of a cluster master, migratable servers can continue to operate, but server migration is not possible. Key cluster master functions are:

- Issues periodic heartbeats to the other servers in the cluster.
- Periodically reads the lease table to verify that each migratable server has a current lease. An expired lease indicates to the cluster master that the migratable server should be restarted.

- Upon determining that a migratable server's lease is expired, waits for period specified by the `FencingGracePeriodMillis` on the `ClusterMBean`, and then tries to invoke the Node Manager process on the machine that hosts the migratable server whose lease is expired, to restart the migratable server.
- If unable to restart a migratable server whose lease has expired on its current machine, the cluster master selects a target machine in this fashion:
- If you configure a list of preferred destination machines for the migratable server, the cluster master chooses a machine on that list in the order the machines are listed. Otherwise, the cluster master chooses a machine on the list of those configured as available for hosting migratable servers in the cluster.

  A list of machines that can host migratable servers can be configured at two levels: for the cluster as a whole, and for an individual migratable server. You can define a machine list at both levels. You must define a machine list at least one level.
- To migrate a server instance to a new machine, the cluster master invokes the Node Manager process on the target machine to create a process for the server instance.

  The time required to perform the migration depends on the server configuration and startup time.
- The maximum time taken for cluster master to restart the migratable server is (`HealthCheckPeriodsUntilFencing` * `HealthCheckIntervalMillis`) + `FencingGracePeriodMillis`.
- The total time before the server becomes available for client requests depends on the server startup time and the application deployment time.

3.10 JMS and JTA High Availability

You can configure JMS and JTA services for high availability by using a migratable target, a special target that can migrate from one server in a cluster to another. A migratable target provides a way to group migratable services that should move together. When a migratable target migrates, all services the target hosts also migrate.

To configure a migratable JMS service for migration, it must be deployed to a migratable target. A migratable target specifies a set of servers that can host a target, and can optionally specify a user-preferred host for the services and an ordered list of candidate backup servers should the preferred server fail. Only one of these servers can host the migratable target at any one time.

After you configure a service to use a migratable target, the service is independent from the server member that is currently hosting it. For example, if a JMS server with a deployed JMS queue is configured to use a migratable target then the queue is independent of when a specific server member is available. That is, the queue is always available when the migratable target is hosted by any server in the cluster.

An administrator can manually migrate pinned migratable services from one server instance to another in the cluster in response to a server failure or as part of regularly scheduled maintenance. If you do not configure a migratable target in the cluster, migratable services can be migrated to any WebLogic Server instance in the cluster.

3.10.1 User-Preferred Servers and Candidate Servers

When deploying a JMS service to the migratable target, you can select a user-preferred server (UPS) target to host the service. When configuring a migratable target, you can also specify constrained candidate servers (CCS) that can host the service if the user-preferred server fails. If the migratable target does not specify a constrained candidate server, you can migrate the JMS server to any available server in the cluster.

WebLogic Server enables you to create separate migratable targets for JMS services. This allows you to always keep each service running on a different server in the cluster, if necessary. Conversely, you can configure the same selection of servers as the constrained candidate servers for both JTA and JMS, to ensure that the services remain co-located on the same server in the cluster.

3.10.2 Considerations for Using File Stores on NFS

If JMS messages and transaction logs are stored on an NFS-mounted directory, Oracle strongly recommends that you verify the behavior of a server restart after abrupt machine failures. Depending on the NFS implementation, different issues can arise post failover/restart.

To verify server restart behavior, abruptly shut down the node that hosts WebLogic servers while the servers are running.

- If you configured the server for server migration, it should start automatically in failover node after the failover period.
- If you did not configure the server for server migration, you can manually restart the WebLogic Server on the same host after the node completely reboots.

If WebLogic Server does not restart after abrupt machine failure, the following error entry may appear in server log files:

```
<MMM dd, yyyy hh:mm:ss a z> <Error> <Store> <BEA-280061> <The persistent
store "_WLS_server_soa1" could not be deployed:
weblogic.store.PersistentStoreException: java.io.IOException:
[Store:280021]There was an error while opening the file store file
"_WLS_SERVER_SOA1000000.DAT"
weblogic.store.PersistentStoreException: java.io.IOException:
[Store:280021]There was an error while opening the file store file
"_WLS_SERVER_SOA1000000.DAT"
        at weblogic.store.io.file.Heap.open(Heap.java:168)
        at weblogic.store.io.file.FileStoreIO.open(FileStoreIO.java:88)
...
java.io.IOException: Error from fcntl() for file locking, Resource
temporarily unavailable, errno=11
```

This error occurs when the NFSv3 system does not release locks on the file stores. WebLogic Server maintains locks on files that store JMS data and transaction logs to prevent data corruption that can occur if you accidentally start two instances of the same managed server. Because the NFSv3 storage device doesn't track lock owners, NFS holds the lock indefinitely if a lock owner fails. As a result, after abrupt machine failure followed by a restart, subsequent attempts by WebLogic Server to acquire locks may fail.

How you resolve this error depends on your NFS environment: (See *Oracle Fusion Middleware Release Notes* for updates on this topic.)

- **For NFSv4 environments**, you can set a tuning parameter on the NAS server to release locks within the approximate time required to complete server migration; you do not need to follow the procedures in this section. See your storage vendor's documentation for information on locking files stored in NFS-mounted directories on the storage device, and test the results.

- **For NFSv3 environments**, the following sections describe how to disable WebLogic file locking mechanisms for: the default file store, a custom file store, a JMS paging file store, a diagnostics file store.

> **WARNING: NFSv3 file locking prevents severe file corruptions that occur if more than one managed server writes to the same file store at any point in time.**
>
> **If you disable NFSv3 file locking, you must implement administrative procedures / policies to ensure that only one managed server writes to a specific file store. Corruption can occur with two managed servers in the same cluster or different clusters, on the same node or different nodes, or on the same domain or different domains.**
>
> **Your policies could include: never copy a domain, never force a unique naming scheme of WLS-configured objects (servers, stores), each domain must have its own storage directory, no two domains can have a store with the same name that references the same directory.**
>
> **If you configure a managed server using a file store for server migration, always configure the database- based leasing option. This option enforces additional locking mechanisms using database tables and prevents automated restart of more than one instance of a particular managed server.**

Disabling File Locking for the Default File Store

To disable file locking for the default file store using the Administration Console:

1. If necessary, click **Lock & Edit** in the Change Center (upper left corner) of the Administration Console to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Environment** node and select **Servers**.
3. In the **Summary of Servers** list, select the server you want to modify.
4. Select the **Configuration > Services** tab.
5. Scroll down to the **Default Store** section and click **Advanced**.
6. Scroll down and deselect the **Enable File Locking** check box.
7. Click **Save**. If necessary, click **Activate Changes** in the Change Center.
8. **Restart** the server you modified for the changes to take effect.

The resulting `config.xml` entry looks like the following:

```
<server>
  <name>examplesServer</name>
  ...
  <default-file-store>
    <synchronous-write-policy>Direct-Write</synchronous-write-policy>
    <io-buffer-size>-1</io-buffer-size>
    <max-file-size>1342177280</max-file-size>
    <block-size>-1</block-size>
    <initial-size>0</initial-size>
    <file-locking-enabled>false</file-locking-enabled>
  </default-file-store>
</server>
```

Disabling File Locking for a Custom File Store

To disable file locking for a custom file store using the Administration Console:

1. If necessary, click **Lock & Edit** in the Change Center (upper left corner) of the Administration Console to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Services** node and select **Persistent Stores**.
3. In the **Summary of Persistent Stores** list, select the custom file store you want to modify.
4. On the **Configuration** tab for the custom file store, click **Advanced**.
5. Scroll down and deselect the **Enable File Locking** check box.
6. Click **Save**. If necessary, click **Activate Changes** in the Change Center.
7. If the custom file store was in use, restart the server for the changes to take effect.

The resulting `config.xml` entry looks like the following:

```
<file-store>
  <name>CustomFileStore-0</name>
  <directory>C:\custom-file-store</directory>
  <synchronous-write-policy>Direct-Write</synchronous-write-policy>
  <io-buffer-size>-1</io-buffer-size>
  <max-file-size>1342177280</max-file-size>
  <block-size>-1</block-size>
  <initial-size>0</initial-size>
  <file-locking-enabled>false</file-locking-enabled>
  <target>examplesServer</target>
</file-store>
```

Disabling File Locking for a JMS Paging File Store

To disable file locking for a JMS paging file store using the Administration Console:

1. If necessary, click **Lock & Edit** in the Change Center (upper left corner) of the Administration Console to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Services** node, expand the **Messaging** node, and select **JMS Servers**.
3. In the **Summary of JMS Servers** list, select the JMS server you want to modify.
4. On the **Configuration > General** tab for the JMS Server, scroll down and deselect the **Paging File Locking Enabled** check box.
5. Click **Save**. If necessary, click **Activate Changes** in the Change Center.
6. **Restart** the server you modified for the changes to take effect.

The resulting `config.xml` file entry will look like the following:

```
<jms-server>
  <name>examplesJMSServer</name>
  <target>examplesServer</target>
  <persistent-store>exampleJDBCStore</persistent-store>
  ...
  <paging-file-locking-enabled>false</paging-file-locking-enabled>
  ...
</jms-server>
```

Disabling File Locking for a Diagnostics File Store

To disable file locking for a Diagnostics file store using the Administration Console:

1. If necessary, click **Lock & Edit** in the Change Center (upper left corner) of the Administration Console to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Diagnostics** node and select **Archives**.
3. In the **Summary of Diagnostic Archives** list, select the server name of the archive that you want to modify.
4. On the **Settings for [server_name]** page, deselect the **Diagnostic Store File Locking Enabled** check box.
5. Click **Save**. If necessary, click **Activate Changes** in the Change Center.
6. **Restart** the server you modified for the changes to take effect.

The resulting `config.xml` file will look like this:

```
  <server>
    <name>examplesServer</name>
    ...
    <server-diagnostic-config>
      <diagnostic-store-dir>data/store/diagnostics</diagnostic-store-dir>
      <diagnostic-store-file-locking-enabled>false</diagnostic-store-file-locking-
enabled>

<diagnostic-data-archive-type>FileStoreArchive</diagnostic-data-archive-type>
      <data-retirement-enabled>true</data-retirement-enabled>
      <preferred-store-size-limit>100</preferred-store-size-limit>
      <store-size-check-period>1</store-size-check-period>
    </server-diagnostic-config>
  </server>
```

3.11 Administration Server and Node Manager High Availability

The Administration Server is the WebLogic Server instance that configures and manages the WebLogic Server instances in its domain.

A domain can include multiple WebLogic Server clusters and non-clustered WebLogic Server instances. A domain can consist of only one WebLogic Server instance—however, in this case that sole server instance would be an Administration Server because each domain must have exactly one Administration Server.

There are a variety of ways to invoke the services of the Administration Server to accomplish configuration tasks. Whichever method is used, the Administration Server for a cluster must be running when you modify the configuration.

> **Note:** Oracle recommends that you set the Administration Server listen address to the hostname that its clients need to access it on, particularly for systems with multiple Middleware homes or Oracle homes.

3.11.1 Administration Server Failure

The failure of an Administration Server for a domain does not affect the operation of managed servers in the domain. If an Administration Server for a domain becomes unavailable while the server instances it manages—clustered or otherwise—are up and running, those managed servers continue to run. If the domain contains clustered server instances, the load balancing and failover capabilities supported by the domain configuration remain available, even if the Administration Server fails.

Note: If an Administration Server fails because of a hardware or software failure on its host machine, other server instances on the same machine may be similarly affected. However, the failure of an Administration Server itself does not interrupt the operation of managed servers in the domain.

For instructions on re-starting an Administration Server, see the *Oracle Fusion Middleware Using Clusters for Oracle Server*.

3.11.2 Node Manager Failure

If Node Manager fails or is explicitly shut down, upon restart, it determines the server instances that were under its control when it exited. Node Manager can restart any failed server instances as needed.

Note: It is advisable to run Node Manager as an operating system service, so that it restarts automatically if its host machine is restarted.

3.12 Load Balancing

Load balancing configuration consists of three pieces of information: the load-balancing algorithm to use, an indicator of whether local affinity should be applied, and weights that are assigned to each member of the topology to influence any routing algorithms that use weights.

The load-balancing algorithm specifies how requests are load balanced across components. Oracle Fusion Middleware uses three load-balancing methods:

- Round Robin - Requests are balanced across a list of available servers by selecting from the list sequentially.
- Random - Requests are balanced across a list of available servers by selecting a random server on each request.
- Weighted - Requests are balanced across a list of available servers using weights assigned to each server to determine the percentage of requests sent to each

Local affinity determines whether clients show a preference to servers that run on the same machine to avoid network latency. If the flag is set to true, then requests are routed across the list of servers on the local machine using the load-balancing algorithm if any local servers are available. If no local servers are available, requests are routed to all available remote servers according to the load-balancing algorithm. If local affinity is set to false, requests are routed across all available servers (local and remote) based on the load-balancing algorithm.

You configure weights as single integer values that are associated with component instances. You can assign weights to components that are not currently in a group, however, the weight is not used unless you later configure the component as a member of a group and select the weighted load-balancing algorithm. The weight is a unitless number. The percentage of requests to be sent to each member is calculated by summing the weights of all available members and dividing the weight for each member by the sum of the weights.

3.13 GridLink Data Sources

A single GridLink data source provides connectivity between WebLogic Server and an Oracle Database Real Application Clusters (RAC). It uses the Oracle Notification Service (ONS) to respond adaptively to state changes in an Oracle RAC. It responds to FAN events to provide Fast Connection Failover (FCF), Runtime Connection Load-Balancing (RCLB), and RAC instance graceful shutdown. It also provides capabilities of Affinities.

Oracle recommends GridLink data sources when you use an Oracle RAC database. For other databases, use multi data sources for high availability.

See Using GridLink Data Sources in the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide for more information about GridLink data sources.

3.14 Multi Data Sources

A multi data source is an abstraction around a group of data sources that provides load balancing or failover processing at the time of connection requests, between the data sources associated with the multi data source. Multi data sources are bound to the JNDI tree or local application context just like data sources are bound to the JNDI tree. Applications look up a multi data source on the JNDI tree or in the local application context (java:comp/env) just as they do for data sources, and then request a database connection. The multi data source determines which data source to use to satisfy the request depending on the algorithm selected in the multi data source configuration: load balancing or failover.

A multi data source can be thought of as a pool of data sources. Multi data sources are best used for failover or load balancing between nodes of a highly available database system, such as redundant databases or Oracle RAC.

3.15 Cluster Configuration and config.xml

The `config.xml` file is an XML document that describes the WebLogic Server domain configuration. The `domain` element in `config.xml` is the top-level element, and all elements in the domain descend from the `domain` element. The `domain` element includes child elements such as the `server`, `cluster`, and `application` elements. These child elements may have children of their own. For example, the `server` element can include the child elements WebServer, SSL and Log. The Application element includes the child elements EJBComponent and WebAppComponent.

Each element has one or more configurable attributes. An attribute defined in `config.dtd` has a corresponding attribute in the configuration API. For example, the Server element has a `ListenPort` attribute, and likewise, the `Weblogic.management.configuration.ServerMBean` has a `ListenPort` attribute. Configurable attributes are readable and writable, that is, `ServerMBean` has a `getListenPort()` and a `setListenPort()` method.

To learn more about `config.xml`, see the *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.16 About Singleton Services

A *singleton service* is a service that must run on only a single server instance at any given time, such as JMS and the JTA transaction recovery system, when the hosting server instance fails. A managed server configured for automatic migration automatically migrates to another machine if a failure occurs.

3.17 WebLogic Server and LDAP High Availability

In a high availability environment, WebLogic Server must be able to access LDAP for these reasons:

- To access users and groups stored in LDAP for which WebLogic Server supports failover.

  For information about configuring failover for LDAP authentication providers, see "Configuring Failover for LDAP Authentication Providers" in the *Oracle Fusion Middleware Securing Oracle WebLogic Server* manual.

- To access the LDAP-based policy store and credential store.

  For information about configuring a domain to use an LDAP-based policy store, see "Configuring a Domain to Use an LDAP-Based Policy Store" in the *Oracle Fusion Middleware Application Security Guide* manual.

  For information about configuring a domain to use an LDAP-based credential store, see "Configuring a Domain to Use an LDAP-Based Credential Store" in the *Oracle Fusion Middleware Application Security Guide* manual.

4

Considerations for High Availability Oracle Database Access

This chapter describes considerations for high availability Oracle database access.

The sections in this chapter are as follows:

- Section 4.1, "Oracle Real Application Clusters and Fusion Middleware"
- Section 4.2, "Protecting Idle Connections from Firewall Timeouts"
- Section 4.3, "Troubleshooting"
- Section 4.4, "Using SCAN Addresses with Oracle Database 11g (11.2)"

4.1 Oracle Real Application Clusters and Fusion Middleware

Most Fusion Middleware components use a database as the persistent store for their data. You can configure the Oracle database back end in any number of high availability configurations, including Cold Failover Clusters, Oracle Real Application Clusters, Oracle Data Guard, or Oracle Streams. For more information, see the *Oracle Database High Availability Overview*. This chapter describes considerations for Oracle Fusion Middleware configured with a high availability Oracle database, Oracle Real Application Clusters.

Oracle Real Application Cluster (Oracle RAC) is a computing environment that harnesses the processing power of multiple, interconnected computers. Along with a collection of hardware (cluster), it unites the processing power of each component to become one robust computing environment. Oracle RAC simultaneously provides a highly scalable and highly available database for Oracle Fusion Middleware.

Every Oracle RAC instance in the cluster has equal access and authority, therefore, node and instance failure may affect performance, but doesn't result in downtime; the database service is available or can be made available on surviving server instances.

For more information on Oracle RAC, see the *Oracle Real Application Clusters Administration and Deployment Guide*.

Oracle Fusion Middleware provides the best integration with an Oracle database in a high availability environment. When Oracle Fusion Middleware behaves as a client for the database (either as a java or system client) it uses special communication and monitoring capabilities that provide fast failover and minimal middle tier disruption in reaction to database failure scenarios.

You can categorize Oracle Fusion Middleware components that access the database as follows:

- Java-based components deployed to Oracle WebLogic Server

- Java-based components that are standalone Java Clients
- Non-Java components

This chapter contains the following sections:

- Section 4.1.1, "Java-Based Oracle Fusion Middleware Components Deployed to Oracle WebLogic Server"
- Section 4.1.2, "GridLink Data Sources and Oracle RAC"
- Section 4.1.3, "Using Multi Data Sources with Oracle RAC"
- Section 4.1.4, "Configuring GridLink Data Sources with Oracle RAC"
- Section 4.1.5, "Configuring Multi Data Sources with Oracle RAC"
- Section 4.1.6, "JDBC Clients"

4.1.1 Java-Based Oracle Fusion Middleware Components Deployed to Oracle WebLogic Server

All Oracle Fusion Middleware components deployed to Oracle WebLogic Server support Oracle RAC. For establishing connection pools, Oracle Fusion Middleware supports GridLink data sources and multi data sources for the Oracle RAC back end for both XA and non-XA JDBC drivers. Oracle Fusion Middleware deployments do not support other connection failover features supported by Oracle JDBC drivers for Oracle RAC. See component specific guides for multi data source configuration details.

When an Oracle RAC node or instance fails, WebLogic Server or the Oracle Thin driver redirect session requests to another node in the cluster. There is no failover of existing connections, however, new connection requests from the application are managed using existing connections in the Oracle WebLogic pool or by new connections to the working Oracle RAC instance. When the database is the transaction manager, in-flight transactions typically roll back. When the WebLogic Server is the transaction manager, in-flight transactions fail over; they are driven to completion or rolled back based on the transaction state when failure occurs. If the application requires load balancing across Oracle RAC nodes, WebLogic Server supports this capability by using JDBC GridLink data sources or JDBC multi data sources configured for load balancing. See Run Time Connection Load Balancing in the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide for more information on GridLink load balancing.

4.1.2 GridLink Data Sources and Oracle RAC

A GridLink data source includes the features of generic data sources plus the following support for Oracle RAC:

- Fast Connection Failover
- Runtime Connection Load Balancing
- Graceful Handling for Oracle RAC Outages
- XA Affinity
- Single Client Access Name (SCAN) Addresses
- Secure Communication Using Oracle Wallet

Oracle provides GridLink data sources and multi data sources to support high availability, load balancing, and failover of database connections. Oracle recommends the following data source types depending on the Oracle RAC Database version you have:

- If you use Oracle RAC database version 11g Release 2 and later, use GridLink data sources.
- If you use an Oracle RAC database version earlier than 11g Release 2 or a non-Oracle database, use multi data sources.

> **Note:** Oracle recommends using the GridLink data sources with Oracle RAC database for maximum availability. For versions of Oracle RAC databases where GridLink data sources are not supported, Oracle recommends using multi data sources for high availability.

See Using GridLink Data Sources in the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide for more information on the following topics:

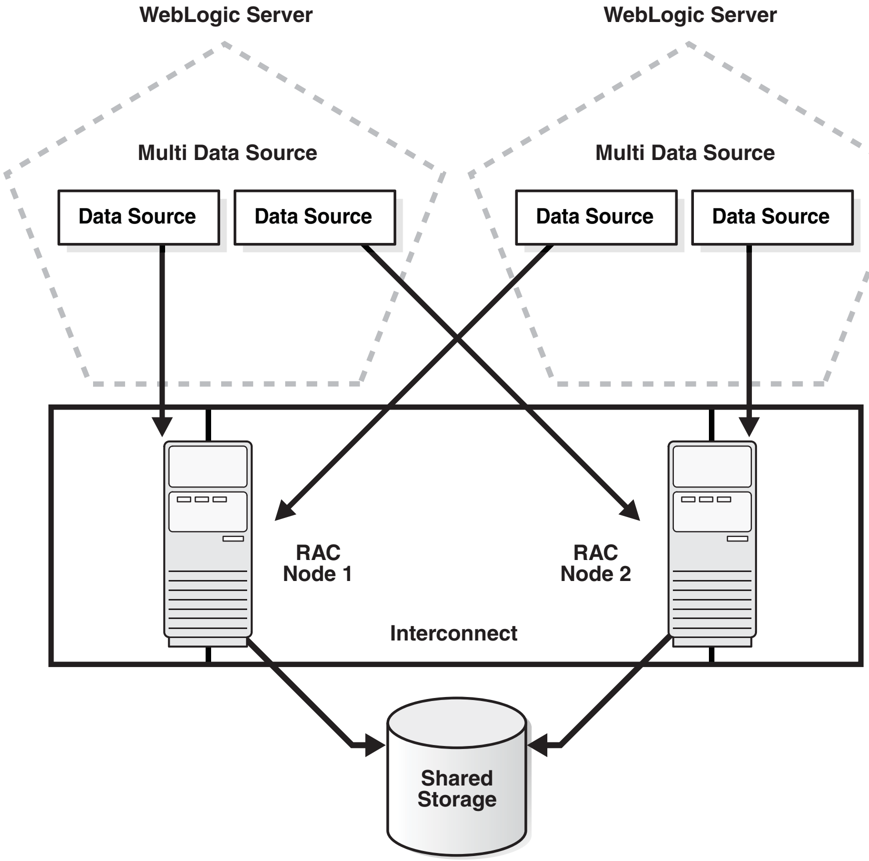
- What is a GridLink Data Source?
- Using Socket Direct Protocol for a GridLink Data Source
- Configuring Connection Pool Features
- Configuring Oracle Parameters
- Configuring an ONS Client
- Tuning GridLink Data Source Connection Pools
- Setting Database Security Credentials
- Monitoring GridLink JDBC Resources

4.1.3 Using Multi Data Sources with Oracle RAC

When you deploy Oracle Fusion Middleware against an Oracle RAC back end it is configured out of the box with multi data sources. The multi data sources have constituent data sources for each RAC instance providing the database service. Oracle recommends that you add an additional data source to the multi data source on the Fusion Middleware tier when you configure additional RAC instances that offer the database service. Ensure that each constituent data source that you create for the multi data source are configured identically for properties in Section 4.1.5, "Configuring Multi Data Sources with Oracle RAC."

When you migrate the database from a non-RAC to a RAC database, you must create an equivalent, new multi data source for each data source that is affected. The multi data source that you create must have consistent data sources for each RAC instance. The data source values must be identical to the original single instance data source for the properties in Section 4.1.5, "Configuring Multi Data Sources with Oracle RAC." For example, if the single instance data source driver is `oracle.jdbc.xa.client.OracleXADataSource`, it must be `oracle.jdbc.xa.client.OracleXADataSource` for each constituent data source of the new multi data source.

Figure 4–1 Multi Data Source Configuration



4.1.3.1 Configuring Multi Data Sources for MDS Repositories

Applications that use an MDS database-based repository can be configured for high availability Oracle database access. With this configuration, failure detection, recovery, and retry by MDS, as well as by the WebLogic infrastructure, result in the application's read-only MDS operations being protected from Oracle RAC database planned and unplanned downtimes.

Multi data sources are exposed as MDS repositories in the Fusion Middleware Control navigation tree. These multi data sources can be selected during deployment plan customization of application deployment, and can be used with MDS WLST commands.

- Configuring an application to retry read-only operations

  To configure an application to retry the connection, you can configure the RetryConnection attribute of the application's MDS AppConfig MBean. See the *Oracle Fusion Middleware Administrator's Guide* for more information.

- Registering an MDS multi data source

  In addition to the steps specified in Section 4.1.5, "Configuring Multi Data Sources with Oracle RAC," consider the following:

  – The child data sources that constitute a multi data source used for an MDS repository must be configured as non-XA data sources.

  – The multi data source's name must be pre-fixed with `mds-`. This ensures that the multi data source can be recognized as an MDS repository that can be used for MDS management functionality through Fusion Middleware Control, WLST, and JDeveloper.

Note: When an MDS data source is added as a child of a multi data source, this data source is no longer exposed as an MDS repository. For example, it does not appear under the Metadata Repositories folder in the Fusion Middleware Control navigation tree, you cannot perform MDS repository operations on it, and it does not appear in the list of selectable repositories during deployment.

- Converting a data source to a multi data source

  There are two considerations when converting an data source to a multi data source to make sure the application is configured correctly:

  – To create a new multi data source with a new, unique name, redeploy the application and select this new multi data source as the MDS repository during deployment plan customization.

  – To avoid redeploying the application, you can delete the data source and recreate the new multi data source using the same name and jndi-name attributes.

4.1.3.2 Oracle RAC Configuration Requirements

This section describes requirements for Oracle RAC configuration:

- **XA Requirements**: Many Oracle components participate in distributed transactions, or are part of container managed transactions. These components require the back-end database setup for XA recovery by Oracle WebLogic Transaction Manager. For repositories created using RCU, this is done automatically. For other databases participating in XA transactions, ensure that XA pre-requisites are met:

  1. Log on to SQL*Plus as a system user, for example:

     ```
     sqlplus "/ as sysdba"
     ```

  2. Grant select on `sys.dba_pending_transactions` to `public`.
  3. Grant run on `sys.dbms_xa` to `public`.
  4. Grant force any transaction to `user`.

Note: Ensure that the `distributed_lock_timeout` parameter for the Oracle database is set to a value higher that the JTA timeout. It should be higher than the highest value on the middle tier - between the default for WebLogic Server, a specific configuration for a data source, or one used by a component for a transaction.)

- **Server-side Load Balancing**: If the server-side load balancing feature is enabled for the Oracle RAC back end (using remote_listeners), the JDBC URL used in the data sources of a multi data source configuration should include the INSTANCE_NAME. For example, you can specify the URL in the following format:

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))
```

By default, the out-of-box installation assumes that `remote_listener` has been configured and creates the URL for data sources in a multi data source accordingly. Any multi data source created outside of the typical installation and configuration should follow the format described in this section.

If `remote_listeners` cannot be specified on the Oracle RAC side, and server side load balancing has been disabled, specifying the INSTANCE_NAME in the URL is not necessary. To disable remote listeners, delete any listed remote listeners in `spfile.ora` file on each Oracle RAC node. For example:

```
*.remote_listener="
```

In this case, the recommended URL that you use in the data sources of a multi data source configuration is:

```
jdbc:oracle:thin:@host-vip:port/dbservice
```

Or

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)(PORT=1521
))(CONNECT_DATA=(SERVICE_NAME=dbservice)))
```

- **Services**: When configuring Oracle Fusion Middleware for the Oracle database and specifically for Oracle RAC, Oracle recommends using the Oracle Services feature. Create the `service_name` provided as part of the database service location specifically for the application.

4.1.3.3 Configuring Schemas for Transactional Recovery Privileges

You need the appropriate database privileges to enable the Oracle WebLogic Server transaction manager to query for transaction state information and issue the appropriate commands, such as commit and rollback, during recovery of in-flight transactions after a WebLogic Server container failure.

To configure the schemas for transactional recovery privileges:

1. Log on to SQL*Plus as a user with sysdba privileges. For example:

   ```
   sqlplus "/ as sysdba"
   ```

2. Grant select on sys.dba_pending_transactions to the appropriate_user.
3. Grant force any transaction to the appropriate_user.

> **Note:** Grant these privileges to the soainfra schema owner, as determined by the RCU operations.

4.1.4 Configuring GridLink Data Sources with Oracle RAC

How you configure a GridLink data source depends on the Oracle component that you are working with and the domain you are creating. For detailed procedures, go to the section that describes the component type you are working with.

Note: Oracle recommends that you use Oracle Single Client Access Name (SCAN) addresses to specify the host and port for both the TNS listener and the ONS listener in the WebLogic console. You do not need to update a GridLink data source containing SCAN addresses if you add or remove Oracle RAC nodes. Contact your network administrator for appropriately configured SCAN URLs for your environment. See SCAN Addresses in the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide.

For a generic overview of how to configure a GridLink data source, see Creating a GridLink Data Source in the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide.

4.1.5 Configuring Multi Data Sources with Oracle RAC

You can configure multi data sources using the following:

- Oracle Fusion Middleware Configuration Wizard during WebLogic Server domain creation
- Oracle Universal Installer Java EE component configuration for Identity Management or Oracle Portal, Forms, Reports, and Discoverer.
- Oracle WebLogic Server Administration Console
- WLST Commands

Multi data sources support load balancing for both XA and non-XA data sources, including all Oracle database versions that Oracle Fusion Middleware components support.

Multi data sources encapsulate individual data sources that pool connections to specific instances of Oracle RAC. For multi data sources created manually, or modified after initial configuration, Oracle strongly recommends the following XA and Non-XA data source property values for optimal high availability behavior. Make changes only after careful consideration and testing if your environment requires that you do so:

Table 4–1 Recommended Multi Data Source Configuration

Property Name	Value
test-frequency-seconds	5
algorithm-type	Load-Balancing

For individual data sources, Oracle recommends the following for high availability environments. Oracle recommends that you set any other parameters according to application requirements.

Table 4–2 XA Data Source Configuration

Property Name	Value
Driver	oracle.jdbc.xa.client.OracleXADataSource
Property command	<property> <name>oracle.net.CONNECT_TIMEOUT</name> <value>10000</value> </property>
initial-capacity	0
connection-creation-retry-frequency-seconds	10
test-frequency-seconds	300
test-connections-on-reserve	true
test-table-name	SQL SELECT 1 FROM DUAL
seconds-to-trust-an-idle-pool-connection	0
global-transactions-protocol	TwoPhaseCommit
keep-xa-conn-till-tx-complete	true
xa-retry-duration-seconds	300
xa-retry-interval-seconds	60

Table 4–3 Non-XA Data Source Configuration

Property Name	Value
Driver	oracle.jdbc.OracleDriver
Property to set	<property> <name>oracle.net.CONNECT_TIMEOUT</name> <value>10000</value> </property>
initial-capacity	0
connection-creation-retry-frequency-seconds	10
test-frequency-seconds	300
test-connections-on-reserve	true
test-table-name	SQL SELECT 1 FROM DUAL
seconds-to-trust-an-idle-pool-connection	0
global-transactions-protocol	None

For examples of recommended multi data sources, see Appendix B, "Recommended Multi Data Sources."

Increasing Transaction Timeout for XA Data Sources

If you see WARNING messages in the server logs that include the following exception:

javax.transaction.SystemException: Timeout during commit processing

```
[ javax.transaction.SystemException: Timeout during commit processing
```

This message may indicate the XA timeout value you have in your setup must be increased. You can increase XA timeout for individual data sources when these warnings appear.

To increase this setting, use Administration Console:

1. Access the data source configuration.
2. Select the **Transaction** tab.
3. Set XA Transaction Timeout to a larger value, for example, **300**.
4. Select the **Set XA Transaction Timeout** checkbox. You *must* select this checkbox for the new XA transaction timeout value to take effect.
5. Click **Save**.

Repeat this configuration for all individual data sources of an XA multi data source.

4.1.6 JDBC Clients

Java J2SE-based Oracle Fusion Middleware components are optimized to work with the high availability features of Oracle RAC. You can deploy the components to use both the Oracle thin JDBC driver or the OCI based JDBC drivers.

The JDBC Thin client is a pure Java, Type IV driver. It is lightweight, easy to install and provides high performance, comparable to the performance of the JDBC Oracle Call Interface (OCI) driver. The JDBC Thin driver communicates with the server using TTC, a protocol developed by Oracle to access data from Oracle database. The driver enables a direct connection to the database by providing an implementation of TCP/IP that implements Oracle Net and TTC on top of Java sockets. The JDBC OCI client is a Type II driver and provides connections to JDBC clients over the Oracle Net. It uses the client side installation of Oracle Net and a deployment can customize behavior using Oracle Net configuration on the middle tier.

> **Note:** These JDBC clients are used as part of standalone Java J2SE programs.

Oracle Virtual Directory

When used with database adapters, Oracle Virtual Directory connects to a database, and the connections are not pooled. For details about configuring database adapters for Oracle RAC, see "Creating Database Adapters" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

Database URL

To configure an Oracle Virtual Directory database adapter for an Oracle RAC database using the Oracle Directory Services Manager:

1. In the **Connection** screen, select **Use Custom URL** from the **URL Type** list.
2. In the **Database URL** field, enter the URL to connect to the Oracle RAC database, for example:

   JDBC Thin

   ```
   jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(LOAD_
   BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=host-name-1)(PORT=1521))(ADDRESS=
   (PROTOCOL=TCP)(HOST=host-name-2)(PORT=1521)))(CONNECT_
   DATA=(SERVER=DEDICATED)(SERVICE_NAME=database-service-name)))
   ```

JDBC OCI

```
jdbc:oracle:oci:@(DESCRIPTION=(ADDRESS_LIST=(LOAD_
BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=host-name-1)(PORT=1521))(ADDRESS=
(PROTOCOL=TCP)(HOST=host-name-2)(PORT=1521)))(CONNECT_
DATA=(SERVER=DEDICATED)(SERVICE_NAME=database-service-name)))
```

Connection Timeout Configuration

To configure the connection timeout for an Oracle RAC database using the Oracle Directory Services Manager:

1. In the **Connection** screen, for **JDBC Thin**, specify the database adapter parameter **oracleNetConnectTimeout** for the timeout parameter in seconds.
2. For JDBC OCI, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora in ORACLE_INSTANCE/config` directory.

4.1.7 System Clients

Oracle Fusion Middleware 11g includes some non-Java components. These components are primarily C-based and include Oracle Internet Directory, Oracle Forms, Oracle Reports, Oracle Discoverer, and Oracle Portal. These components use the Oracle Call Interface layer to interact with Oracle databases. For Oracle RAC-based systems, some components integrate with the Oracle high availability Event Notification database feature.

High availability Event Notification provides a signal to the non-Java application if database failure occurs. The applications can register a callback on the environment to monitor the database connection. When a database failure related to the non-Java client occurs, the callback is invoked. This callback contains information about the database failure, including the event payload, and a list of connections (server handles) that were disconnected as a result of the failure.

If another instance, for example, instance C, of the same database, goes down, the client is not notified, since it does not affect any of the client's connections.

High availability Event Notification improves the application response time if database failure occurs. Without Event Notification, database failure would result in the connection being broken only after the TCP time out expired, which could take minutes. With high availability Event Notification, OCI automatically breaks and cleans up standalone, connection pool, and session pool connections and the application callback is invoked within seconds of failure. If any server handles are TAF-enabled, OCI automatically engages failover.

The following section describes the recommended setting for non-Java client connections to Oracle RAC databases.

4.2 Protecting Idle Connections from Firewall Timeouts

Because most production deployments involve firewalls and database connections are made across firewalls, Oracle recommends configuring the firewall not to timeout the database connection. For Oracle RAC case, this specifically means not timing out the connections made on Oracle RAC VIPs and the database listener port.

If such a configuration is not possible, on the database server side, set `SQLNET.EXPIRE_TIME=n` in `ORACLE_HOME/network/admin/sqlnet.ora`. For Oracle RAC, this needs to be set on all the Oracle Homes. The `n` is in minutes. It should be set to less than the known value of the network device (firewall) timeout. Since the order of these times is normally more than ten minutes, and in some cases hours, the value should be set to the highest possible value.

4.3 Troubleshooting

If an Oracle RAC instance goes down, WebLogic Server determines the database status using a SELECT 1 FROM DUAL query. This query typically takes less than a few seconds to complete. However, if the database response is slow, WebLogic Server gives up and assumes the database is unavailable. The following is an example of the type of exception that results in the logs:

```
<Mar 30, 2009 2:14:37 PM CDT> <Error> <JDBC> <BEA-001112> <Test "SELECT 1 FROM
 DUAL" set up for pool SOADataSource-rac1" failed with exception:
 oracle.jdbc.xa.OracleXAException".> [TopLink Warning]: 2009.03.30
 14:14:37.890--UnitOfWork(14568040)--Exception [TOPLINK-4002] (Oracle TopLink -
 11g Release 1 (11.1.1.1.0) (Build 090304)):
 oracle.toplink.exceptions.DatabaseException Internal Exception:
 java.sql.SQLException: Internal error: Cannot obtain XAConnection Creation of
 XAConnection for pool SOADataSource failed after waitSecs:30 :
 weblogic.common.ResourceException: SOADataSource(SOADataSource-rac1): Pool
 SOADataSource-rac1 has been @ disabled because of hanging connection tests,
 cannot allocate resources to applications. We waited 10938 milliseconds. A
 typical test has been taking 16.
```

You can set the WebLogic Server parameter, `-Dweblogic.resourcepool.max_test_wait_secs=30` to increase the time WebLogic Server waits for a response from the database. This parameter is located in the setDomainEnv.sh file. By setting this parameter, WebLogic Server waits 30 seconds for the database to respond to the SELECT 1 FROM DUAL query before giving up.

4.4 Using SCAN Addresses with Oracle Database 11g (11.2)

If your 11.2 RDBMS Oracle RAC database is not configured with Single Client Access Name (SCAN), you can provide details of the Oracle RAC instances in the Configuration Wizard and Oracle Universal Installer, just as you entered them for previous database releases. The instance address is in the `host:port` format.

If your 11.2 RDBMS Oracle RAC database is configured with SCAN, provide Oracle RAC instance details with the SCAN address. In Fusion Middleware wiring to an Oracle RAC instance, each Oracle RAC instance is uniquely identified using the service name, instance name, host, and port. Because the `host:port` address of all such instances is the SCAN `host:port`, Oracle recommends that you use this same common address for all instances.

Follow these guidelines based on the installation type:

- **In RCU installations** against an Oracle RAC database, specify the hostname as `scan-hostname-address`.
- **In Oracle Universal Installer installations**, specify the following for Oracle RAC databases depending on the input format Oracle Universal Installer requires.

  ```
  scan-address-hostname:port:instance1^scan-address-hostname:port:instance2@servicename
  ```

  or:

  ```
  scan-address-hostname:port^scan-address-hostname:port@servicename
  ```

- **In Configuration Wizard installations that use a multi data source**, the `scan-address-hostname,port,service-name` must be the same for each constituent data source. The instance names must be specific for each constituent data source and targeted to the Oracle RAC end instances.

  GridLink with SCAN does not use an instance name. The following example shows a GridLink connection string that does not use the RAC instance:

  ```
  jdbc:oracle:thin:@(DESCRIPTION=(ENABLE=BROKEN)(ADDRESS_
  LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=<scan-host-name>)(PORT=<scan-port>)))(CONNECT
  _DATA=(SERVICE_NAME=rac-service-name)))
  ```

- If the connect string is specified explicitly, use the following base format:

  ```
  (DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=scan-hostname-address)
  (PORT=port)))(CONNECT_DATA=(SERVICE_NAME=service-name))) when the whole Oracle
  RAC database needs to be specified

  (DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=scan-hostname-address)
  (PORT=port)))(CONNECT_DATA=(SERVICE_NAME=service-name)(INSTANCE_NAME=inst1)))
  when a specific Oracle RAC instance needs to be specified
  ```

> **Note:** For more information about SCAN, see Single Client Access Name (SCAN) for the Cluster in the *Oracle Grid Infrastructure Installation Guide* and SCAN Addresses for Simplified Client Access in the *Oracle Real Application Clusters Installation Guide*.

SCAN Run Time Implications and Limitations

Table 4–4 describes supported scenarios when you configure against RAC:

Table 4–4 SCAN Run Time Implications and Limitations

Scenario	Description	High Availability Run time Outcome and Limitations
1. Non SCAN	Multi data source with each subordinate multi data source pointing to a separate RAC instance	Gives run time High Availability of the database connections, which the WLS multi data source implementation manages.
2. SCAN	Multi data source with each subordinate data source pointing to the SCAN address	Gives run time High Availability of the database connections, which the WLS multi data source implementation manages. **Limitation**: Even if you reference a SCAN address, you are using the limited High Availability features of the WLS multi data source.
3. SCAN	A single data source pointing to the SCAN address.	Does not give runtime High Availability of the database Connections. **Limitation**: A SCAN address virtualizes the entry point to the RAC instances, however, if you specify a single Data Source on WLS, doing so does not provide High Availability. The reason for this is that each server is effectively bound to a single RAC instance.
4. SCAN	GridLink data source pointing to the SCAN address	You must have the correct database version that supports SCAN and set up ONS correctly, using the FAN enabled setting to receive the ONS status messages that the database sends.

Part II

Configuring High Availability for Oracle Identity and Access Management Components

Part II describes procedures that are unique to certain component products.

This part contains the following chapters:

- Chapter 5, "Configuring High Availability for Oracle Identity Manager Components"
- Chapter 6, "Configuring High Availability for Oracle Access Management Access Manager Components"
- Chapter 7, "Configuring High Availability for Oracle Adaptive Access Manager Components"
- Chapter 8, "Configuring High Availability for Oracle Access Management Security Token Service"
- Chapter 9, "Configuring High Availability for Identity Federation Components"
- Chapter 10, "Configuring High Availability for Oracle Entitlements Server"
- Chapter 11, "Configuring High Availability for Mobile and Social"
- Chapter 12, "Configuring High Availability for Oracle Privileged Account Manager Components"
- Chapter 13, "Configuring High Availability for Oracle Identity Navigator"
- Chapter 14, "Oracle Unified Directory"
- Chapter 15, "Active-Passive Topologies for Oracle Fusion Middleware High Availability"

5

Configuring High Availability for Oracle Identity Manager Components

This chapter describes how to design and deploy a high availability environment for Oracle Identity Manager.

Oracle Identity Manager is a user provisioning and administration solution that automates the process of adding, updating, and deleting user accounts from applications and directories. It also improves regulatory compliance by providing granular reports that attest to who has access to what. Oracle Identity Manager is available as a stand-alone product or as part of Oracle Identity and Access Management Suite.

Automating user identity provisioning can reduce IT administration costs and improve security. Provisioning also plays an important role in regulatory compliance. Key features of Oracle Identity Manager include password management, workflow and policy management, identity reconciliation, reporting and auditing, and extensibility through adapters.

Oracle Identity Manager provides the following functionalities:

- User Administration
- Workflow and Policy
- Password Management
- Audit and Compliance Management
- User Provisioning
- Organization and Role Management

For details about Oracle Identity Manager, see the *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Manager*.

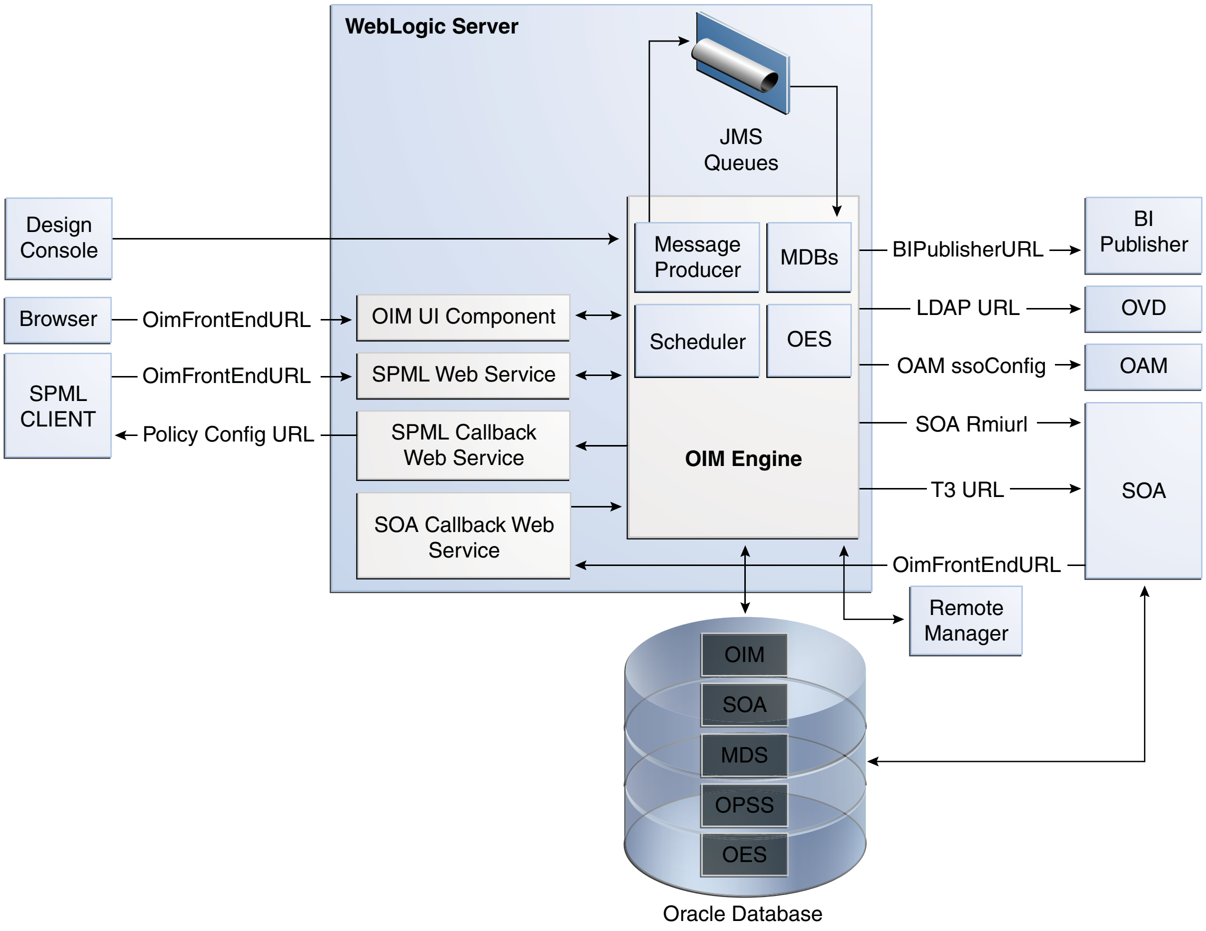
This section includes the following topics:

- Section 5.1, "Oracle Identity Manager Component Architecture"
- Section 5.2, "Oracle Identity Manager High Availability Concepts"
- Section 5.3, "Oracle Identity Manager High Availability Configuration Steps"

5.1 Oracle Identity Manager Component Architecture

Figure 5–1 shows the Oracle Identity Manager architecture:

Figure 5–1 Oracle Identity Manager Component Architecture



5.1.1 Oracle Identity Manager Component Characteristics

Oracle Identity Manager Server is Oracle's self-contained, standalone identity management solution, based on Java EE standards. It provides User Administration, Workflow and Policy, Password Management, Audit and Compliance Management, User Provisioning and Organization and Role Management functionalities.

Oracle Identity Manager is a standard Java EE application that is deployed on Oracle WebLogic Sever and uses a database to store runtime and configuration data. The MDS schema contains configuration information; the runtime and user information is stored in the OIM schema.

Oracle Identity Manager connects to the SOA managed servers over RMI to invoke the SOA EJBs.

Oracle Identity Manager uses the human workflow module of the Oracle SOA Suite to manage its request workflow. Oracle Identity Manager connects to SOA using the T3 URL for the SOA server, which is the front end URL for SOA. Oracle recommends using the load balancer or web server URL for clustered SOA servers. When the workflow completes, SOA calls back Oracle Identity Manager web services using OIMFrontEndURL. Oracle SOA is deployed along with the Oracle Identity Manager.

Several Oracle Identity Manager modules use JMS queues. Each queue is processed by a separate Message Driven Bean (MDB), which is also part of the Oracle Identity Manager application. Message producers are also part of the Oracle Identity Manager application.

Oracle Identity Manager uses embedded Oracle Entitlements Server, which is also part of the Oracle Identity Manager engine. Oracle Entitlements Server (OES) is used for authorization checks inside Oracle Identity Manager. For example, one of the policy constraints determines that only users with certain roles are allowed to create users. This is defined using the Oracle Identity Manager user interface.

Oracle Identity Manager uses a Quartz based scheduler for scheduled activities. There are various scheduled activities that happen in the background. For example, one of the scheduled tasks is to disable users after the end date of the users.

Oracle Identity Manager links to Oracle BI Publisher for all the reporting features. BI Publisher is expected to be in a different domain or same domain, so the integration is only a simple static URL integration. There is no interaction between BI Publisher and Oracle Identity Manager runtime components. BI Publisher is configured to use the same OIM database schema for reporting purposes.

When you enable LDAPSync to communicate directly with external Directory Servers such as Oracle Internet Directory, ODSEE, and Microsoft Active Directory, support for high availability/failover features requires that you configure the Identity Virtualization Library (libOVD).

To configure libOVD, use the WLST command `addLDAPHost`. To manage libOVD, see Managing Identity Virtualization Library (libOVD) Adapters in *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Manager* for a list of WLST commands.

5.1.2 Runtime Processes

Oracle Identity Manager is a Java EE application that is deployed on WebLogic Server as a no-stage application. The Oracle Identity Manager server is initialized when the WebLogic Server it is deployed on starts up. As part of the application initialization, the quartz-based scheduler is also started. Once initialization is done, the system is ready to receive requests from clients.

Remote Manager and Design Console need to be started as standalone utilities separately.

5.1.3 Component and Process Lifecycle

Oracle Identity Manager is deployed to an Oracle WebLogic Server as an externally managed application. By default, the WebLogic Server starts, stops, monitors and manages other lifecycle events for the Oracle Identity Manager application.

Oracle Identity Manager is a standard Java EE application that starts after the application server components start. Also, Oracle Identity Manager uses the authenticator which is part of the Oracle Identity Manager component mechanism; it starts up before the WebLogic JNDI are initialized and the application is started. This is loaded from the OIM ORACLE_HOME.

Oracle Identity Manager uses a Quartz technology-based scheduler. Quartz starts the scheduler thread on all the WebLogic Server instances. It uses the database as the centralized storage for picking and running the scheduled activities. If one of the scheduler instances picks up a job, the other instances will not pick up that same job.

Oracle Identity Manager caches certain system configuration values in the cache in memory of the server instance from the database. These caches load independently and are not shared among the servers. Any changes to the system configuration triggers cache cleanup, which notifies all servers in the cluster. Oracle Identity Manager uses oscache, jgroups for this purpose. Jgroups use multicast addresses. A valid multicast address is randomly generated during installation and seeded to the Oracle Identity Manager metadata repository file.

WebLogic Node Manager can be configured to monitor the server process and restart it in case of failure.

The Oracle Enterprise Manager Fusion Middleware Control is used to monitor as well as to modify the configuration of the application.

5.1.4 Starting and Stopping Oracle Identity Manager

You can manage Oracle Identity Manager lifecycle events with one or more of the following command line tools and consoles:

- Oracle WebLogic Scripting Tool (WLST)
- WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle WebLogic Node Manager

5.1.5 Configuration Artifacts

The configuration for the Oracle Identity Manager server is stored in the MDS repository and is managed using MBeans. The oim-config.xml file is the main configuration file. Manage the OIM configuration using the MBean browser through the Oracle Enterprise Manager Fusion Middleware Control or through the command line MDS utilities. The oim-config.xml file is stored in the /db/oim-config.xml location of the MDS repository.

For more information about the MDS utilities, see the MDS utilities section of the *Oracle Fusion Middleware Developer's Guide for Oracle Identity Manager*.

JMS is configured out-of-the-box by the installer. All the necessary JMS queues, connection pools, data sources and so on are configured on the WebLogic application servers. The following queues are created when Oracle Identity Manager deploys:

- oimAttestationQueue
- oimAuditQueue
- oimDefaultQueue
- oimKernelQueue
- oimProcessQueue
- oimReconQueue
- oimSODQueue

The `xlconfig.xml` file stores Design Console and Remote Manager configuration.

5.1.6 External Dependencies

Oracle Identity Manager is a Java EE application that is deployed on an Oracle WebLogic Managed Server. Oracle Identity Manager uses the Worklist and Human workflow modules of the Oracle SOA Suite for request flow management. Oracle Identity Manager interacts with external repositories to store configuration and runtime data. Oracle Identity Manager requires these repositories to be available during initialization and during runtime. All Oracle Identity Manager credentials are stored in the OIM repository. The external components required for the Oracle Identity Manager server to function are listed below:

- WebLogic Server
  - Administration Server
  - Managed Server
- Data Repositories
  - Configuration Repository (MDS Schema)
  - Runtime Repository (OIM Schema)
  - User Repository (OIM Schema)
  - SOA Repository (SOA Schema)
- External LDAP Stores (when using LDAP Sync)
- BI Publisher

The Design Console is a tool used by the administrator for development and customization. The Design Console communicates directly with the Oracle Identity Manager engine, so it relies on the same components that the Oracle Identity Manager server relies on.

Remote Manager is an optional independent standalone application, which calls the custom APIs on the local system. So it needs the JAR files for those custom APIs in its classpath.

5.1.7 Oracle Identity Manager Log File Locations

Oracle Identity Manager is a Java EE application deployed on Oracle WebLogic Server. All server related logs messages are logged in the server log file and all Oracle Identity Manager specific messages are logged into the diagnostic log file of the WebLogic Server where the application is deployed.

The WebLogic Server log files are located under the following directory:

```
DOMAIN_HOME/servers/serverName/logs
```

The three main log files are *serverName*.log, *serverName*.out, and *serverName*-diagnostic.log, where *serverName* is the name of the WebLogic Server. For example, if the WebLogic Server name is wls_OIM1, then the diagnostic log file name would be wls_OIM1-diagnostic.log.

You can view the log files using the Oracle Enterprise Manager Fusion Middleware Control.

5.2 Oracle Identity Manager High Availability Concepts

This section provides an introduction to Oracle Identity Management high availability concepts and describes how to design and deploy a high availability environment for Oracle Identity Manager.

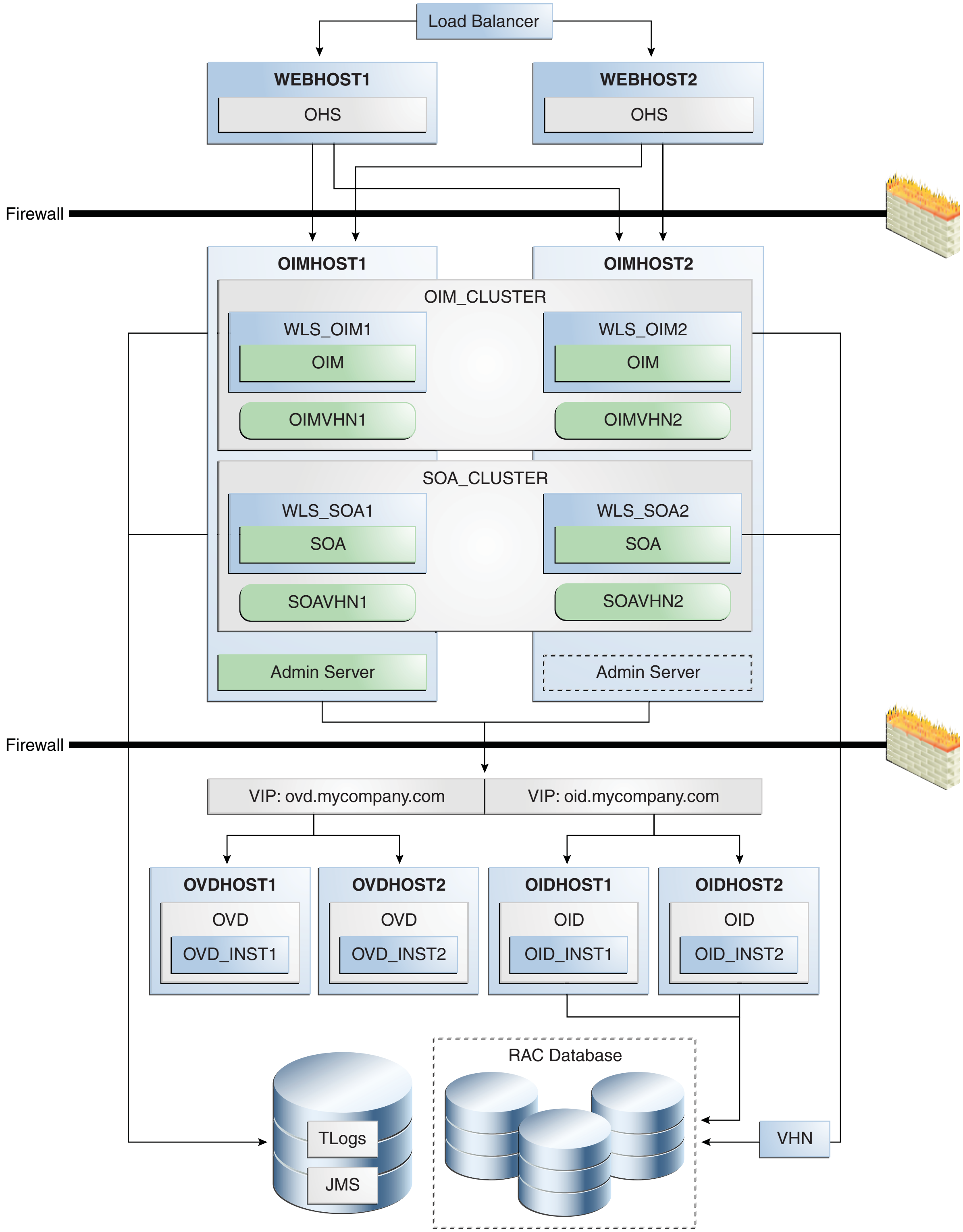
Note: Be aware of the following when you deploy Oracle Identity Manager in a high availability configuration:

- Oracle Identity Manager can be deployed on an Oracle RAC database, but Oracle RAC failover is not transparent for Oracle Identity Manager in this release. If an Oracle RAC failover occurs, end users may have to resubmit their requests.
- Oracle Identity Manager always requires that at least one of the nodes in the SOA cluster be available. If the SOA cluster is not available, end user requests will fail. Oracle Identity Manager does not retry for a failed SOA call. Therefore, the end user must retry when a SOA call fails.

5.2.1 Oracle Identity Manager High Availability Architecture

Figure 5–2 shows Oracle Identity Manager deployed in a high availability architecture in an active-active configuration.

Figure 5–2 Oracle Identity Manager High Availability Architecture



On OIMHOST1, the following installations have been performed:

- An Oracle Identity Manager instance has been installed in the WLS_OIM1 Managed Server and a SOA instance has been installed in the WLS_SOA1 Managed Server.

- The Oracle RAC database has been configured in a JDBC multi data source to protect the instance from Oracle RAC node failure.
- A WebLogic Server Administration Server has been installed. Under normal operations, this is the active Administration Server.

On OIMHOST2, the following installations have been performed:

- An Oracle Identity Manager instance has been installed in the WLS_OIM2 Managed Server and a SOA instance has been installed in the WLS_SOA2 Managed Server.
- The Oracle RAC database has been configured in a JDBC multi data source to protect the instance from Oracle RAC node failure.
- The instances in the WLS_OIM1 and WLS_OIM2 Managed Servers on OIMHOST1 and OIMHOST2 are configured as the OIM_Cluster cluster.
- The instances in the WLS_SOA1 and WLS_SOA2 Managed Servers on OIMHOST1 and OIMHOST2 are configured as the SOA_Cluster cluster.
- A WebLogic Server Administration Server has been installed. Under normal operations, this is the passive Administration Server. You make this Administration Server active if the Administration Server on OIMHOST1 becomes unavailable.

The following virtual host names are used in the Oracle Identity Manager high availability configuration in Figure 5–2:

- OIMVHN1 is the virtual hostname that maps to the listen address for the WLS_OIM1 managed server, and it fails over with server migration of the WLS_OIM1 managed server. It is enabled on the node where the WLS_OIM1 managed server is running (OIMHOST1 by default).
- OIMVHN2 is the virtual hostname that maps to the listen address for the WLS_OIM2 managed server, and it fails over with server migration of the WLS_OIM2 managed server. It is enabled on the node where the WLS_OIM2 managed server is running (OIMHOST2 by default).
- SOAVHN1 is the virtual hostname that is the listen address for the WLS_SOA1 managed server, and it fails over with server migration of the WLS_SOA1 managed server. It is enabled on the node where the WLS_SOA1 managed server is running (OIMHOST1 by default).
- SOAVHN2 is the virtual hostname that is the listen address for the WLS_SOA2 managed server, and it fails over with server migration of the WLS_SOA2 managed server. It is enabled on the node where the WLS_SOA2 managed server is running (OIMHOST2 by default).
- VHN refers to the virtual IP addresses for the Oracle Real Application Clusters (Oracle RAC) database hosts.

5.2.2 Starting and Stopping the Oracle Identity Manager Cluster

In a high availability architecture, Oracle Identity Manager is deployed on an Oracle WebLogic cluster that has at least two servers as members of the cluster.

By default, WebLogic Server starts, stops, monitors, and manages the various lifecycle events for the application. The Oracle Identity Manager application leverages the high availability features of the underlying Oracle WebLogic clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

You can use one or more of the following command line tools and consoles to manage Oracle Identity Manager lifecycle events:

- WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle WebLogic Scripting Tool (WLST)

5.2.3 Cluster-Wide Configuration Changes

For high availability environments, changing the configuration of one Oracle Identity Manager instance changes the configuration of all the other instances, because all the Oracle Identity Manager instances share the same configuration repository.

5.2.4 Considerations for Synchronizing with LDAP

Synchronization information between LDAP and the Oracle Identity Manager database is handled by a process called Reconciliation, which is a scheduled process that runs in the background primarily. It is also possible to manually run this process.

If an LDAP outage occurs during the Synchronization process, the data which did not get into Oracle Identity Manager will be picked up during the next run of the reconciliation task.

5.3 Oracle Identity Manager High Availability Configuration Steps

This section provides high-level instructions for setting up a high availability deployment for Oracle Identity Manager.

This section includes the following topics:

- Section 5.3.16, "Configuring a Default Persistence Store for Transaction Recovery"
- Section 5.3.17, "Install Oracle HTTP Server on WEBHOST1 and WEBHOST2"
- Section 5.3.18, "Configuring Oracle Identity Manager to Work with the Web Tier"
- Section 5.3.19, "Validate the Oracle HTTP Server Configuration"
- Section 5.3.20, "Oracle Identity Manager Failover and Expected Behavior"
- Section 5.3.21, "Troubleshooting Oracle Identity Manager High Availability"
- Section 5.3.22, "Scaling Up and Scaling Out the Oracle Identity Manager Topology"

5.3.1 Prerequisites for Oracle Identity Manager Configuration

Before you configure Oracle Identity Manager for high availability, you must:

- Run the Repository Creation Utility to create the OIM schemas in a database. See Section 5.3.1.1, "Running RCU to Create the OIM Schemas in a Database" for instructions.
- Install WebLogic Server on OIMHOST1 and OIMHOST2. See Section 5.3.1.1.1, "Installing Oracle WebLogic Server" for instructions.
- Install the Oracle SOA Suite on OIMHOST1 and OIMHOST2. See Section 5.3.1.2, "Installing the Oracle SOA Suite on OIMHOST1 and OIMHOST2" for instructions.
- Install the Oracle Identity Management software on OIMHOST1 and OIMHOST2. See Section 5.3.1.3, "Installing the Oracle Identity Manager on OIMHOST1 and OIMHOST2" for instructions.
- Ensure that a highly available LDAP implementation is available.

  Note: This section is required only for LDAPSync-enabled Oracle Identity Manager installations and for Oracle Identity Manager installations that integrate with Oracle Access Management.

  If you are not planning to enable the LDAPSync option or to integrate with Oracle Access Management, you can skip this section.

  Note that Oracle Identity Manager does not communicate directly with Oracle Internet Directory. It communicates with Oracle Virtual Directory, which communicates with Oracle Internet Directory.
- Create the `wlfullclient.jar` file on OIMHOST1 and OIMHOST2.

  Oracle Identity Manager requires the `wlfullclient.jar` library for certain operations. For example, the Design Console uses the library for server connections. Oracle does not ship this library; you must create it manually. Oracle recommends creating this library under the *MW_HOME*`/wlserver_10.3/server/lib` directory on all the machines in the application tier of your environment. You do not need to create this library on directory tier machines such as OIDHOST1, OIDHOST2, OVDHOST1 and OVDHOST2. See Developing a WebLogic Full Client in the guide *Oracle Fusion Middleware Programming Stand-alone Clients for Oracle WebLogic Server* for more information.

  To create the wlfullclient.jar file:

  1. Go to the *MW_HOME*`/wlserver_10.3/server/lib` directory.

2. Set your *JAVA_HOME* to *MW_HOME*/`jdk160_18` and ensure that your JAVA_HOME/`bin` directory is in your path.

3. Create the `wlfullclient.jar` file by running:

   java -jar wljarbuilder.jar

5.3.1.1 Running RCU to Create the OIM Schemas in a Database

Before you can install the Oracle Identity Manager and SOA instances on OIMHOST1 and OIMHOST2, you must use the Repository Creation Utility (RCU) to create the collection of schemas used by Oracle Identity Manager.

RCU ships on its own CD as part of the Oracle Fusion Middleware 11*g* kit.

To run RCU and create the Oracle Identity Manager schemas in an Oracle RAC database repository:

1. Issue this command:

   ```
   prompt> RCU_HOME/bin/rcu &
   ```

2. On the Welcome screen, click **Next**.

3. On the Create Repository screen, select the **Create** operation to load the component schemas into an existing database.

   Click **Next**.

4. On the Database Connection Details screen, enter connection information for the existing database as follows:

   - **Database Type**: Oracle Database
   - **Host Name**: Name of the computer on which the database is running. For an Oracle RAC database, specify the VIP name or one node name. Example: OIMDBHOST1-VIP or OIMDBHOST2-VIP
   - **Port**: The port number for the database. For example: 1521
   - **Service Name**: The service name of the database. For example: `oim.example.com`
   - **Username**: SYS
   - **Password**: The SYS user password
   - **Role**: SYSDBA

   Click **Next**.

5. Click **OK** on the Checking Prerequisites screen after the Global Prerequisites complete successfully.

6. On the Select Components screen, create a new prefix and select the components to be associated with this deployment:

   Create a New Prefix: ha

   Components:

   - Under **Identity Management**:
     - Oracle Identity Manager - OIM
     - Oracle Platform Security Services - OPSS
     - Note that Metadata Services - MDS is selected by default.

- Under **SOA and BAM Infrastructure**:
  - SOA Infrastructure - SOAINFRA is selected by default.
  - User Messaging Service - ORASDPM is selected by default.

Click **Next**.

7. Click **OK** on the Checking Prerequisites screen after the Component Prerequisites complete successfully.
8. On the Schema Passwords screen, enter the passwords for the mail and additional (auxiliary) schema users.

   Click **Next**.
9. On Map Tablespaces screen, select the tablespaces for the components.
10. On the Summary screen, click **Create**.
11. On the Completion Summary screen, click **Close**.

5.3.1.1.1 Installing Oracle WebLogic Server Prior to installing the WebLogic Server, ensure that your machines meet the system, patch, kernel, and other requirements that the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server* specifies.

Start the installer, then perform these steps on OIMHOST1 and OIMHOST2:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, select **Create a New Middleware Home**.

   For Middleware Home Directory, enter:

   ORACLE_BASE`/product/fmw`

   > **Note:** *ORACLE_BASE* is the base directory under which Oracle products are installed. The recommended value is `/u01/app/oracle`.

   Click **Next**.
3. On the Register for Security Updates screen, enter your contact information so that you can be notified of security updates.

   Click **Next**.
4. On the Choose Install Type screen, select **Custom**.

   Click **Next**.
5. On the Choose Products and Components screen, select only **Oracle JRockit SDK**, and click **Next**.
6. On the Choose Product Installation Directories screen, accept the directory *ORACLE_BASE*`/product/fmw/wlserver_10.3`.

   Click **Next**.
7. On the Installation Summary screen, click **Next**.
8. On the Installation Complete screen, deselect **Run Quickstart**.

   Click **Done**.

5.3.1.2 Installing the Oracle SOA Suite on OIMHOST1 and OIMHOST2

Perform these steps to install Oracle Fusion Middleware Components on OIMHOST1 and OIMHOST2.

If the `/etc/oraInst.loc` file exists, verify that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory. If the `/etc/oraInst.loc` file does not exist, you can skip this step.

Start the installer for Oracle Fusion Middleware Component as follows:

```
HOST1> runInstaller
```

When the installer prompts you for a `JRE/JDK` location, enter the Oracle SDK location created in the WebLogic Server installation, for example, *`ORACLE_BASE`*`/product/fmw/jrockit_160_14_`<*version*>.

Then proceed as follows:

1. On the Welcome screen, click **Next**.
2. On the Prerequisite Checks screen, verify that the checks complete successfully, then click **Next**.
3. On the Specify Installation Location screen, enter the following values:
   - Oracle Middleware Home: Select the previously installed Middleware home from the list for `MW_HOME`, for example

     ```
     /u01/app/oracle/product/fmw
     ```
   - Oracle Home Directory:
     - Enter `soa` as the Oracle home directory name when installing the Oracle SOA Suite in the `ORACLE_HOME`

   Click **Next**.
4. On the Installation Summary screen, click **Install**.
5. On the Installation Complete screen, click **Finish**.

5.3.1.3 Installing the Oracle Identity Manager on OIMHOST1 and OIMHOST2

Perform these steps to install Oracle Fusion Middleware Components on OIMHOST1 and OIMHOST2.

If the `/etc/oraInst.loc` file exists, verify that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory. If the `/etc/oraInst.loc` file does not exist, you can skip this step.

Start the installer for Oracle Fusion Middleware Component as follows:

```
HOST1> runInstaller
```

When the installer prompts you for a `JRE/JDK` location, enter the Oracle SDK location created in the WebLogic Server installation, for example, *`ORACLE_BASE`*`/product/fmw/jrockit_160_`<*version*>.

Then proceed as follows:

1. On the Welcome screen, click **Next**.
2. On the Prerequisite Checks screen, verify that the checks complete successfully, then click **Next**.
3. On the Specify Installation Location screen, enter the following values:

- Oracle Middleware Home: Select the previously installed Middleware home from the list for MW_HOME, for example

  ```
  /u01/app/oracle/product/fmw
  ```

- Oracle Home Directory:
  - Enter iam as the Oracle home directory name when installing the Oracle Identity and Access Management Suite in the ORACLE_HOME

Click **Next**.

4. On the Installation Summary screen, click **Install**.
5. On the Installation Complete screen, click **Finish**.

5.3.2 Creating and Configuring the WebLogic Domain for OIM and SOA on OIMHOST1

To create the domain on OIMHOST1:

1. Start the Configuration Wizard by running this command:

   ```
   MW_HOME/oracle_common/common/bin/config.sh
   ```

2. On the Welcome screen, select **Create a WebLogic Domain**.

   Click **Next**.

3. On the Select Domain Source screen, select **Generate a domain configured automatically to support the following added products**.

   From the list, select:

   - **Oracle Identity Manager**

     Note that Oracle SOA Suite, Oracle Enterprise Manager, Oracle Platform Security Service, Oracle JRF, Oracle JRF WebServices Asynchronous services, and Oracle WSM Policy Manager are selected automatically.

   - **Oracle Enterprise Manager**

     Note that Oracle JRF is selected automatically.

4. On the Specify Domain Name and Location screen, enter the name and location for the domain and all its applications.

   Provide the following:

   - **Domain Name**: IDMDomain
   - **Domain Location**: Accept the default.
   - **Application Location**: Accept the default.

   Click **Next**.

5. On the Configure Administration Server Username and Password screen, provide the following:

   - **Name**: weblogic
   - **User Password**: Enter the password for the weblogic user.
   - **Confirm User Password**: Enter the password for the weblogic user.
   - **Description**: Provide a description for the user.

   Click **Next**.

6. On the Configure Server Start Mode and JDK screen, select **Production Mode** and **JRockit SDK 1.6.n**.

   Click **Next**.

7. On the Configure JDBC Component Schemas screen, select the Component Schemas shown below:

   - SOA Infrastructure
   - User Messaging Service
   - OIM MDS Schema
   - OWSM MDS Schema
   - SOA MDS Schema
   - OIM Infrastructure
   - OPSS Schema

   Select the check box next to **Configure selected component schemas as RAC multi data source schemas** in the next panel.

   > **Note:** You can also use GridLink data sources. See Section 3.13, "GridLink Data Sources" for more information.

   Click **Next**.

8. On the Configure RAC Multi Data Source Component Schemas screen, select all the multi data source schemas and enter the following:

   - **Service Name**: `oim.example.com`
   - For the first RAC node:
     - **Host Name**: `OIMDBHOST1-VIP.example.com`
     - **Instance Name**: `oimdb1`
     - **Port**: 1521
   - For the second RAC node:
     - **Host Name**: `OIMDBHOST2-VIP.example.com`
     - **Instance Name**: `oimdb2`
     - **Port**: 1521

   Select each schema individually and enter the schema's username and password, as shown in Table 5–1:

Table 5–1 Entering the Schema Owner and Password for Each multi data Source Schema

Schema Name	Schema Owner	Password
SOA Infrastructure	HA_SOAINFRA	<enter the password>
User Messaging Service	HA_ORASDPM	<enter the password>
OIM MDS Schema	HA_MDS	<enter the password>
OWSM MDS Schema	HA_MDS	<enter the password>
SOA MDS Schema	HA_MDS	<enter the password>

Table 5–1 (Cont.) Entering the Schema Owner and Password for Each multi data Source Schema

Schema Name	Schema Owner	Password
OIM Schema	HA_OIM	*<password>*
OPSS Schema	HA_OPSS	*<password>*

Click **Next**.

9. On the Test Component Schema screen, select **All the Schemas** and then click **Test Connections**. Validate that the test for all the schemas completed successfully.

   Click **Next**.

10. On the Select Optional Configuration screen, select:

    - **Administration Server**
    - **JMS Distributed Destination** (required only on the domain that has OIM)
    - **Managed Servers, Clusters and Machines**
    - **JMS File Store** (required only on the domain that has OIM)

    Click **Next**.

11. In the Configure the Administration Server screen, enter the following values:

    - **Name**: AdminServer
    - **Listen Address**: oimhost1.example.com
    - **Listen Port**: 7001
    - **SSL listen port**: Not applicable
    - **SSL enabled**: Leave unchecked

    Click **Next**.

12. On the Select JMS Distributed Destination Type screen, ensure that all JMS System Resources listed are Uniform Distributed Destinations. If they are not, select **UDD** from the drop down box. Validate that the entries look like those in Table 5–2:

Table 5–2 Values to Choose for JMS System Resources

JMS System Resource	Uniform/Weighted Distributed Destination
UMSJMSSystemResource	UDD
SOAJMSModule	UDD
OIMJMSModule	UDD
BPMJMSModule	UDD

Click **Next**.

An Override Warning box with the following message opens:

"CFGFWK-40915: At least one JMS system resource has been selected for conversion to a Uniform Distributed Destination (UDD). This conversion will take place only if the JMS System resource is assigned to a cluster."

Click **OK** on the Warning box.

13. When you first enter the Configure Managed Servers screen, the configuration wizard will have created two default managed servers (oim_server1 and soa_server1) for you. Change the details of the default managed servers and then add the second managed server. Follow the steps below:

    For the oim_server1 entry, change the entry to the following values:

    - **Name**: WLS_OIM1
    - **Listen Address**: OIMVHN1
    - **Listen Port**: 14000

    For the soa_server1 entry, change the entry to the following values:

    - **Name**: WLS_SOA1
    - **Listen Address**: SOAVHN1
    - **Listen Port**: 8001

    For the second OIM Server, click **Add** and supply the following information:

    - **Name**: WLS_OIM2
    - **Listen Address**: OIMVHN2
    - **Listen Port**: 14000

    For the second SOA Server, click **Add** and supply the following information:

    - **Name**: WLS_SOA2
    - **Listen Address**: SOAVHN2
    - **Listen Port**: 8001

    Click **Next**.

    > **Note:** Follow the steps for adding the second managed server to add additional managed servers.

14. On the Configure Clusters screen, create two clusters by clicking **Add**.

    Supply the following information for the OIM Cluster:

    - **Name**: oim_cluster
    - **Cluster Messaging Mode**: unicast
    - **Cluster Address**: *oimhost1*:14000,*oimhost2*:14000

    Supply the following information for the SOA Cluster:

    - **Name**: soa_cluster
    - **Cluster Messaging Mode**: unicast
    - **Cluster Address:** *oimhost1*:8001,*oimhost2*:8001

    Leave all the other fields at the default settings and click **Next**.

15. On the Assign Servers to Clusters screen, associate the managed servers with the cluster. Click on the cluster name in the right window.

    Click on the managed server under **servers**, then click on the arrow to assign it to the cluster.

Assign the **WLS_OIM1 and WLS_OIM2** managed servers to be members of the **oim_cluster**.

Assign the **WLS_SOA1 and WLS_SOA2** managed servers to be members of the **soa_cluster**.

Click **Next**.

16. On the Configure Machines screen, create a machine for each host in the topology.

    Click on the **Unix** tab if your hosts use a Unix operating system or click **Machines**.

    Provide the following information:

    - **Name**: Name of the host. A good practice is to use the DNS name here.
    - **Node Manager Listen Address**: Enter the DNS name of the machine here.
    - **Node Manager Port**: Supply a port for Node Manager to use.

    Click **Next**.

    > **Note:** On UNIX, delete the default local machine entry under the **Machines** tab.

17. On the Assign Servers to Machines screen, you assign the managed servers that will run on the machines you just created. Follow these steps:

    Click on a machine in the right hand window.

    Click on the managed servers you want to run on that machine in the left window.

    Click on the arrow to assign the managed servers to the machine.

    Repeat these steps until all managed servers are assigned to the appropriate machine.

    A typical example would be:

    - Host1: **Admin Server, WLS_OIM1, and WLS_SOA1**
    - Host2: **WLS_OIM2 and WLS_SOA2**

    Click **Next**.

18. On the Configure JMS File Stores screen, update the directory locations for the JMS file stores. Enter the following values:

Name	Directory
UMSJMSFileStore_auto_1	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/UMSJMSFileStore_auto_1`
UMSJMSFileStore_auto_2	/u01/app/oracle/admin/domain_name/soa_cluster/jms/UMSJMSFileStore_auto_2
BPMJMSFileStore_auto_1	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/BPMJMSFileStore_auto_1`
BPMJMSFileStore_auto_2	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/BPMJMSFileStore_auto_2`
JRFWSAsyncFileStore_auto_1	/u01/app/oracle/admin/domain_name/oim_cluster/jms/JRFWSAsyncFileStore_auto_1
JRFWSAsyncFileStore_auto_2	/u01/app/oracle/admin/domain_name/oim_cluster/jms/JRFWSAsyncFileStore_auto_2

Name	Directory
SOAJMSFileStore_auto_1	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/SOAJMSFileStore_auto_1`
SOAJMSFileStore_auto_2	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/SOAJMSFileStore_auto_2`
OIMJMSFileStore_auto_1	`/u01/app/oracle/admin/domain_name/oim_cluster/jms/OIMJMSFileStore_auto_1`
OIMJMSFileStore_auto_2	`/u01/app/oracle/admin/domain_name/oim_cluster/jms/OIMJMSFileStore_auto_2`
PS6SOAJMSFileStore_auto_1	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/PS6SOAJMSFileStore_auto_1`
PS6SOAJMSFileStore_auto_2	`/u01/app/oracle/admin/domain_name/soa_cluster/jms/PS6SOAJMSFileStore_auto_2`

19. On the Configuration Summary screen, click **Create** to create the domain.

5.3.3 Upgrading Oracle Platform Security Services Schemas

You must upgrade the Oracle Platform Security Services schemas using Patch Set Assistant. To do this, complete the following steps:

1. Start the Patch Set Assistant from the location MW_HOME/oracle_common/bin using the following command:

   ```
   ./psa
   ```

2. Select **opss**.
3. Specify the Database connection details and select the schema to be upgraded.
4. After you upgrade Oracle Platform Security Services schema, verify the upgrade by checking the log file at the location *MW_HOME*`/oracle_common/upgrade/logs/psa`<*timestamp*>`.log`

   The timestamp refers to the actual date and time when Patch Set Assistant was run. If the upgrade fails, check the log files to fix the errors and run the Patch Set Assistant again.

5.3.4 Configuring the Database Security Store for the Domain

The Database Security Store contains security policies, audit meta-data, credentials, keys, and other security artifacts managed by OPSS. It facilitates high availability by serving as the single point of truth for policies in a topology, regardless of the domain organization (single, multiple, or extended).

You must configure the Database Security Store after you configure the domain but before you start the Administration Server. See Configuring Database Security Store for an Oracle Identity and Access Management Domain in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for more information.

5.3.5 Post-Installation Steps on OIMHOST1

This section describes the post-installation steps to perform on OIMHOST1. It includes these topics:

- Section 5.3.5.2, "Update Node Manager on OIMHOST1"

- Section 5.3.5.3, "Start Node Manager on OIMHOST1"
- Section 5.3.5.4, "Start the Administration Server on OIMHOST1"

5.3.5.1 Creating boot.properties for the Administration Server on OIMHOST1

This section describes how to create a boot.properties file for the Administration Server on OIMHOST1. The boot.properties file enables the Administration Server to start without prompting for the administrator username and password.

To create the boot.properties file:

1. On OIMHOST1, create the following directory:

   ```
   MW_HOME/wlserver_10.3/server/bin
   ```

   For example:

   ```
   $ mkdir -p
   MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
   ```

2. Use a text editor to create a file called boot.properties under the security directory. Enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

> **Note:** When you start the Administration Server, the username and password entries in the file get encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

5.3.5.2 Update Node Manager on OIMHOST1

Before you start the managed servers using the Administration Console, Node Manager requires that the StartScriptEnabled property be set to true.

To do this, run the setNMProps.sh script located under the following directory:

```
MW_HOME/oracle_common/common/bin
```

5.3.5.3 Start Node Manager on OIMHOST1

Start the Node Manager on OIMHOST1 using the startNodeManager.sh script located under the following directory:

```
MW_HOME/wlserver_10.3/server/bin
```

5.3.5.4 Start the Administration Server on OIMHOST1

Follow these steps to start the Administration Server and validate its startup:

1. Start the Administration Server on OIMHOST1 by issuing the command:

   ```
   DOMAIN_HOME/bin/startWebLogic.sh
   ```

2. Validate that the Administration Server started up successfully by opening a web browser and accessing the following pages:

   - Administration Console at:

```
http://oimhost1.example.com:7001/console
```

- Oracle Enterprise Manager Fusion Middleware Control at:

```
http://oimhost1.example.com:7001/em
```

Log into these consoles using the `weblogic` user credentials.

5.3.6 Configuring Oracle Identity Manager on OIMHOST1

This section describes how to configure the Oracle Identity Manager and SOA managed servers before starting them.

This section includes the following topics:

- Section 5.3.6.1, "Prerequisites for Configuring Oracle Identity Manager"
- Section 5.3.6.2, "Updating the Coherence Configuration for the Coherence Cluster"
- Section 5.3.6.3, "Running the Oracle Identity Management Configuration Wizard"

5.3.6.1 Prerequisites for Configuring Oracle Identity Manager

Before configuring Oracle Identity Manager, ensure that the following tasks have been performed:

> **Note:** This section is required only for LDAPSync-enabled Oracle Identity Manager installations and for Oracle Identity Manager installations that integrate with Oracle Access Management.
>
> If you are not planning to enable the LDAPSync option or to integrate with Oracle Access Management, you can skip this section.

1. "Extending the Directory Schema for Oracle Identity Manager"
2. "Creating Users and Groups for Oracle Identity Manager"
3. "Creating Adapters in Oracle Virtual Directory"

Extending the Directory Schema for Oracle Identity Manager

Pre-configuring the Identity Store extends the schema in the back end directory regardless of directory type.

To pre-configure the Identity Store, perform these steps on OIMHOST1:

1. Set the environment variables `MW_HOME`, `JAVA_HOME` and `ORACLE_HOME`.

   Set `ORACLE_HOME` to `IAM_ORACLE_HOME`.

2. Create a properties file `extend.props` that contains the following:

   ```
   IDSTORE_HOST : idstore.example.com
   IDSTORE_PORT : 389
   IDSTORE_BINDDN: cn=orcladmin
   IDSTORE_USERNAMEATTRIBUTE: cn
   IDSTORE_LOGINATTRIBUTE: uid
   IDSTORE_USERSEARCHBASE: cn=Users,dc=example,dc=com
   IDSTORE_GROUPSEARCHBASE: cn=Groups,dc=example,dc=com
   IDSTORE_SEARCHBASE: dc=example,dc=com
   IDSTORE_SYSTEMIDBASE: cn=systemids,dc=example,dc=com
   ```

   Where:

- IDSTORE_HOST and IDSTORE_PORT are, respectively, the host and port of your Identity Store directory. If you are using a non-OID directory, then specify the Oracle Virtual Directory host (which should be IDSTORE.example.com.)
- IDSTORE_BINDDN is an administrative user in the Identity Store Directory
- IDSTORE_USERSEARCHBASE is the location in the directory where Users are Stored.
- IDSTORE_GROUPSEARCHBASE is the location in the directory where Groups are Stored.
- IDSTORE_SEARCHBASE is the location in the directory where Users and Groups are stored.
- IDSTORE_SYSTEMIDBASE is the location of a container in the directory where users can be placed when you do not want them in the main user container. This happens rarely but one example is the Oracle Identity Manager reconciliation user which is also used for the bind DN user in Oracle Virtual Directory adapters.

3. Configure the Identity Store using the command idmConfigTool which is located at IAM_ORACLE_HOME/idmtools/bin.

   The command syntax is:

   ```
   idmConfigTool.sh -preConfigIDStore input_file=configfile
   ```

   For example:

   ```
   idmConfigTool.sh -preConfigIDStore input_file=extend.props
   ```

   After the command runs, the system prompts you to enter the password of the account with which you are connecting to the ID Store.

   Sample command output:

   ```
   ./preconfig_id.sh
   Enter ID Store Bind DN password :
   Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idm_idstore_groups_
   template.ldif
   Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idm_idstore_groups_acl_
   template.ldif
   Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/systemid_pwdpolicy.ldif
   Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFileINFO: ->
   LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idstore_tuning.ldifApr
   5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oid_schema_extn.ldif
   The tool has completed its operation. Details have been logged to
   automation.log
   ```

4. Check the log file for any errors or warnings and correct them.

Creating Users and Groups for Oracle Identity Manager

Follow the steps in the procedure to add the oimadmin user to the Identity Store and assign it to an Oracle Identity Manager administrative group. You must also create a user outside of the standard cn=Users location to perform reconciliation. Oracle recommends that you select this user as the bind DN when connecting to directories with Oracle Virtual Directory.

> **Note:** This command also creates a container in your Identity Store for reservations.

To add the xelsysadm user to the Identity Store and assign it to an administrative group, perform the following tasks on OIMHOST1:

1. Set the Environment Variables: MW_HOME, JAVA_HOME, IDM_HOME, and ORACLE_HOME

   Set IDM_HOME to IDM_ORACLE_HOME

   Set ORACLE_HOME to IAM_ORACLE_HOME

2. Create a properties file oim.props that contains the following:

```
IDSTORE_HOST : idstore.example.com
IDSTORE_PORT : 389
IDSTORE_BINDDN : cn=orcladmin
IDSTORE_USERNAMEATTRIBUTE: cn
IDSTORE_LOGINATTRIBUTE: uid
IDSTORE_USERSEARCHBASE:cn=Users,dc=example,dc=com
IDSTORE_GROUPSEARCHBASE: cn=Groups,dc=us,dc=oracle,dc=com
IDSTORE_SEARCHBASE: dc=example,dc=com
POLICYSTORE_SHARES_IDSTORE: true
IDSTORE_SYSTEMIDBASE: cn=systemids,dc=example,dc=com
IDSTORE_OIMADMINUSER: oimadmin
IDSTORE_OIMADMINGROUP:OIMAdministrators
```

   Where:

   - IDSTORE_HOST and IDSTORE_PORT are, respectively, the host and port of your Identity Store directory. Specify the back-end directory here, rather than OVD.
   - IDSTORE_BINDDN is an administrative user in the Identity Store Directory
   - IDSTORE_OIMADMINUSER is the name of the administration user you would like to use to log in to the Oracle Identity Manager console.
   - IDSTORE_OIMADMINGROUP is the name of the group you want to create to hold your Oracle Identity Manager administrative users.
   - IDSTORE_USERSEARCHBASE is the location in your Identity Store where users are placed.
   - IDSTORE_GROUPSEARCHBASE is the location in your Identity Store where groups are placed.
   - IDSTORE_SYSTEMIDBASE is the location in your directory where the Oracle Identity Manager reconciliation user are placed.
   - POLICYSTORE_SHARES_IDSTORE is set to true if your Policy and Identity stores are in the same directory. If not, it is set to false.

3. Configure the Identity Store by using the command `idmConfigTool` located at `IAM_ORACLE_HOME/idmtools/bin`:

   ```
   idmConfigTool.sh -prepareIDStore mode=OIM input_file=configfile
   ```

   For example:

   ```
   idmConfigTool.sh -prepareIDStore mode=OIM input_file=oim.props
   ```

   When the command runs, the system prompts you for the account password with which you are connecting to the Identity Store. The system also requests the passwords you want to assign to the accounts:

   ```
   IDSTORE_OIMADMINUSER
   oimadmin
   ```

   Oracle recommends that you set `oimadmin` to the same value as the account you create as part of the Oracle Identity Manager configuration.

   Sample command output:

   ```
   Enter ID Store Bind DN password :
   *** Creation of oimadmin ***
   Apr 5, 2011 4:58:51 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oim_user_template.ldif
   Enter User Password for oimadmin:
   Confirm User Password for oimadmin:
   Apr 5, 2011 4:59:01 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oim_group_template.ldif
   Apr 5, 2011 4:59:01 AM oracle.ldap.util.LDIFLoader loadOneLdifFileINFO: ->
   LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oim_group_member_
   template.ldif
   Apr 5, 2011 4:59:01 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oim_groups_acl_
   template.ldif
   Apr 5, 2011 4:59:01 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oim_reserve_
   template.ldif
   *** Creation of Xel Sys Admin User ***
   Apr 5, 2011 4:59:01 AM oracle.ldap.util.LDIFLoader loadOneLdifFileINFO: ->
   LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oam_user_template.ldif
   Enter User Password for xelsysadm:
   Confirm User Password for xelsysadm:
   The tool has completed its operation. Details have been logged to
   /home/oracle/idmtools/oim.log
   ```

4. Check the log file for errors and warnings and correct them.

Creating Adapters in Oracle Virtual Directory

Oracle Identity Manager uses Oracle Virtual Directory to connect to external LDAP stores. You must create a user adapter and a change log adapter in Oracle Virtual Directory to enable Oracle Identity Manager to connect to the external LDAP store, such as Oracle Internet Directory. Follow these steps to create the adapters.

> **Note:** Creating adapters in Oracle Virtual Directory is not required if your implementation is Oracle Internet Directory only.

User Adapter

Create the user adapter on the Oracle Virtual Directory instances running on `OVDHOST1` and `OVDHOST2` individually.

To create the User Adapter in Oracle Virtual Directory using Oracle Directory Services Manager:

1. Open a browser and bring up the ODSM console at `http://admin.example.com/odsm`.

> **Note:** Although Oracle Directory Services Manager is not shown in Figure 5–2, it is required to manage Oracle Internet Directory and Oracle Virtual Directory. Oracle Directory Services Manager must exist in your environment.

2. Create connections to each of the Oracle Virtual Directory instances running on `OVDHOST1` and `OVDHOST2`, if they do not already exist
3. Connect to each Oracle Virtual Directory instance by using the appropriate connection entry.
4. On the Home page, click the **Adapter** tab.
5. Start the New Adapter Wizard by clicking **Create Adapter** at the top of the adapter window.
6. Create a new adapter using the New Adapter Wizard, with the following parameters:

> **Note:** If you created a User Adapter previously, skip the steps to create the Adapter and follow the steps to edit the Adapter.

Screen	Field	Value/Step
Type	**Adapter Type**	`LDAP`
	Adapter Name	`User Adapter`
	Adapter Template	`User_OID`
Connection	**Use DNS Setting**	No
	Host	`oid.example.com`
	Port	`389`
	Server Proxy Bind DN	`cn=oimadmin,cn=systemids,dc=example, dc=com`

Screen	Field	Value/Step
	Proxy Password	`oimadmin` password. This is the same password in "Extending the Directory Schema for Oracle Identity Manager".
Connection Test		Validate that the test succeeds.
Namespace	**Remote Base**	`dc=example,dc=com`
	Mapped Namespace	`dc=example,dc=com`
Summary		Verify that the summary is correct and then click **Finish**.

7. Edit the User Adapter as follows:
   a. Select the OIM User Adapter.
   b. Click the **Plug-ins** Tab.
   c. Click the **User Management** Plug-in, then click **Edit** in the plug-ins table. The plug-in editing window appears.
   d. In the Parameters table, update the parameter values as follows:

Parameter	value
directoryType	oid
pwdMaxFailure	10
oamEnabled	true

   e. Click **OK**.
   f. Click **Apply**.

Change Log Adapter

Create the change log adapter on the Oracle Virtual Directory instances running on `OVDHOST1` and `OVDHOST2` individually. Follow these steps to create the Change Log Adapter in Oracle Virtual Directory using Oracle Directory Services Manager.

1. Open a browser and bring up the ODSM console at `http://admin.example.com/odsm`.
2. Create connections to each of the Oracle Virtual Directory instances running on `OVDHOST1` and `OVDHOST2`, if they do not already exist.
3. Connect to an Oracle Virtual Directory instance by using the appropriate connection entry.
4. On the Home page click on the **Adapter** tab.
5. Start the New Adapter Wizard by clicking **Create Adapter** at the top of the adapter window.
6. Create a new adapter using the New Adapter Wizard and the following parameters:

Screen	Field	Value/Step
Type	**Adapter Type**	LDAP
	Adapter Name	OIM Change Log Adapter

Screen	Field	Value/Step
	Adapter Template	`Changelog_OID`
Connection	**Use DNS Setting**	No
	Host	`oid.example.com`
	Port	`389`
	Server Proxy Bind DN	`cn=oimadmin,cn=systemids,dc=example,dc=com`
	Proxy Password	`oimadmin` password. This is the same password provided in "Extending the Directory Schema for Oracle Identity Manager".
Connection Test		Validate that the test succeeds.
Naming Space	**Remote Base**	`cn=changelog`
Mapped Namespace		`cn=changelog`
Summary		Verify that the summary is correct then click **Finish**.

7. To edit the change adapter, follow these steps.

   a. Select the OIM Change Log Adapter.

   b. Click the **Plug-ins** tab.

   c. In the Deployed Plus-ins table, click the **changelog** plug-in, then click "**Edit** in the plug-ins table. The plug-in editing window appears.

   d. In the Parameters table, update the parameter values.

   e. Click **OK**.

   f. Click **Apply**.

   Edit the Change Log Adapter to either add or modify the properties so that they match the values shown in the following table. You must add the `mapObjectclass`, `modifierDNFilter`, `sizeLimit`, and `targetDNFilter` proprieties to the adapter.

Parameter	Value
directoryType	`oid`
mapAttribute	`targetGUID=orclguid`
requiredAttribute	`orclguid`
modifierDNFilter	`cn=oimadmin,cn=systemids,dc=example,dc=com`
sizeLimit	`1000`
targetDNFilter	`dc=example,dc=com` Search based from which reconciliation needs to happen. This value must be the same as the LDAP SearchDN that is specified during OIM installation.
mapUserState	`true`
oamEnabled	`true`

Stopping and Starting Oracle Internet Directory and Oracle Virtual Directory

Stop and Start:

- The Oracle Virtual Directory instances running on both `OVDHOST1` and `OVDHOST2`.
- The Oracle Internet Directory instances running on both `OIDHOST1` and `OIDHOST2`.

5.3.6.2 Updating the Coherence Configuration for the Coherence Cluster

Follow the steps below to update the Coherence configuration for the SOA managed servers:

1. Log into the Administration Console.
2. In the Domain Structure window, expand the **Environment** node.
3. Click **Servers**. The Summary of Servers page appears.
4. Click the name of the server (represented as a hyperlink) in the **Name** column of the table. The settings page for the selected server appears.
5. Click the Server Start tab.
6. Enter the following for WLS_SOA1 and WLS_SOA2 into the **Arguments** field.

   For WLS_SOA1, enter the following (on a single line, without a carriage return):

   ```
   -Dtangosol.coherence.wka1=soahost1vhn1 -Dtangosol.coherence.wka2=soahost2vhn1
   -Dtangosol.coherence.localhost=soahost1vhn1
   ```

   For WLS_SOA2, enter the following (on a single line, without a carriage return):

   ```
   -Dtangosol.coherence.wka1=soahost1vhn1 -Dtangosol.coherence.wka2=soahost2vhn1
   -Dtangosol.coherence.localhost=soahost2vhn1
   ```

7. Click **Save** and activate the changes.

   This change requires that you restart the SOA servers.

5.3.6.3 Running the Oracle Identity Management Configuration Wizard

You must configure the Oracle Identity Manager server instances before you can start the Oracle Identity Manager managed servers. You perform these configuration steps only once, for example, during the initial creation of the domain. The Oracle Identity Management Configuration Wizard loads the OIM metadata into the database and configures the instance.

Before running the Configuration Wizard, you must verify the following:

- The administration server is up and running.
- You have set coherence for `wls_soa1`.
- `wls_soa1` is running.
- The environment variables `DOMAIN_HOME` and `WL_HOME` are ***not*** set in the current shell.

The Oracle Identity Management Configuration Wizard is located under the Identity Management Oracle home. Type:

```
IAM_ORACLE_HOME/bin/config.sh
```

Proceed as follows:

1. On the Welcome screen, click **Next**

2. On the Components to Configure screen, select **OIM Server**. Select **OIM Remote Manager**, if required in your topology.

   Click **Next**.

3. On the Database screen, provide the following values:

   - **Connect String**: The connect string for the OIM database. For example:

     ```
     oimdbhost1-vip.example.com:1521:oimdb1^oimdbhost2-vip.example.com:1
     521:oimdb2@oim.example.com
     ```

   - **OIM Schema User Name**: `HA_OIM`
   - **OIM Schema password**: *`password`*
   - **MDS Schema User Name**: `HA_MDS`
   - **MDS Schema Password**: *`password`*

   Select **Next**.

4. On the WebLogic Administration Server screen, provide the following details for the WebLogic Administration Server:

   - **URL**: The URL to connect to the WebLogic Administration Server. For example: `t3://oimhost1.example.com:7001`
   - **UserName**: `weblogic`
   - **Password**: Password for the `weblogic` user

   Click **Next**.

5. On the OIM Server screen, enter the following values:

   - **OIM Administrator Password**: Password for the OIM Administrator. This is the password for the `xelsysadm` user, the same password you entered earlier for idmconfigtool.
   - **Confirm Password**: Confirm the password·
   - **OIM HTTP URL**: Proxy URL for the OIM Server. This is the URL for the Hardware load balancer that is front ending the OHS servers for OIM. For example: `http://oiminternal.example.com:80`.
   - **Key Store Password**: Key store password. The password must have an uppercase letter and a number. For example: `MyPassword1`

   Click **Next**.

6. On the LDAP Sync and OAM screen, select **Configure BI Publisher** and provide the **BI Publisher URL**, if required in your environment. Enter the URL to connect to the BI Publisher in your environment.

   Select **Enable LDAP Sync**.

   Notes:

   - Do not select **Enable Identity Administration Integration with OAM**.
   - BI Publisher is not a part of the *IDMDomain*.

   Click **Next**.

7. On the LDAP Server screen, provide the following LDAP server details:

   - **Directory Server Type:** The directory server type. Select OID, ACTIVE_DIRECTORY, IPLANET, or OVD. The default is OID.
   - **Directory Server ID**: The directory server ID.
   - **Server URL**: The URL to access the LDAP server. For example: `ldap://ovd.example.com:389` if you use the Oracle Virtual Directory Server, `ldap://oid.example.com:389` if you use Oracle Internet Directory.
   - **Server User**: The username to connect to the server. For example: `cn=orcladmin`.
   - **Server Password**: The password to connect to the LDAP server.
   - **Server SearchDN**: The Search DN. For example: `dc=example,dc=com`.

   Click **Next**.

8. On the LDAP Server Continued screen, enter the following LDAP server details:

   - **LDAP Role Container**: The DN for the Role Container. This is the container where the OIM roles are stored. For example: `cn=Groups,dc=example,dc=com` .
   - **LDAP User Container**: The DN for the User Container. This is the container where the OIM users are stored. For example: `cn=Users,dc=example,dc=com`.
   - **User Reservation Container**: The DN for the User Reservation Container.

   > **Note:** Use the same container DN Values that `idmconfigtool` creates during the procedure "Creating Users and Groups for Oracle Identity Manager."

   Click **Next**.

9. On the Remote Manager screen, provide the following values:

   > **Note:** This screen appears only if you selected the Remote Manager utility in step 2.

   - **Service Name**: `HA_RManager`
   - **RMI Registry Por**t: `12345`
   - **Listen Port (SSL)**: `12346`

10. On the Configuration Summary screen, verify the summary information.

    Click **Configure** to configure the Oracle Identity Manager instance.

11. On the Configuration Progress screen, once the configuration completes successfully, click **Next**.

12. On the Configuration Complete screen, view the details of the Oracle Identity Manager Instance configured.

    Click **Finish** to exit the Configuration Assistant.

13. Stop the WebLogic Administration Server.

14. Start the WebLogic Administration Server.

5.3.7 Post-Configuration Steps for the Managed Servers

This section describes post-configuration procedure for the managed servers.

5.3.7.1 Start the WLS_SOA1 and WLS_OIM1 Managed Servers on OIMHOST1

To start the WLS_SOA1 and WLS_OIM1 managed servers on OIMHOST1:

1. Stop the WebLogic Administration Server on OIMHOST1. Use the WebLogic Administration Console to do this
2. Start the WebLogic Administration Server on OIMHOST1 using the `startWebLogic.sh` script under the *DOMAIN_HOME*`/bin` directory. For example:

   ```
   /u01/app/oracle/admin/OIM/bin/startWebLogic.sh > /tmp/admin.out 2>1&
   ```

3. Validate that the WebLogic Administration Server started up successfully by bringing up the WebLogic Administration Console.
4. Restart the WLS_SOA1 managed server using the WebLogic Administration Console.
5. Start the WLS_OIM1 managed server using the WebLogic Administration Console.

5.3.8 Validate the Oracle Identity Manager Instance on OIMHOST1

Validate the Oracle Identity Manager Server instance on OIMHOST1 by bringing up the Oracle Identity Manager Console using a web browser.

The URL for the Oracle Identity Manager Console is:

```
http://identityvhn1.example.com:14000/identity
```

Log in using the `xelsysadm` password.

5.3.9 Propagating Oracle Identity Manager to OIMHOST2

Once the configuration has succeeded on OIMHOST1, the configuration can be propagated to OIMHOST2. This is done by packing the domain on OIMHOST1 and then unpacking it on OIMHOST2.

Follow these steps to pack the domain on OIMHOST1 and unpack it on OIMHOST2:

1. On OIMHOST1, invoke the `pack` utility in the *ORACLE_HOME*`/oracle_common/common/bin` directory:

   ```
   pack.sh -domain=MW_HOME/user_projects/domains/OIM_Domain -
   template =/u01/app/oracle/admin/templates/oim_domain.jar -
   template_name="OIM Domain" -managed=true
   ```

2. The previous step created the `oim_domain.jar` file in the following directory:

   ```
   /u01/app/oracle/admin/templates
   ```

   Copy the `oim_domain.jar` file from OIMHOST1 to a temporary directory on OIMHOST2.

3. On OIMHOST2, invoke the `unpack` utility in the *MW_HOME*`/oracle_common/common/bin` directory and specify the location of the `oim_domain.jar` file in its temporary directory:

   ```
   unpack.sh -domain=MW_HOME/user_projects/domains/OIM_Domain -
   template=/tmp/oim_domain.jar
   ```

5.3.10 Post-Installation Steps on OIMHOST2

This section describes the post-installation steps to perform on OIMHOST2. It includes these sections:

- Section 5.3.10.1, "Update Node Manager on OIMHOST2"
- Section 5.3.10.2, "Start Node Manager on OIMHOST2"
- Section 5.3.10.3, "Start the WLS_SOA2 and WLS_OIM2 Managed Servers on OIMHOST2"

5.3.10.1 Update Node Manager on OIMHOST2

Before managed servers can be started via the WebLogic Administration Console, Node Manager requires that the `StartScriptEnabled` property be set to true.

To do this, run the `setNMProps.sh` script located under the following directory:

```
MW_HOME/oracle_common/common/bin
```

5.3.10.2 Start Node Manager on OIMHOST2

Start the Node Manager on OIMHOST2 using the startNodeManager.sh script located under the following directory:

```
MW_HOME/wlserver_10.3/server/bin
```

5.3.10.3 Start the WLS_SOA2 and WLS_OIM2 Managed Servers on OIMHOST2

Follow these steps to start the WLS_SOA2 and WLS_OIM2 managed servers on OIMHOST2:

1. Validate that the WebLogic Administration Server started up successfully by bringing up the WebLogic Administration Console.
2. Start the WLS_SOA2 managed server using the WebLogic Administration Console.
3. Start the WLS_OIM2 managed server using the WebLogic Administration Console. The WLS_OIM2 managed server must be started after the WLS_SOA2 managed server is started.

5.3.11 Validate the Oracle Identity Manager Instance on OIMHOST2

Validate the Oracle Identity Manager Server instance on OIMHOST2 by bringing up the Oracle Identity Manager Console using a web browser.

The URL for the Oracle Identity Manager Console is:

```
http://identityvhn2.example.com:14000/oim
```

Log in using the `xelsysadm` password.

5.3.12 Updating SOA Server Default Composite

In an integrated environment, Oracle Identity Manager is front ended by OHS. All SOA server default composites must be updated. See Updating SOA Server Default

Composite in the *Oracle Fusion Middleware Integration Guide for Oracle Identity Management Suite* for the procedure to update SOA server default composites.

5.3.13 Configuring Oracle Internet Directory using the LDAP Configuration Post-setup Script

> **Note:** This section is required only for LDAPSync-enabled Oracle Identity Manager installations and for Oracle Identity Manager installations that integrate with Oracle Access Management.
>
> If you do not plan to enable the LDAP-Sync option or to integrate with Oracle Access Management, you can skip this section.

In the current release, the `LDAPConfigPostSetup` script enables all the LDAPSync-related incremental Reconciliation Scheduler jobs, which are disabled by default. The LDAP configuration post-setup script is located under the *`IAM_ORACLE_HOME`*`/server/ldap_config_util` directory. To run the script, follow these steps:

1. Edit the `ldapconfig.props` file located under the *`IAM_ORACLE_HOME`*`/server/ldap_config_util` directory and provide the following values:

Parameter	Value	Description
`OIMProviderURL`	`t3://OIMHOST1VHN.example.com:14000,OIMHOST2VHN.example.com:14000`	List of Oracle Identity Manager managed servers.
`LDAPURL`	Specify the URL for the Oracle Virtual Directory instance, for example: `ldap://idstore.example.com:389`	Identity Store URL. Only required if IDStore is accessed using Oracle Virtual Directory.
`LDAPAdminUserName`	`cn=oimadmin,cn=systemids,dc=example,dc=com`	Name of user to connect to Identity Store. Only required if your Identity Store is in Oracle Virtual Directory. This user should **not** be located in `cn=Users,dc=example,dc=com`.
`LIBOVD_PATH_PARAM`	*`MSERVER_HOME`*`/config/fmwconfig/ovd/oim`	Required unless you access your identity store using Oracle Virtual Directory.

> **Note:** `usercontainerName`, `rolecontainername`, and `reservationcontainername` are not used in this step.

2. Save the file.
3. Set the *`JAVA_HOME`*, *`WL_HOME`*, *`APP_SERVER`*, *`OIM_ORACLE_HOME`*, and *`DOMAIN_HOME`* environment variables, where:
   - `JAVA_HOME` is set to *`MW_HOME`*`/jrockit_`*version*
   - `WL_HOME` is set to *`MW_HOME`*`/wlserver_10.3`

- `APP_SERVER` is set to `weblogic`
- `OIM_ORACLE_HOME` is set to *`IAM_ORACLE_HOME`*
- `DOMAIN_HOME` is set to *`MSERVER_HOME`*

4. Run LDAPConfigPostSetup.sh. The script prompts for the LDAP administrator password and the Oracle Identity Manager administrator password. For example:

```
IAM_ORACLE_HOME/server/ldap_config_util/LDAPConfigPostSetup.sh path_to_
property_file
```

For example:

```
IAM_ORACLE_HOME/server/ldap_config_util/LDAPConfigPostSetup.sh IAM_ORACLE_
HOME/server/ldap_config_util
```

5.3.14 Configuring Server Migration for the OIM and SOA Managed Servers

For this high availability topology, Oracle recommends that you configure server migration for the WLS_OIM1, WLS_SOA1, WLS_OIM2, and WLS_SOA2 managed servers. See Section 3.9, "Whole Server Migration" for information on the benefits of using Whole Server Migration and why Oracle recommends it.

- The WLS_OIM1 and WLS_SOA1 managed servers on OIMHOST1 are configured to restart automatically on OIMHOST2 if a failure occurs on OIMHOST1.
- The WLS_OIM2 and WLS_SOA2 managed servers on OIMHOST2 are configured to restart automatically on OIMHOST1 if a failure occur on OIMHOST2.

In this configuration, the WLS_OIM1, WLS_SOA1, WLS_OIM2 and WLS_SOA2 servers listen on specific floating IPs that WebLogic Server Migration fails over.

The following steps enable server migration for the WLS_OIM1, WLS_SOA1, WLS_OIM2, and WLS_SOA2 managed servers, which in turn enables a managed server to fail over to another node if a server or process failure occurs:

- Step 1: Setting Up a User and Tablespace for the Server Migration Leasing Table
- Step 2: Creating a Multi Data Source Using the Oracle WebLogic Administration Console
- Step 3: Editing Node Manager's Properties File
- Step 4: Setting Environment and Superuser Privileges for the wlsifconfig.sh Script
- Step 5: Configuring Server Migration Targets
- Step 6: Testing the Server Migration

5.3.14.1 Setting Up a User and Tablespace for the Server Migration Leasing Table

The first step to set up a user and tablespace for the server migration leasing table:

> **Note:** If other servers in the same domain are already configured with server migration, you can use the same tablespace and data sources. In this case, you don't need to recreate the data sources and multi data source for database leasing, however, you must retarget them to the clusters you're configuring for server migration.

1. Create a tablespace named `leasing`. For example, log on to SQL*Plus as the sysdba user and run the following command:

```
SQL> create tablespace leasing logging datafile 'DB_
HOME/oradata/orcl/leasing.dbf' size 32m autoextend on next 32m maxsize 2048m
extent management local;
```

2. Create a user named `leasing` and assign to it the leasing tablespace:

```
SQL> create user leasing identified by welcome1;
SQL> grant create table to leasing;
SQL> grant create session to leasing;
SQL> alter user leasing default tablespace leasing;
SQL> alter user leasing quota unlimited on LEASING;
```

3. Create the leasing table using the leasing.ddl script:

   a. Copy the leasing.ddl file located in either the *WL_HOME*/server/db/oracle/817 or the *WL_HOME*/server/db/oracle/920 directory to your database node.

   b. Connect to the database as the **leasing** user.

   c. Run the leasing.ddl script in SQL*Plus:

   ```
   SQL> @Copy_Location/leasing.ddl;
   ```

5.3.14.2 Creating a Multi Data Source Using the Oracle WebLogic Administration Console

You create a data source to each of the Oracle RAC database instances during the process of setting up the multi data source, both for these data sources and the global leasing multi data source. When you create a data source:

- Ensure that this is a non-XA data source.
- The names of the multi data sources are in the format of *<MultiDS>-rac0*, *<MultiDS>-rac1*, and so on.
- Use Oracle's Driver (Thin) Version 9.0.1, 9.2.0, 10, 11.
- Data sources do not require support for global transactions. Therefore, do *not* use any type of distributed transaction emulation/participation algorithm for the data source (do not choose the **Supports Global Transactions** option, or the **Logging Last Resource**, **Emulate Two-Phase Commit**, or **One-Phase Commit** options of the **Supports Global Transactions** option), and specify a service name for your database.
- Target these data sources to the cluster(s).
- Make sure the data source's connection pool initial capacity is set to 0 (zero). To do this, select **Services**, **JDBC** and then **Datasources**. In the Datasources screen, click the **Datasource Name**, click the **Connection Pool** tab, and enter **0** (zero) in the **Initial Capacity** field.

Creating a Multi Data Source

To create a multi data source:

1. Log into the Administration Console using administrator credentials.
2. In the Domain Structure window, expand the **Services** node then expand the **DataSource** node.
3. Click **Lock and Edit** in the Change Center.
4. Click **New** then click **Multi Data Sources**.

5. Enter `leasing` as the name.
6. Enter `jdbc/leasing` as the JNDI name.
7. Select **Failover as algorithm** (default). Click **Next**.
8. Select the target cluster(s). Click **Next**.
9. Select **non-XA driver** (the default). Click **Next**.
10. Click **Create New Data Source**.
11. Enter `leasing-rac0` as the name. Enter `jdbc/leasing-rac0` as the JNDI name. Enter `oracle` as the database type. For the driver type, select Oracle Driver (Thin) for RAC server-Instance connection Version 10,11.

> **Note:** When you create multi data sources for the leasing table, enter names in the format <*MultiDS*>-rac0, <*MultiDS*>-rac1, and so on.

12. Click **Next**.
13. Deselect **Supports Global Transactions**. Click **Next**.
14. Enter the service name, database name (the RAC Node instance name such as `racdb1`, `racdb2`), host port, and password for your leasing schema. Click **Next**.
15. Click **Test Configuration** and verify that the connection works. Click **Next**.

> **Note:** Do not click **Finish** until you are done creating all of the data sources that you want to make. If you do not see the data sources that you created, this indicates that you clicked **Finish** too soon. When you click **Finish**, the Administration Console does not show the screen in which you target the data sources.

16. Target the data source to the cluster(s).
17. Select the data source and add it to the right screen.
18. Click **Create a New Data Source** for the second instance of your Oracle RAC database, target it to the cluster(s), repeating the steps for the second instance of your Oracle RAC database.
19. Add the second data source to your multi data source.
20. Save and click **Activate Changes**.

5.3.14.3 Editing Node Manager's Properties File

You must edit the `nodemanager.properties` file to add the following properties for each node where you configure server migration:

```
Interface=eth0
NetMask=255.255.255.0
UseMACBroadcast=true
```

- `Interface`: Specifies the interface name for the floating IP (such as `eth0`).

> **Note:** Do not specify the sub interface, such as `eth0:1` or `eth0:2`. This interface is to be used without the `:0`, or `:1`. The Node Manager's scripts traverse the different `:X` enabled IPs to determine which to add or remove. For example, the valid values in Linux environments are `eth0`, `eth1`, or, `eth2`, `eth3`, `eth`*n*, depending on the number of interfaces configured.

- `NetMask`: Net mask for the interface for the floating IP. The net mask should the same as the net mask on the interface; `255.255.255.0` is an example. The actual value depends on your network.
- `UseMACBroadcast`: Specifies whether or not to use a node's MAC address when sending ARP packets, that is, whether or not to use the `-b` flag in the `arping` command.

Verify in Node Manager's output (shell where Node Manager starts) that these properties are being used or problems may arise during migration. You should see an entry similar to the following in Node Manager's output:

```
...
StateCheckInterval=500
Interface=eth0
NetMask=255.255.255.0
...
```

> **Note:** The following steps are not required if the server properties (start properties) are properly set and Node Manager can start the servers remotely.

1. Set the following property in the `nodemanager.properties` file:
   - `StartScriptEnabled`: Set this property to `true` to enable Node Manager to start the managed servers.
2. Start Node Manager on OIMHOST1 and OIMHOST2 by running the `startNodeManager.sh` script in the *WL_HOME*/server/bin directory.

> **Note:** When running Node Manager from a shared storage installation, multiple nodes start using the same `nodemanager.properties` file. However, each node may require different NetMask or Interface properties. In this case, specify individual parameters on a per-node basis using environment variables. For example, to use a different interface (eth3) in HOST*n*, use the Interface environment variable as follows: `HOSTn> export JAVA_OPTIONS=-DInterface=eth3` and start Node Manager after the variable is set in the shell.

5.3.14.4 Setting Environment and Superuser Privileges for the wlsifconfig.sh Script

To set environment and superuser privileges for the `wlsifconfig.sh` script for each node where you configure server migration:

1. Ensure that your PATH environment variable includes these files:

Table 5–3 Files Required for the PATH Environment Variable

File	Located in this directory
wlsifconfig.sh	*DOMAIN_HOME*/bin/server_migration
wlscontrol.sh	*WL_HOME*/common/bin
nodemanager.domains	*WL_HOME*/common

2. Grant sudo configuration for the `wlsifconfig.sh` script.
   - Configure sudo to work without a password prompt.
   - For security reasons, Oracle recommends restricting to the subset of commands required to run the `wlsifconfig.sh` script. For example, perform the following steps to set the environment and superuser privileges for the `wlsifconfig.sh` script:
   - Grant sudo privilege to the WebLogic user (`oracle`) with no password restriction, and grant execute privilege on the `/sbin/ifconfig` and `/sbin/arping binaries`.
   - Ensure that the script is executable by the WebLogic user. The following is an example of an entry inside `/etc/sudoers` granting sudo execution privilege for `oracle` and also over `ifconfig` and `arping`:

     ```
     oracle ALL=NOPASSWD: /sbin/ifconfig,/sbin/arping
     ```

> **Note:** Ask the system administrator for the sudo and system rights as appropriate to this step.

5.3.14.5 Configuring Server Migration Targets

You first assign all the available nodes for the cluster's members and then specify candidate machines (in order of preference) for each server that is configured with server migration. To configure cluster migration in a migration in a cluster:

1. Log into the Administration Console.
2. In the Domain Structure window, expand **Environment** and select **Clusters**.
3. Click the cluster you want to configure migration for in the Name column.
4. Click the **Migration** tab.
5. Click **Lock and Edit**.
6. In the **Available** field, select the machine to which to enable migration and click the right arrow.
7. Select the data source to use for automatic migration. In this case, select the leasing data source.
8. Click **Save**.
9. Click **Activate Changes**.
10. Set the candidate machines for server migration. You must perform this task for all managed servers as follows:
    a. In the Domain Structure window of the Administration Console, expand **Environment** and select **Servers**.

> **Tip:** Click **Customize this table** in the Summary of Servers page and move Current Machine from the Available window to the Chosen window to view the machine that the server runs on. This will be different from the configuration if the server migrates automatically.

b. Select the server that you want to configure migration for.

c. Click the **Migration** tab.

d. In the **Available** field, located in the Migration Configuration section, select the machines you want to enable migration to and click the right arrow.

e. Select **Automatic Server Migration Enabled**. This enables Node Manager to start a failed server on the target node automatically.

f. Click **Save** then Click **Activate Changes**.

g. Repeat the steps above for any additional managed servers.

h. Restart the administration server, Node Managers, and the servers for which server migration has been configured.

5.3.14.6 Testing the Server Migration

To verify that server migration works properly:

From OIMHOST1:

1. Stop the WLS_OIM1 managed server. To do this, run this command:

   ```
   OIMHOST1> kill -9 pid
   ```

   where *pid* specifies the process ID of the managed server. You can identify the pid in the node by running this command:

   ```
   OIMHOST1> ps -ef | grep WLS_OIM1
   ```

2. Watch the Node Manager console. You should see a message indicating that WLS_OIM1's floating IP has been disabled.

3. Wait for Node Manager to try a second restart of WLS_OIM1. It waits for a fence period of 30 seconds before trying this restart.

4. Once Node Manager restarts the server, stop it again. Node Manager should now log a message indicating that the server will not be restarted again locally.

From OIMHOST2:

1. Watch the local Node Manager console. After 30 seconds since the last try to restart WLS_OIM1 on OIMHOST1, Node Manager on OIMHOST2 should prompt that the floating IP for WLS_OIM1 is being brought up and that the server is being restarted in this node.

2. Access the soa-infra console in the same IP.

Follow the steps above to test server migration for the WLS_OIM2, WLS_SOA1, and WLS_SOA2 managed servers.

Table 5–4 shows the managed servers and the hosts they migrate to in case of a failure.

Table 5–4 WLS_OIM1, WLS_OIM2, WLS_SOA1, WLS_SOA2 Server Migration

Managed Server	Migrated From	Migrated To
WLS_OIM1	OIMHOST1	OIMHOST2
WLS_OIM2	OIMHOST2	OIMHOST1
WLS_SOA1	OIMHOST1	OIMHOST2
WLS_SOA2	OIMHOST2	OIMHOST1

Verification From the Administration Console

Migration can also be verified in the Administration Console:

1. Log into the Administration Console at http://oimhost1.example.com:7001/console using administrator credentials.
2. Click **Domain** on the left console.
3. Click the **Monitoring** tab and then the **Migration** sub tab.

   The Migration Status table provides information on the status of the migration.

> **Note:** After a server is migrated, to fail it back to its original node/machine, stop the managed server from the Oracle WebLogic Administration Console and then start it again. The appropriate Node Manager will start the managed server on the machine to which it was originally assigned.

5.3.15 Configuring a Shared JMS Persistence Store

Configure the location for all of the persistence stores as a shared directory that is visible from both OIMHOST1 and OIMHOST2. As JMS messages are persisted in the file system for each server's local file system, shared storage is necessary for the JMS persistence store for WebLogic server migration. Without shared storage, a migrated server cannot access pending JMS messages.

> **Note:** See Considerations for Using File Stores on NFS for information about JMS messages and transaction logs, as well as releasing locks on file stores.

To change all persistent stores to use a shared base directory:

1. Log into the Administration Console at http://oimhost1.example.com:7001/console using the administrator credentials.
2. In the Domain Structure window, expand the **Services** node and then click the **Persistence Stores** node. The Summary of Persistence Stores page is displayed.
3. Select the hyperlink(s) for the persistence store from the **Name** column of the table: BPM, SOA, OIM, and UMS. The Settings page for the persistence store opens.
4. On the Configuration tab, in the **Directory** field, enter the location of a persistent storage solution (such as NAS or SAN) that is available to other servers in the cluster. Specifying this location enables pending JMS messages to be sent.
5. The location should follow this directory structure:

- For the WLS_SOA1 and WLS_SOA2 servers, use a directory structure similar to:

  ```
  ORACLE_BASE/admin/domainName/soaClusterName/jms
  ```

- For the WLS_OIM1 and WLS_OIM2 servers, use a directory structure similar to:

  ```
  ORACLE_BASE/admin/domainName/oimClusterName/jms
  ```

> **Note:** The WLS_OIM1 and WLS_OIM2 servers must be able to access this directory.
>
> The WLS_SOA1 and WLS_SOA2 servers must be able to access this directory.
>
> This directory must exist *before* you restart the server.

6. Click **Save and Activate**.
7. Restart the servers to make the change in the persistent stores take effect.

5.3.16 Configuring a Default Persistence Store for Transaction Recovery

Each Oracle WebLogic Managed Server has a transaction log that stores information about inflight transactions that are coordinated by the server that may not have been completed. The WebLogic Server uses this transaction log for recovery from system fails or network failures. To leverage the migration capability of the Transaction Recovery Service for the servers within a cluster, store the transaction log in a location accessible to all the servers in the cluster. Without shared storage, other servers in the cluster cannot do a transaction recovery in the case of a server failure, so the operation may need to be retried.

> **Note:** Preferably, this location should be a dual-ported SCSI disk or on a Storage Area Network (SAN).

Perform these steps to set the location for the default persistence stores for the Oracle Identity Manager and SOA Servers:

1. Log into the Administration Console at http://oimhost1.*example*.com:7001/console using administrator credentials.
2. In the Domain Structure window, expand the **Environment** node and then click the **Servers** node. The Summary of Servers page is displayed.
3. Select the name of the server (represented as a hyperlink) in the **Name** column of the table. The Settings page for the selected server is displayed, and it defaults to the Configuration tab.
4. Click the Services tab.
5. In the Default Store section of the page, enter the path to the folder where the default persistent stores will store their data files. The directory structure of the path should be:
   - For the WLS_SOA1 and WLS_SOA2 servers, use a directory structure similar to:

ORACLE_BASE/admin/*domainName*/*soaClusterName*/tlogs

- For the WLS_OIM1 and WLS_OIM2 servers, use a directory structure similar to:

  ORACLE_BASE/admin/*domainName*/*oimClusterName*/tlogs

6. Click **Save**.

> **Note:** To enable migration of the Transaction Recovery Service, specify a location on a persistent storage solution that is available to the managed servers in the cluster. WLS_SOA1, WLS_SOA2, WLS_OIM1, and WLS_OIM2 must be able to access this directory.

5.3.17 Install Oracle HTTP Server on WEBHOST1 and WEBHOST2

Install Oracle HTTP Server on WEBHOST1 and WEBHOST2.

5.3.18 Configuring Oracle Identity Manager to Work with the Web Tier

This section describes how to configure Oracle Identity Manager to work with the Oracle Web Tier.

5.3.18.1 Prerequisites

Verify that the following tasks have been performed:

1. Oracle Web Tier has been installed on WEBHOST1 and WEBHOST2.
2. Oracle Identity Manager has been installed and configured on OIMHOST1 and OIMHOST2.
3. The load balancer has been configured with a virtual hostname (`sso.example.com`) pointing to the web servers on WEBHOST1 and WEBHOST2.
4. The load balancer has been configured with a virtual hostname (`oiminternal.example.com`) pointing to web servers WEBHOST1 and WEBHOST2.

5.3.18.2 Configuring Oracle HTTP Servers to Front End the OIM and SOA Managed Servers

1. On each of the web servers on `WEBHOST1` and `WEBHOST2`, create a file called `oim.conf` in the directory *`ORACLE_INSTANCE`*`/config/OHS/component/moduleconf`. This file must contain the following information:

```
# oim admin console(idmshell based)
   <Location /admin>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
   </Location>

# oim self and advanced admin webapp consoles(canonic webapp)

  <Location /oim>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
   </Location>

  <Location /identity>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
    </Location>

  <Location /sysadmin>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
    </Location>

# SOA Callback webservice for SOD - Provide the SOA Managed Server Ports
  <Location /sodcheck>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:8001,oimvhn2.example.com:8001
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
   </Location>

# Callback webservice for SOA. SOA calls this when a request is
approved/rejected
# Provide the SOA Managed Server Port
  <Location /workflowservice>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>

# xlWebApp - Legacy 9.x webapp (struts based)
   <Location /xlWebApp>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>

# Nexaweb WebApp - used for workflow designer and DM
  <Location /Nexaweb>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>

# used for FA Callback service.
  <Location /callbackResponseService>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>

# spml xsd profile
  <Location /spml-xsd>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>

  <Location /HTTPClnt>
    SetHandler weblogic-handler
    WLCookieName    oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>


  <Location /reqsvc>
    SetHandler weblogic-handler
    WLCookieName oimjsessionid
    WebLogicCluster oimvhn1.example.com:8001,oimvhn2.example.com:8001
    oimvh1.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>


  <Location /integration>
    SetHandler weblogic-handler
    WLCookieName oimjsessionid
    WebLogicCluster oimvhn1.example.com:8001,oimvhn2.example.com:8001
    oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>


  <Location /provisioning-callback>
```

```
    SetHandler weblogic-handler
    WLCookieName oimjsessionid
    WebLogicCluster oimvhn1.example.com:14000,oimvhn2.example.com:14000
    WLLogFile "${ORACLE_INSTANCE}/diagnostics/logs/mod_wl/oim_component.log"
    WLProxySSL ON
    WLProxySSLPassThrough ON
  </Location>
```

2. Create a file called `virtual_hosts.conf` in *ORACLE_INSTANCE*`/config/OHS/component/moduleconf`. The file must contain the following information:

```
NameVirtualHost *:7777
<VirtualHost *:7777>

  ServerName http://sso.example.com:7777
  RewriteEngine On
  RewriteOptions inherit
  UseCanonicalName On
  </VirtualHost>

<VirtualHost *:7777>
  ServerName http://oiminternal.example.com:80
  RewriteEngine On
  RewriteOptions inherit
  UseCanonicalName On
</VirtualHost>
```

3. Save the file on both `WEBHOST1` and `WEBHOST2`.

4. Stop and start the Oracle HTTP Server instances on both `WEBHOST1` and `WEBHOST2`.

5.3.19 Validate the Oracle HTTP Server Configuration

To validate that Oracle HTTP Server is configured properly, follow these steps:

1. In a web browser, enter the following URL for the Oracle Identity Manager Console:

   ```
   http://sso.example.com:7777/identity
   ```

   The Oracle Identity Manager Console login page should display.

2. Log into the Oracle Identity Manager Console using the credentials for the `xelsysadm` user.

5.3.20 Oracle Identity Manager Failover and Expected Behavior

In a high availability environment, WebLogic Node Manager is configured to monitor the Oracle WebLogic Servers. In case of failure, Node Manager restarts the WebLogic Server.

In a high availability environment, a hardware load balancer is used to load balance requests between the multiple Oracle Identity Manager instances. If one of the Oracle Identity Manager instances fails, the load balancer detects the failure and routes requests to the surviving instances.

In a high availability environment, the state and configuration information is stored in a database that is shared by all the members of the cluster.

The surviving Oracle Identity Manager instances will continue to seamlessly process any unfinished transactions started on the failed instance, since the state information is in the shared database and is available to all the members in the cluster.

When an Oracle Identity Manager instance fails, its database and LDAP connections are released. The surviving instances in the active-active deployment make their own connections to continue processing unfinished transactions on the failed instance.

Be aware of the following when you deploy Oracle Identity Manager in a high availability configuration:

- Oracle Identity Manager can be deployed on an Oracle RAC database, but Oracle RAC failover is not transparent for Oracle Identity Manager in this release. If an Oracle RAC failover occurs, end users may have to resubmit their requests.
- Oracle Identity Manager always requires that at least one of the nodes in the SOA cluster be available. If the SOA cluster is not available, end user requests will fail. Oracle Identity Manager does not retry for a failed SOA call. Therefore, the end user must retry when a SOA call fails.

5.3.21 Troubleshooting Oracle Identity Manager High Availability

If you are creating a user in Oracle Identity Manager (by logging into Oracle Identity Manager, clicking the Administration tab, clicking the **Create User** link, entering the required information in the fields, and clicking **Save**) in an active-active Oracle Identity Manager configuration, and the Oracle Identity Manager server that is handling the request fails, you may see a ResourceConnectionValidationxception in the Oracle Identity Manager log file, similar to:

```
[2010-06-14T15:14:48.738-07:00] [oim_server2] [ERROR] [] [XELLERATE.SERVER]
[tid: [ACTIVE].ExecuteThread: '0' for queue: 'weblogic.kernel.Default
(self-tuning)'] [userId: xelsysadm] [ecid:
004YGJGmYrtEkJV6u3M6UH00073A0005EI,0:1] [APP: oim#11.1.1.3.0] [dcid:
12eb0f9c6e8796f4:-785b18b3:12938857792:-7ffd-0000000000000037] [URI:
/admin/faces/pages/Admin.jspx] Class/Method:
PooledResourceConnection/heartbeat encounter some problems: Operation timed
out[[
com.oracle.oim.gcp.exceptions.ResourceConnectionValidationxception: Operation
timed out
        at
oracle.iam.ldapsync.impl.repository.LDAPConnection.heartbeat(LDAPConnection.ja
va:162)
        at
com.oracle.oim.gcp.ucp.PooledResourceConnection.heartbeat(PooledResourceConnec
tion.java:52)
         .
         .
         .
```

Although this exception is received, the user is created fine.

5.3.22 Scaling Up and Scaling Out the Oracle Identity Manager Topology

You can scale out or scale up the Oracle Identity Manager high availability topology. When you scale up the topology, you add new managed servers to nodes that are already running one or more managed servers. When you scale out the topology, you add new managed servers to new nodes.

5.3.22.1 Scaling Up Oracle Identity Manager

In this case, you already have a node that runs a managed server configured with SOA components. The node contains a Middleware home, an Oracle HOME (SOA), and a domain directory for existing managed servers.

You can use the existing installations (Middleware home and domain directories) for creating new WLS_OIM and WLS_SOA servers. You do not need to install the OIM and SOA binaries in a new location or run pack and unpack.

To scale up the topology:

1. Using the Administration Console, clone either WLS_OIM1 or WLS_SOA1 into a new managed server. The source managed server to clone should be one that already exists on the node where you want to run the new managed server.

   To clone a managed server:

   a. Select **Environment -> Servers** from the Administration Console.

   b. Select the managed server that you want to clone (WLS_OIM1 or WLS_SOA1).

   c. Select **Clone**.

   d. Name the new managed server WLS_OIM`n` or WLS_SOA`n`, where `n` is a number to identity the new managed server.

   The rest of the steps assume that you are adding a new server to OIMHOST1, which is already running WLS_SOA1 and WLS_OIM1.

2. For the listen address, assign the hostname or IP to use for this new managed server. If you are planning to use server migration as recommended for this server, this should be the VIP (also called a floating IP) to enable it to move to another node. The VIP should be different from the one used by the managed server that is already running.

3. Create JMS Servers for SOA, OIM, BPM, UMS, JRFWSAsync, and PS6SOA on the new managed server.

   a. Use the Administration Console to create a new persistent store for the new SOAJMSServer and name it, for example, `SOAJMSFileStore_n`. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms
   ```

   Note: This directory must exist before the managed server is started or the start operation fails.

   b. Create a new JMS Server for SOA, for example, SOAJMSServer_n. Use the SOAJMSFileStore_n for this JMSServer. Target the SOAJMSServer_n Server to the recently created Managed Server (WLS_SOAn).

   c. Create a new persistence store for the new UMSJMSServer, for example, UMSJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms/UMSJMSFileStore_n
   ```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new UMS JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

d. Create a new JMS Server for UMS, for example, UMSJMSServer_n. Use the UMSJMSFileStore_n for this JMSServer. Target the UMSJMSServer_n Server to the recently created managed server (WLS_SOAn).

e. Create a new persistence store for the new OIMJMSServer, for example, OIMJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms/OIMJMSFileStore_n
   ```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new OIM JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

f. Create a new JMS Server for OIM, for example, OIMJMSServer_n. Use the OIMJMSFileStore_n for this JMSServer. Target the OIMJMSServer_n Server to the recently created Managed Server (WLS_OIMn).

g. Create a new persistence store for the new JRFWSAsyncJMSServer, for example, JRFWSAsyncJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms/JRFWSAsyncJMSFileStore_n
   ```

h. Create a new JMS Server for JRFWSAsync, for example, JRFWSAsyncJMSServer_n. Use the JRFWSAsyncJMSFileStore_n for this JMSServer. Target the JRFWSAsyncJMSServer_n Server to the recently created Managed Server (WLS_OIMn).

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new JRFWSAsync JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

i. Create a new persistence store for the new PS6JMSServer, for example, PS6JMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms/PS6JMSFileStore_n
   ```

j. Create a new JMS Server for PS6, for example, PS6JMSServer_n. Use the PS6JMSFileStore_n for this JMSServer. Target the PS6JMSServer_n Server to the recently created Managed Server (WLS_OIMn).

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new PS6 JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

k. Update the SubDeployment targets for the SOA JMS Module to include the recently created SOA JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **SOAJMSModule** (represented as a hyperlink in the **Names** column of the table). The Settings page for SOAJMSModule appears. Click the **SubDeployments** tab. The subdeployment module for SOAJMS appears.

Note: This subdeployment module name is a random name in the form of SOAJMSServerXXXXXX resulting from the Config Wizard JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2).

l. Click the **SOAJMSServerXXXXXX** subdeployment. Add the new JMS Server for SOA called SOAJMSServer_n to this subdeployment. Click **Save**.

m. Update the SubDeployment targets for the UMSJMSSystemResource to include the recently created UMS JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **UMSJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for UMSJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for UMSJMS appears.

Note: This subdeployment module name is a random name in the form of UMSJMSServerXXXXXX resulting from the Config Wizard JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2).

n. Click the **UMSJMSServerXXXXXX** subdeployment. Add the new JMS Server for UMS called UMSJMSServer_n to this subdeployment. Click **Save**.

o. Update the SubDeployment targets for OIMJMSModule to include the recently created OIM JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **OIMJMSSystemResource** (represented as a hyperlink in the **Names** column of the table). The Settings page for OIMJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for OIMJMS appears.

> **Note:** This subdeployment module name is a random name in the form of OIMJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_OIM1 and WLS_OIM2).

p. Click the **OIMJMSServerXXXXXX** subdeployment. Add the new JMS Server for OIM called OIMJMSServer_n to this subdeployment. Click **Save**.

q. Update the SubDeployment targets for the JRFWSAsyncJMSSystemResource to include the recently created JRFWSAsync JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **JRFWSAsyncJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for JRFWSAsyncJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for JRFWSAsyncJMS appears.

> **Note:** This subdeployment module name is a random name in the form of JRFWSAsyncJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_OIM1 and WLS_OIM2).

r. Click the **JRFWSAsyncJMSServerXXXXXX** subdeployment. Add the new JMS Server for JRFWSAsync called JRFWSAsyncJMSServer_n to this subdeployment. Click **Save**.

s. Update the SubDeployment targets for the PS6SOAJMSSystemResource to include the recently created PS6SOA JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **PS6SOAJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for PS6SOAJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for PS6SOAJMS appears.

> **Note:** This subdeployment module name is a random name in the form of PS6SOAJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2).

t. Click the **PS6SOAJMSServerXXXXXX** subdeployment. Add the new JMS Server for PS6SOA called PS6SOAJMSServer_n to this subdeployment. Click **Save**.

4. Configure Oracle Coherence for deploying composites.

> **Note:** Only the localhost field needs to be changed for the server. Replace the localhost with the listen address of the new server added, for example:
>
> Dtangosol.coherence.localhost=SOAHOST1VHNn

5. Configure the transaction persistent store for the new server. This should be a shared storage location visible from other nodes.

   From the Administration Console, select **Server_name > Services** tab. Under Default Store, in Directory, enter the path to the folder where you want the default persistent store to store its data files.

6. Disable hostname verification for the new managed server. Before starting and verifying the WLS_SOAn managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Administration Server and the Node Manager in SOAHOSTn. If the source server from which the new one has been cloned had already disabled hostname verification, these steps are not required (the hostname verification settings is propagated to the cloned server).

   To disable hostname verification:

   a. In the Oracle Enterprise Manager Console, select Oracle WebLogic Server Administration Console.

   b. Expand the **Environment** node in the Domain Structure window.

   c. Click **Servers**. Select WLS_SOAn in the **Names** column of the table.

   d. Click the SSL tab. Click **Advanced.**

   e. Set **Hostname Verification** to **None**. Click **Save**.

7. Start and test the new managed server from the Administration Console.

   a. Shut down the existing managed servers in the cluster.

   b. Ensure that the newly created managed server, WLS_SOAn, is up.

   c. Access the application on the newly created managed server (http://*vip:port*/soa-infra). The application should be functional

8. Log in to the Administration Console. Select **Services** then the **Foreign JNDI provider** link. Select **ForeignJNDIProvider-SOA** then add the new server to the existing value.

9. Configure Server Migration for the new managed server.

   > **Note:** Because this is a scale up operation, the node must already contain a Node Manager, an environment configured for server migration, and the floating IP for the new SOA managed server.

   To configure server migration:

   a. Log into the Administration Console.

   b. In the left pane, expand **Environment** and select **Servers**.

   c. Select the server (represented as a hyperlink) for which you want to configure migration. The Settings page for that server appears.

   d. Click the Migration tab.

   e. In the Available field, in the Migration Configuration section, select the machines to which to enable migration and click the right arrow. Select the same migration targets as for the servers that already exist on the node.

For example, for new managed servers on SOAHOST1, which is already running WLS_SOA1, select SOAHOST2. For new managed servers on SOAHOST2, which is already running WLS_SOA2, select SOAHOST1.

Make sure the appropriate resources are available to run the managed servers concurrently during migration.

f. Select the **Automatic Server Migration Enabled** option. This enables the Node Manager to start a failed server on the target node automatically.

g. Click **Save**.

h. Restart the Administration Server, managed servers, and Node Manager.

i. Repeat the steps in this list for configuring server migration for the newly created WLS_OIMn managed server.

10. To test server migration for this new server, follow these steps from the node where you added the new server:

a. Stop the WLS_SOAn managed server.

To do this, run `kill -9` *pid* on the PID of the managed server. To identify the PID of the node, enter `ps -ef | grep WLS_SOAn`

b. Watch the Node Manager Console for a message indicating that WLS_SOA1's floating IP is disabled.

c. Wait for the Node Manager to try a second restart of WLS_SOAn. Node Manager waits for a fence period of 30 seconds before trying this restart.

d. After Node Manager restarts the server, stop it again. Node Manager logs a message indicating that the server will not restart again locally.

5.3.22.2 Scaling Out Oracle Identity Manager

When you scale out the topology, you add new managed servers configured with SOA to new nodes.

Before performing the steps in this section, check that you meet these requirements:

- There must be existing nodes running managed servers configured with SOA within the topology.
- The new node can access the existing home directories for WebLogic Server and SOA. (Use the existing installations in shared storage for creating a new WLS_SOA or WLS_OIM managed server. You do not need to install WebLogic Server or SOA binaries in a new location but you do need to run pack and unpack to bootstrap the domain configuration in the new node.)

Note: If there is no existing installation in shared storage, you must install WebLogic Server and SOA in the new nodes.

> **Note:** When an *ORACLE_HOME* or *WL_HOME* is shared by multiple servers in different nodes, Oracle recommends keeping the Oracle Inventory and Middleware home list in those nodes updated for consistency in the installations and application of patches. To update the oraInventory in a node and "attach" an installation in a shared storage to it, use *ORACLE_HOME*/oui/bin/attachHome.sh. To update the Middleware home list to add or remove a WL_HOME, edit the user_home/bea/beahomelist file. See the following steps.

Follow these steps for scaling out the topology:

1. On the new node, mount the existing Middleware home, which should include the SOA installation and the domain directory, and ensure that the new node has access to this directory, just like the rest of the nodes in the domain.
2. To attach *ORACLE_HOME* in shared storage to the local Oracle Inventory, run the following commands:

   ```
   SOAHOSTn>cd ORACLE_BASE/product/fmw/soa/
   SOAHOSTn>./attachHome.sh -jreLoc ORACLE_BASE/fmw/jrockit_160_<version>
   ```

   To update the Middleware home list, create (or edit, if another WebLogic installation exists in the node) the *MW_HOME*/bea/beahomelist file and add *ORACLE_BASE*/product/fmw to it.
3. Log in to the Oracle WebLogic Administration Console.
4. Create a new machine for the new node that will be used, and add the machine to the domain.
5. Update the machine's Node Manager's address to map the IP of the node that is being used for scale out.
6. Use the Administration Console to clone WLS_SOA1 into a new managed server. Name it WLS_SOAn, where n is a number.

   > **Note:** These steps assume that you are adding a new server to node n, where no managed server was running previously.

7. Assign the hostname or IP to use for the new managed server for the listen address of the managed server.

   If you are planning to use server migration for this server (which Oracle recommends), this should be the VIP (also called a floating IP) for the server. This VIP should be different from the one used for the existing managed server.
8. Create JMS servers for SOA, OIM (if applicable), UMS, BPM, JRFWSAsync, and PS6SOA on the new managed server.

   a. Use the Administration Console to create a new persistent store for the new SOAJMSServer and name it, for example, SOAJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

   ```
   ORACLE_BASE/admin/domain_name/cluster_name/jms/SOAJMSFileStore_n
   ```

Note: This directory must exist before the managed server is started or the start operation fails.

b. Create a new JMS Server for SOA, for example, SOAJMSServer_n. Use the SOAJMSFileStore_n for this JMSServer. Target the SOAJMSServer_n Server to the recently created Managed Server (WLS_SOAn).

c. Create a new persistence store for the new UMSJMSServer, for example, UMSJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

```
ORACLE_BASE/admin/domain_name/cluster_name/jms/UMSJMSFileStore_n
```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new UMS JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

d. Create a new JMS Server for UMS, for example, UMSJMSServer_n. Use the UMSJMSFileStore_n for this JMSServer. Target the UMSJMSServer_n Server to the recently created managed server (WLS_SOAn).

e. Create a new persistence store for the new OIMJMSServer, for example, OIMJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

```
ORACLE_BASE/admin/domain_name/cluster_name/jms/OIMJMSFileStore_n
```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new OIM JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

f. Create a new JMS Server for OIM, for example, OIMJMSServer_n. Use the OIMJMSFileStore_n for this JMSServer. Target the OIMJMSServer_n Server to the recently created Managed Server (WLS_OIMn).

g. Create a new persistence store for the new BPMJMSServer, for example, BPMJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

```
ORACLE_BASE/admin/domain_name/cluster_name/jms/BPMJMSFileStore_n
```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new BPM JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

h. Create a new JMS Server for BPM, for example, BPMJMSServer_n. Use the BPMJMSFileStore_n for this JMSServer. Target the BPMJMSServer_n Server to the recently created managed server (WLS_SOAn).

i. Create a new persistence store for the new JRFWSAsyncJMSServer, for example, JRFWSAsyncJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

```
ORACLE_BASE/admin/domain_name/cluster_name/jms/JRFWSAsyncJMSFileStore_n
```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new JRFWSAsync JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

j. Create a new JMS Server for JRFWSAsync, for example, JRFWSAsyncJMSServer_n. Use the JRFWSAsyncJMSFileStore_n for this JMSServer. Target the JRFWSAsyncJMSServer_n Server to the recently created managed server (WLS_SOAn).

k. Create a new persistence store for the new PS6SOAJMSServer, for example, PS6SOAJMSFileStore_n. Specify the path for the store. This should be a directory on shared storage.

```
ORACLE_BASE/admin/domain_name/cluster_name/jms/PS6SOAJMSFileStore_n
```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_n as store for the new PS6SOA JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

l. Create a new JMS Server for PS6SOA, for example, PS6SOAJMSServer_n. Use the PS6SOAJMSFileStore_n for this JMSServer. Target the PS6SOAJMSServer_n Server to the recently created managed server (WLS_SOAn).

m. Update the SubDeployment targets for the SOA JMS Module to include the recently created SOA JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **SOAJMSModule** (represented as a hyperlink in the **Names** column of the table). The Settings page for SOAJMSModule appears. Click the **SubDeployments** tab. The subdeployment module for SOAJMS appears.

Note: This subdeployment module name is a random name in the form of SOAJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2).

---

n. Click the **SOAJMSServerXXXXXX** subdeployment. Add the new JMS Server for SOA called SOAJMSServer_n to this subdeployment. Click **Save**.

o. Update the SubDeployment targets for the UMSJMSSystemResource to include the recently created UMS JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **UMSJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for UMSJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for UMSJMS appears.

Note: This subdeployment module name is a random name in the form of UMSJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2).

---

p. Click the **UMSJMSServerXXXXXX** subdeployment. Add the new JMS Server for UMS called UMSJMSServer_n to this subdeployment. Click **Save**.

q. Update the SubDeployment targets for OIMJMSModule to include the recently created OIM JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Administration Console. The JMS Modules page appears. Click **OIMJMSSystemResource** (represented as a hyperlink in the **Names** column of the table). The Settings page for OIMJMSSystemResource appears. Click the SubDeployments tab. The subdeployment module for OIMJMS appears.

Note: This subdeployment module name is a random name in the form of OIMJMSServerXXXXXX resulting from the Configuration Wizard JMS configuration for the first two servers (WLS_OIM1 and WLS_OIM2).

---

r. Click the **OIMJMSServerXXXXXX** subdeployment. Add the new JMS Server for OIM called OIMJMSServer_n to this subdeployment. Click **Save**.

9. Run the pack command on SOAHOST1 to create a template pack as follows:

```
SOAHOST1> cd ORACLE_COMMON_HOME/common/bin
SOAHOST1> ./pack.sh -managed=true/
-domain=MW_HOME/user_projects/domains/soadomain/
-template=soadomaintemplateScale.jar -template_name=soa_domain_templateScale
```

Run the following command on SOAHOST1 to copy the template file created to SOAHOSTn:

```
SOAHOST1> scp soadomaintemplateScale.jar oracle@SOAHOSTN:/
ORACLE_BASE/product/fmw/soa/common/bin
```

Run the unpack command on SOAHOSTn to unpack the template in the managed server domain directory as follows:

```
SOAHOSTn> cd ORACLE_BASE/product/fmw/soa/common/bin
SOAHOSTN> ./unpack.sh /
-domain=ORACLE_BASE/product/fmw/user_projects/domains/soadomain/
-template=soadomaintemplateScale.jar
```

10. Configure Oracle Coherence for deploying composites.

> **Note:** Only the localhost field needs to be changed for the server. Replace the localhost with the listen address of the new server added, for example:
>
> Dtangosol.coherence.localhost=SOAHOST1VHNn

11. Configure the transaction persistent store for the new server. This should be a shared storage location visible from other nodes.

    From the Administration Console, select **Server_name > Services** tab. Under Default Store, in Directory, enter the path to the folder where you want the default persistent store to store its data files.

12. Disable hostname verification for the new managed server. Before starting and verifying the WLS_SOAn managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in SOAHOSTn. If the source server from which the new one has been cloned had already disabled hostname verification, these steps are not required (the hostname verification settings is propagated to the cloned server).

    To disable hostname verification:

    a. In the Oracle Enterprise Manager Console, select Oracle WebLogic Server Administration Console.

    b. Expand the **Environment** node in the Domain Structure window.

    c. Click **Servers**. The Summary of Servers page appears.

    d. Select WLS_SOAn in the **Names** column of the table. The Settings page for the server appears.

    e. Click the SSL tab.

    f. Click **Advanced**.

    g. Set **Hostname Verification** to **None**.

    h. Click **Save**.

13. Start the Node Manager on the new node. To start the Node Manager, use the installation in shared storage from the existing nodes, and start Node Manager by passing the hostname of the new node as a parameter as follows:

```
SOAHOSTn> WL_HOME/server/bin/startNodeManager new_node_ip
```

14. Start and test the new managed server from the Administration Console.

    a. Shut down the existing managed servers in the cluster.

    b. Ensure that the newly created managed server, WLS_SOAn, is up.

   c. Access the application on the newly created managed server (http://*vip:port*/soa-infra). The application should be functional

15. Configure Server Migration for the new managed server.

   Note: Since this new node is using an existing shared storage installation, the node is already using a Node Manager and environment configured for server migration that includes netmask, interface, wlsifconfig script superuser privileges. The floating IP for the new SOA managed server is already in the new node.

   Configure server migration by following these steps:

   a. Log into the Administration Console.

   b. In the left pane, expand **Environment** and select **Servers**.

   c. Select the server (represented as a hyperlink) for which you want to configure migration. The Settings page for that server appears.

   d. Click the Migration tab.

   e. In the Available field, in the Migration Configuration section, select the machines to which to enable migration and click the right arrow.

   Note: Specify the least-loaded machine as the migration target for the new server. The required capacity planning must be completed so that this node has enough available resources to sustain an additional managed server.

   f. Select the **Automatic Server Migration Enabled** option. This enables the Node Manager to start a failed server on the target node automatically.

   g. Click **Save**.

   h. Restart the Administration Server, managed servers, and Node Manager.

16. Test server migration for this new server. Follow these steps from the node where you added the new server:

   a. Stop the WLS_SOAn managed server.

   To do this, run `kill -9` *pid* on the PID of the managed server. You can identify the PID of the node using `ps -ef | grep WLS_SOAn`.

   b. Watch the Node Manager Console: you should see a message indicating that WLS_SOA1's floating IP has been disabled.

   c. Wait for the Node Manager to try a second restart of WLS_SOAn. Node Manager waits for a fence period of 30 seconds before trying this restart.

   d. Once Node Manager restarts the server, stop it again. Now Node Manager should log a message indicating that the server will not be restarted again locally.

6

Configuring High Availability for Oracle Access Management Access Manager Components

This chapter provides an introduction to Oracle Access Management Access Manager (Access Manager) and describes how to design and deploy a high availability environment for Access Manager.

Access Manager provides a single authoritative source for all authentication and authorization services. The core service provided is the checking of valid session tokens, the requesting of credentials if the session token is invalid or missing, and the issuing of session tokens, intercepting resource requests and evaluating access control policies to control access to resources. Access Manager features a pure Java server while continuing to use Oracle Single Sign-On 10*g* and Oracle Access Manager 10*g* agent components. For more information on Access Manager, see Introduction to Oracle Access Management Access Manager in the *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management*.

This chapter includes the following topics:

- Section 6.1, "Access Manager Component Architecture"
- Section 6.2, "Access Manager High Availability Concepts"
- Section 6.3, "Access Manager High Availability Configuration Steps"

6.1 Access Manager Component Architecture

Figure 6–1 shows the Access Manager component architecture.

Figure 6–1 Access Manager Single Instance Architecture

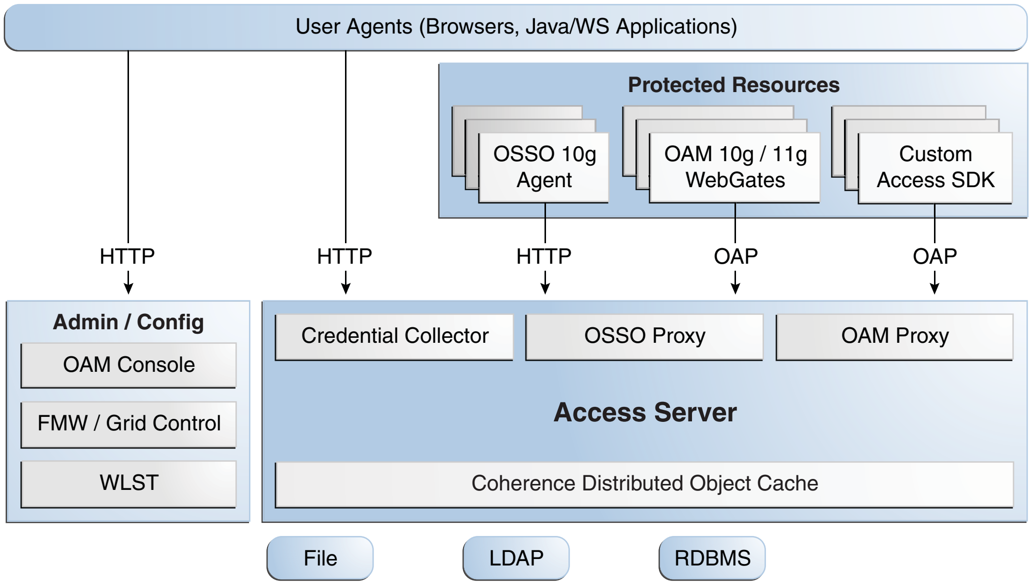


Figure 6–1 shows the following components:

- User agents: These include web browsers, Java applications, and Web services applications. The user agents access the Access Server and the administration and configuration tools using HTTP.
- Protected resources: A protected resource is an application or web page to which access is restricted. Access to protected resources is controlled by WebGates or Custom Agents.
- Administration and configuration tools: Administer and configure Access Manager with the Oracle Access Management Console, the Oracle Enterprise Manager Fusion Middleware Control and the Oracle Enterprise Manager Grid Control, and the WebLogic Scripting Tool (WLST).
- Access Server: The Access Server includes the Credential Collector, OSSO Proxy, and OAM Proxy components. The Coherence Distributed Object Cache is used to propagate configuration file changes between Access Servers

6.1.1 Access Manager Component Characteristics

An Access Manager deployment consists of these system entities:

- Access Manager Agents - Access Manager agents are extensions of the Access Server that are responsible for ensuring that access is controlled according to policies that Access Server manages.

  Agents require the Access Server component to perform their functions. If the Access Server is unavailable, access to protected servers is not permitted and users are locked out of the system.
- Protected Resources (partnered applications) - Applications that are protected by Access Manager. Access to these resources is subject to the access control policies in Access Manager and is enforced by Access Manager agents that are deployed in the access path of the protected resource (for example, Access Manager agents deployed in the Web Server).

- Access Server - The server side component that provides core runtime access management services.
- JMX Mbeans - Runtime Mbeans are packaged as part of the Access Server package. Config Mbeans are packaged as standalone WAR files.
- WebLogic 11*g* SSPI providers are composed of Java classes that implement the SSPI interface along with the Access Java Access JDK. AccessGates are built using the pure Java Access JDK.
- Oracle Access Management Console - A J2EE application that hosts the Administration Console and provides services like Administration/Configuration to manage the Access Manager deployment.
- WebLogic Scripting Tool (WLST) is composed of Java classes, which are included in the Access Server package. Limited administration of the Access Manager deployment is supported via the command line.
- Fusion Middleware Control and Enterprise Manager Grid Control - Access Manager integrates with the Enterprise Manager Grid Control to display performance metrics and deployment topology.
- Coherence Distributed Object Cache - Access Manager components rely on this infrastructure for real time change propagation.
- The Access Manager Proxy is a customized version of the Apache MINA server (based on the JCA architecture), which includes MessageDrivenBeans and ResourceAdapters in addition to Java Server classes.
- The Oracle Single Sign-On (OSSO) Proxy comprises Java classes, which are included in the Access Server package.
- Data Repositories - Access Manager handles different types of information including Identity, Policy, Partner, Session and Transient data:
  - LDAP for Identity data
  - Files for Configuration and Partner data
  - Coherence in-memory for Session and Transient Data
  - Policy data will be stored in files or in an RDBMS
- Oracle Access Manager 10*g* WebGates are C-based agents that are intended to be deployed in web servers.
- Oracle Single Sign-On Apache modules are C-based agents that are intended to be deployed in Oracle HTTP Server web servers.
- Access Manager 11*g* WebGates are C-based agents that are intended to be deployed in web servers.

6.1.1.1 Access Manager State Information

Authenticated user session information is persisted via the Coherence Distributed Object Cache. Use the Coherence Distributed Object Cache in-memory mode for Access Manager.

Access Manager may create a transient state for unauthenticated users during the login processing. This state is generally not replicated among Access Manager nodes. To protect against effects of node failures during the login processing, the state may be optionally stored in an encrypted client cookie.

To store the transient state for unauthenticated users during login processing, change the OAM server parameter RequestCacheType from BASIC to COOKIE by following these steps:

1. Set up the environment for WLST by running this command:

   ```
   DOMAIN_HOME/bin/setDomainEnv.sh
   ```

2. Start WLST by issuing this command:

   ```
   Start WLST by issuing this command:
   ORACLE_HOME/common/bin/wlst.sh
   ```

3. Connect to your domain:

   ```
   wls:/IDM_Domain/serverConfig> connect()
   ```

4. Enter the WebLogic Administration username and password, and enter the URL for the Administration Server in the format:

   ```
   t3://OAMHOST1.example.com:7001
   ```

5. Issue this command:

   ```
   wls:/IDM_Domain/serverConfig> configRequestCacheType(type="COOKIE")
   ```

6. Check that the command worked by issuing this command:

   ```
   wls:/IDM_Domain/serverConfig> displayRequestCacheType()
   ```

7. Restart the Access Manager managed servers.

6.1.1.2 Access Manager Request Flow

The following list describes the steps in an Access Manager request flow:

1. The user tries to access a Access Manager protected Web Resource using a web browser.
2. The Access Manager agent[1] intercepts the request and tries to ascertain if the user has an authenticated session.
3. Because this is the user's first access, the user is redirected to the Access Manager 11*g* Access Server for authentication.
4. Access Server's credential collector[2] component shows a Login Form.[3] The user submits his credentials to the Access Server.
5. Access Server validates the user credentials and generates a security token (cookie). The user is redirected to the resource he tried to access in Step 1.
6. The Access Manager agent intercepts the request and extracts the security token (cookie).
7. The Access Manager agent makes a back channel call[4] to the Access Server (OAP over TCP) to validate the session and authorize the request.

[1] The agent in use is specific to a deployment and different types of agents (with different features) can be used in a deployment.

[2] In addition to the built-in Credential Collector, Access Manager is capable of supporting external credential collectors.

[3] The credential collection will be different for non-username and password authentication schemes.

[4] Only WebGates support back channel communication.

8. Access server verifies the user's permissions against the configured policy for the web resource.
9. Access server responds to the WebGate request indicating that access is enabled.
10. The Access Manager agent allows the request to go through.[5]
11. The user can now access the web resource he tried to access in Step 1.

6.1.1.2.1 Access Manager Process Lifecycle You can start Access Server and Administration Console from the user interface and command line tools that WebLogic Server provides.

The Access Server supports a health check request (a ping request over HTTP) that the load balancer can use for death detection.

Access Manager agents are native applications that reside in the protected application environment. No tools are provided as part of OAM but it is expected that environment specific tooling, where available, will be leveraged for the above purpose.

Access Manager is instrumented for server side metrics (using DMS) and this information is published to the WebLogic Administration Console. Using DMS metrics collection, you can monitor the agent and server component metrics as a proxy for component monitoring. In addition, Access Manager supports fine-grained real time activity tracing, which can also serve as a proxy for component monitoring.

On startup, Access Server initializes connections to the backend repositories. If the repository is not reachable, the Access Server retries the connections to the repositories, using a timeout that grows exponentially with a configurable upper bound.

Access Server will provide continuity of service based on locally cached data if the backend connections go down. This will continue until the caches grow stale or the backend connections become alive again.

6.1.1.2.2 Access Manager Configuration Artifacts The Access Manager configuration artifacts include these files:

- *DOMAIN_HOME*/config/fmwconfig/oam-config.xml

  The configuration file, which contains instance specific information.

- *DOMAIN_HOME*/config/fmwconfig/oam-policy.xml
- *DOMAIN_HOME*/config/fmwconfig/.oamkeystore

  This is used for storing symmetric and asymmetric keys.

- *DOMAIN_HOME*/config/fmwconfig/component_events.xml

  Used for audit definition.

- *DOMAIN_HOME*/config/fmwconfig/jazn-data.xml

  Used for Administration Console permissions

- *DOMAIN_HOME*/config/fmwconfig/servers/*instanceName*/logging.xml

  Used for logging configuration. Do not edit this file manually.

- *DOMAIN_HOME*/config/fmwconfig/servers/*instanceName*/dms_config.xml

  Used for tracing logging. Do not edit this file manually.

[5] The agent may perform some housekeeping tasks, such as session refresh, before enabling the request to go through to the web resource.

- *DOMAIN_HOME*/config/fmwconfig/cwallet.sso

  Used to store passwords that OAM uses to connect to identity stores, database, and other entities. This is not for end user passwords.

6.1.1.2.3 Access Manager External Dependencies

Access Manager has external runtime dependencies on:

- LDAP based Identity Store
  - User Identity Repository
  - LDAP access abstracted by User/Role API.

  > **Note:** Access Manager always connects to one Identity store, which can be a physical server or a load balancer IP. If the primary is down, Access Manager reconnects and expects the load balancer to connect it to the secondary.

- OCSP Responder Service
  - Real-time X.509 Certification Validation
- RDBMS Policy Store
  - Policy (Authentication and Authorization) Repository
  - RDBMS access abstracted by the OAM policy engine
- Oracle Identity Manager (when OIM-based password management is enabled)
  - Oracle Identity Manager provides Password Management Services and replaces the Oracle Access Manager 10*g* Identity Server
- Oracle Identity Manager Policy Store (when Oracle Identity Manager-based password management is enabled)
  - LDAP Repository containing Oblix Schema elements that are used to store Configuration, Metadata, and so on.
- Oracle Adaptive Access Manager (when Oracle Adaptive Access Manager Advanced Authentication Scheme is selected)
- Identity Federation (when Identity Federation Authentication Scheme is selected)

6.1.1.2.4 Access Manager Log File Location

Access Manager is deployed on WebLogic Server. All log messages log to the server log file of the WebLogic Server that the application is deployed on. The default server log location is:

```
WL_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

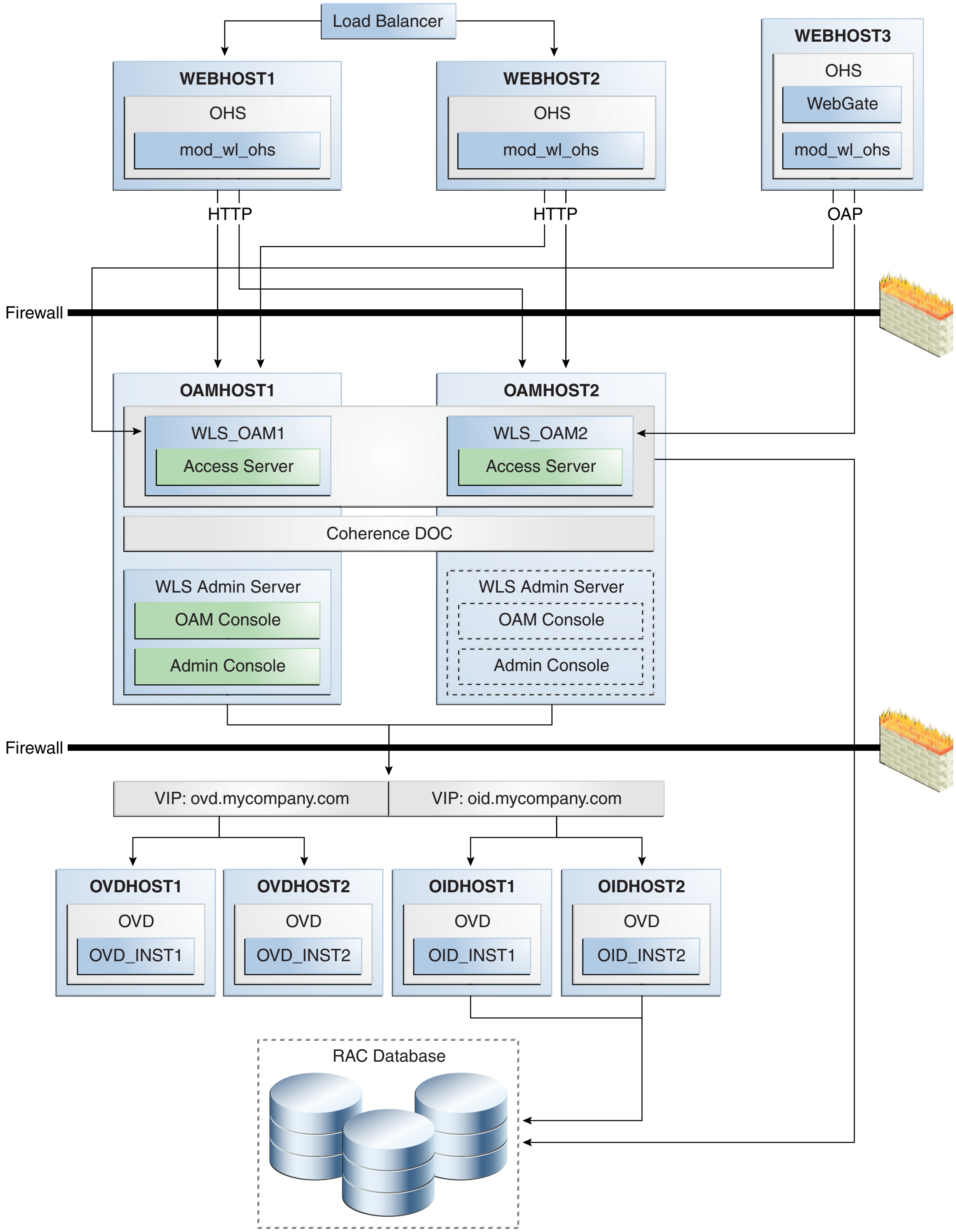
6.2 Access Manager High Availability Concepts

This section provides conceptual information about using Access Manager in a high availability two-node cluster.

6.2.1 Access Manager High Availability Architecture

Figure 6–2 shows an Access Manager high availability architecture:

Figure 6–2 Access Manager High Availability Architecture



In Figure 6–2, incoming authentication requests are received by the hardware load balancer, which routes them to WEBHOST1 or WEBHOST2 in the web tier. These hosts have Oracle HTTP Server installed. The Oracle HTTP Server then forwards requests on to the WebLogic managed servers using the WebLogic plugin `mod_wl_ohs.conf`. A

Distributed Credential Collector enables a WebGate to collect the credentials and send them to the Access Manager server by means of the OAP protocol.

The load balancing router should use session stickiness for HTTP traffic only. OAP traffic does not use a load balancing router, so session stickiness is not required for OAP traffic.

Applications that other Oracle HTTP Servers access that in turn have resources with restricted access must have a WebGate, Oracle Single Sign-On Server agent (mod_osso agent), OpenSSO Policy agent, or custom agent configured. The WebGate on WEBHOST3 communicates with the Access Servers on OAMHOST1 and OAMHOST2 in the application tier using OAP. WEBHOST3 is an application web server, and for authentication, HTTP redirect is used to route requests to the load balancer and to WEBHOST1 and WEBHOST2. For a high availability deployment, you can optionally configure another host (for example, WEBHOST4) with the same components as WEBHOST3.

OAMHOST1 and OAMHOST2 deploy managed servers which host the Oracle Access Server application. These managed servers are configured in a cluster which enables the Access Servers to work in an active-active manner.

The WebLogic Administration Server runs on OAMHOST1 and deploys the WebLogic Administration Console, Oracle Enterprise Manager Fusion Middleware Control, and the Oracle Access Management Console. The Administration Server can be configured to run in active-passive mode on OAMHOST2, which means that if OAMHOST1 becomes unavailable, then Administration Server can be manually started on OAMHOST2.

In the directory tier, the virtual IP `ovd.example.com` is set up to route Oracle Virtual Directory requests to OVDHOST1 and OVDHOST2, which comprise an active-active Oracle Virtual Directory cluster. The virtual IP `oid.example.com` is set up to route Oracle Internet Directory requests to OIDHOST1 and OIDHOST2, which comprise an active-active Oracle Internet Directory cluster.

An Oracle RAC database provides high availability in the data tier.

In Access Manager 11*g*, only one Access Manager cluster is supported per WebLogic Server domain. Access Manager clusters cannot span WebLogic Server domains.

A single instance Access Manager deployment satisfies the following high availability requirements:

- Load handling
- External connection management and monitoring
- Recovery
- Fault containment
- Fault diagnostics
- Administration Server offline

A multiple instance Access Manager deployment satisfies the following additional high availability requirements:

- Redundancy
- Client connection failover/continuity
- Client load balancing
- State management

Use of an external load balancing router is recommended for inbound HTTP connections. Outbound external connections to LDAP Servers (or OAM policy engine PDP/PIP) are load balanced with support for connection failover. Therefore, a load balancer is not required. Access Manager agents, typically WebGates, can load balance connections across multiple Access Servers.

Access Manager agents open persistent TCP connections to the Access Servers. This requires firewall connection timeouts to be sufficiently large to avoid premature termination of TCP connections.

The Access Server and Access Manager Administration Console interface with the OAM policy engine for policy evaluation and management. The OAM policy engine internally depends on a database as the policy repository. The database interactions are encapsulated within the OAM policy engine, with only the connectivity configuration information managed by Access Manager. The high availability characteristics of the interaction between Access Manager and the OAM policy engine are:

- The database connection information is configured in the Access Manager configuration file synchronized among the Access Manager instances.
- Database communication is managed within the OAM policy engine, and generally decoupled from Access Manager and OAM policy engine interactions. The very first startup of an OAM server instance will fail, however, if the database is unreachable. An OAM policy engine bootstrap failure is treated as fatal by Access Manager, and the startup operation is aborted.
- Transient database unavailability is transparently tolerated by OAM policy engine policy evaluation services, enabling Access Manager server runtimes to continue functioning uninterrupted. After the initial OAM policy engine bootstrap, the Access Manager instances may restart while the database is unreachable; the OAM policy engine continues to operate against its locally cached policies.
- Access Manager policy management interfaces (in the Oracle Access Management Console and the CLI tool) fail if the database is unreachable, as seen by the OAM policy engine management service interfaces. The operation may be retried at a later point in time, but no automated retry is provided for management operations.
- Following a successful policy modification in the database repository, the OAM policy engine layer in the OAM server runtimes retrieves and activates the changes within a configurable OAM policy engine database poll interval (configured through Access Manager configuration). A positive acknowledgement of a policy change must be received from each OAM server runtime, otherwise the policy change cannot be considered successfully activated. The administrator can use the Oracle Access Management Console to remove any Access Manager instance with a policy activation failure from service.

6.2.1.1 Starting and Stopping the Cluster

In a high availability architecture, OAM server is deployed on an Oracle WebLogic Cluster, which has at least two servers as a part of the cluster.

By default, WebLogic Server starts stops, monitors and manages the various lifecycle events for the application. The Access Manager application leverages the high availability features of the underlying Oracle WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic Servers. In case of failure, Node Manager restarts the WebLogic Server.

6.2.1.1.1 Cluster-Wide Configuration Changes The standard Java EE artifacts that Access Manager uses are configured as part of the Oracle WebLogic domain in which Access Manager is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data sources, across the WebLogic Server domain. At the same time, the WebLogic Server cluster synchronizes the deployments and libraries used by the Access Manager components.

Additionally, Access Manager application level configuration is stored in the Access Manager repository. Propagation of Access Manager configuration changes to all the cluster members is based on a distribution mechanism that leverages the Coherence Distributed Object Cache. All Access Manager components are notified of change events from the coherence layer, which are then taken up by the components. To ensure atomicity of the change, Access Manager components reload their entire configuration every time a change happens.

Access Manager configuration applies to all instances in a cluster. The only exceptions to the above (instance-specific configuration) supported in Access Manager are the Access Manager proxy host, Access Manager proxy port, and the instance-specific Coherence configuration (when Well Known Addresses (WKA) is used). The IP address of the proxy host and proxy port are stored in a configuration file. The Access Manager proxy port is the endpoint for OAP requests from agents. The IP address of the Coherence WKA is also stored in a configuration file. The Coherence WKA is used to determine the Coherence nodes that are authorized to receive Access Manager-specific traffic. The `oam-config.xml` file is the configuration file that stores this configuration information.

Adding and removing Access Server instances is transparent to other Access Manager Access Server instances in the cluster. However, take care to ensure that the removal of a specific Access Server does not affect the load balancing and failover characteristics of the agents.

Restarting an Access Manager Access Server has no impact on any other running components or members of the cluster.

6.2.2 Protection from Failures and Expected Behaviors

This section describes how an Oracle Identity Management high availability cluster deployment protects components from failure and expected behavior if component failure occurs.

The WebLogic Server infrastructure protects the Identity Management Service Infrastructure system from all process failures.

The following features protect the Access Manager high availability configuration from failure:

- Back channel OAP bindings use a primary/secondary model for failover. Front Channel HTTP bindings use a load balancing router for failover.
- Session state is maintained in the Coherence Distributed Object Cache, which provides replication and failover for all session state information. Data stored in the Coherence cache is written asynchronously to a database. This data should survive a restart of all access servers, however, a small amount of data can be lost due to the asynchronous nature of the write through.

- If an Access Server fails, a WebGate with a persistent connection to that server waits for the connection to timeout, then it switches over to the secondary (backup) Access Server. Outstanding requests fail over to the secondary server.
- Access Manager Access Servers support a heartbeat check. Also, the WebLogic Node Manager on the Managed Server can monitor the application and restart it.
- If a WebLogic Server node fails, external connection failover is based on the configuration, the retry timeout, and the number of retries. Access Manager Agent-Access Server failover is based on a timeout.
- If the load balancing router or proxy server detects a WebLogic Server node failure, subsequent client connections route to the active instance, which picks up the session state and carries on with processing.
- When the lifetime of a connection expires, pending requests complete before the connection terminates. The connection object returns to the pool.
- When it receives an exception from another service, Access Manager retries external connection requests. You can configure the number of retries.

6.2.2.1 WebLogic Server Crash

If a WLS_OAMx server fails, Node Manager attempts to restart it locally

Ongoing requests from the HTTP Server timeout and new requests are directed to the other WLS_OAMx server. Once the server's restart completes on the failed node, Oracle HTTP Server resumes routing any incoming requests to the server.

> **Note:** Access Manager servers support a heartbeat check to determine if the access server can service its requests. It checks:
>
> - Whether the LDAP store can be accessed
> - Whether the policy store can be accessed
>
> If the heartbeat succeeds, the Access Server can service requests and requests are sent to it. If the heartbeat fails, requests do not route to the Access Server.

6.2.2.2 Node Failure

Node failures are treated in the same way as WebLogic Server fails.

6.2.2.3 Database Failure

The Access Manager service Infrastructure is protected against failures in the database by using multi data sources. These multi-data sources are typically configured during the initial set up of the system (Oracle Fusion Middleware Configuration Wizard enables you to define these multi-pools directly at installation time). They guarantee that when an Oracle RAC database instance fails, the connections are reestablished with available database instances. The multi data source enables you to configure connections to multiple instances in an Oracle RAC database.

For information about multi data source configuration with Oracle RAC and the MDS repository, see Section 4.1.3, "Using Multi Data Sources with Oracle RAC."

6.3 Access Manager High Availability Configuration Steps

This section provides high-level instructions for setting up a high availability deployment for Access Manager. This deployment includes two Oracle HTTP Servers, which distribute requests to two OAM servers. These OAM servers interact with an Oracle Real Application Clusters (Oracle RAC) database and, optionally, an external LDAP store. If any single component fails, the remaining components continue to function.

This section includes the following topics:

- Section 6.3.1, "Prerequisites for Access Manager Configuration"
- Section 6.3.2, "Running the Repository Creation Utility to Create the Database Schemas"
- Section 6.3.3, "Installing Oracle WebLogic Server"
- Section 6.3.4, "Installing and Configuring the Access Manager Application Tier"
- Section 6.3.5, "Configuring the Database Security Store"
- Section 6.3.6, "Creating boot.properties for the Administration Server on OAMHOST1"
- Section 6.3.7, "Starting OAMHOST1"
- Section 6.3.8, "Validating OAMHOST1"
- Section 6.3.9, "Configuring OAM on OAMHOST2"
- Section 6.3.10, "Starting OAMHOST2"
- Section 6.3.11, "Validating OAMHOST2"
- Section 6.3.12, "Configure Access Manager to Work with Oracle HTTP Server"
- Section 6.3.13, "Configuring Access Manager to use an External LDAP Store"
- Section 6.3.14, "Validating the Access Manager Configuration"
- Section 6.3.15, "Configuring Oracle Coherence to Keep Configuration Files in Sync"
- Section 6.3.16, "Scaling Up and Scaling Out the Access Manager Topology"

6.3.1 Prerequisites for Access Manager Configuration

Before you configure Access Manager for high availability, you must:

- Run the Repository Creation Utility to create the Access Manager schemas in a database. See Section 6.3.2, "Running the Repository Creation Utility to Create the Database Schemas".
- Install Oracle WebLogic Server on OAMHOST1 and OAMHOST2. See Section 6.3.3, "Installing Oracle WebLogic Server".
- Install the Oracle Identity Management executables on OAMHOST1 and OAMHOST2. See the Section 6.3.4, "Installing and Configuring the Access Manager Application Tier".
- Ensure that a highly available LDAP implementation is available.

6.3.2 Running the Repository Creation Utility to Create the Database Schemas

The schemas you create depend on the products you want to install and configure. Use the Repository Creation Utility (RCU) that is version compatible with the product you are installing. See the *Oracle Fusion Middleware Installation Planning Guide for Oracle Identity and Access Management* and *Oracle Fusion Middleware Repository Creation Utility User's Guide* to run RCU.

6.3.3 Installing Oracle WebLogic Server

Prior to installing the Oracle WebLogic Server, ensure that your machines meet the system, patch, kernel, and other requirements as specified in *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

Start the installer, then proceed as follows:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, select **Create a New Middleware Home**.

   For Middleware Home Directory, enter:

   ORACLE_BASE`/product/fmw`

   > **Note:** *ORACLE_BASE* is the base directory under which Oracle products are installed. The recommended value is `/u01/app/oracle`.

   Click **Next**.
3. Enter your contact information so that you can be notified of security updates.

   Click **Next**.
4. On the Choose Install Type screen, select **Custom**.

   Click **Next**.
5. On the Choose Products and Components screen, select only **Oracle JRockit SDK**, and click **Next**.
6. On the Choose Product Installation Directories screen, accept the directory *ORACLE_BASE*`/product/fmw/wlserver_10.3`.

   Click **Next**.
7. On the Installation Summary screen, click **Next**.
8. On the Installation Complete screen, deselect **Run Quickstart**.

   Click **Done**.

6.3.4 Installing and Configuring the Access Manager Application Tier

This section describes how to install and configure Oracle Fusion Middleware components on OAMHOST1 and OAMHOST2.

6.3.4.1 Install Oracle Fusion Middleware for Identity Management

This section includes the steps for installing the Oracle Identity Management software into the previously created Middleware Home directory. Perform the steps on OAMHOST1 and OAMHOST2.

If the `/etc/oraInst.loc` file exists, verify that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory. If the `/etc/oraInst.loc` file does not exist, you can skip this step.

Start the installer for Oracle Fusion Middleware as follows:

```
OAMHOST1> runInstaller
```

When the installer prompts you for a JRE/JDK location, enter the Oracle SDK location created in the WebLogic Server installation, for example, *ORACLE_BASE*`/product/fmw/jrockit_160_`<*version*>.

Then proceed as follows:

1. On the Welcome screen, click **Next**.
2. On the Prerequisite Checks screen, verify that the checks complete successfully, then click **Next**.
3. On the Specify Installation Location screen, enter the following values:
   - Oracle Middleware Home: Select the previously installed Middleware home from the list for `MW_HOME`, for example:

     ```
     /u01/app/oracle/product/fmw
     ```

   - Oracle Home Directory: Enter `idm`.

   Click **Next**.
4. On the Installation Summary screen, click **Install**.
5. On the Installation Complete screen, click **Finish**.

6.3.4.2 Configure Oracle Access Manager on OAMHOST1

This section creates the domain on OAMHOST1.

Start the configuration wizard by running the command:

```
MW_HOME/oracle_common/common/bin/config.sh
```

Then proceed as follows:

1. In the Welcome screen, select **Create a New WebLogic Domain**, and then click **Next**.
2. In the Select Domain Source Screen:

   Select **Generate a domain configured automatically to support the following products**:

   And select the following products:

   - **Oracle Enterprise Manager**
   - **Oracle JRF** (selected by default)
   - **Oracle Access Management**
   - **Oracle Platform Security Service**

   Click **Next**.
3. In the Specify Domain and Location screen enter:
   - **Domain name**: IDM_Domain
   - **Domain Location**: Accept the default.

- **Application Directory**: Accept the default.

   Click **Next**.

4. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator, and click **Next**.

5. In the Configure Server Start Mode and JDK screen, make the following selections:

   - **WebLogic Domain Startup Mode**: Select **Production Mode**.
   - **JDK Selection**: Select **JROCKIT SDK1.6.0_***<version>*.

6. In the Configure JDBC Component Schema screen, select all of the data sources, then select **Configure selected data sources as RAC multi data sources**.

   Click **Next**.

7. In the Configure RAC Multi Data Source Component Schema screen, select the first data source, **OAM Infrastructure**, and enter the following:

   - **Data source**: OAM
   - **Service Name**: `oam.example.com`
   - **User Name**: OAM_OAM (assuming OAM was used as the RCU prefix)
   - **Password**: The password for above account

   In the top right box, click **Add** to add an Oracle RAC host. Enter the following information:

   - **Host Name**: OAMDBHOST1
   - **Instance Name**: oamdb1
   - **Port**: 1521

   Click **Add** again to add the second database host and enter the following information:

   - **Host Name**: OAMDBHOST2
   - **Instance Name**: oamdb2
   - **Port**: 1521

   Click **Next**.

8. In the Test Component Schema screen, the configuration wizard attempts to validate the data source.

   If the data source validation succeeds, click **Next**.

   If it fails, click **Previous**, correct the issue, and try again.

9. In the Select Optional Configuration screen, select:

   - **Administration Server**
   - **Managed Server Clusters and Machines**

   Click **Next**.

10. In the Customize Server and Cluster Configuration screen, select **Yes**, and click **Next**.

11. In the Configure the Administration Server screen, enter the following values:

    - **Name**: AdminServer

- **Listen Address**: OAMHOST1.example.com
- **Listen Port**: 7001
- **SSL listen port**: Not applicable
- **SSL enabled**: leave unchecked

Click **Next**.

12. On the Configure Managed Servers screen, create an entry for each OAMHOST in the topology, that is, one for the OAM Server running on OAMHOST1 and one for the OAM Server running on OAMHOST2.

    Select the OAM_SERVER entry and change the entry to the following values:

    - **Name**: WLS_OAM1
    - **Listen Address**: OAMHOST1.example.com
    - **Listen Port**: 14100

    For the second OAM_SERVER, click **Add** and supply the following information:

    - **Name**: WLS_OAM2
    - **Listen Address**: OAMHOST2.example.com
    - **Listen Port**: 14100

    Click **Next**.

13. In the Configure Clusters screen, create a cluster by clicking **Add**.

    Enter name: OAM_Cluster

    Leave all other fields at the default settings.

    Click **Next**.

14. On the Assign Servers to Clusters screen, associate the managed servers with the cluster, as follows:

    - Click the cluster name **OAM_Cluster** in the right window.
    - Click the managed server **WLS_OAM1**, then click the arrow to assign it to the cluster.
    - Repeat for managed server **WLS_OAM2**.

    Click **Next**.

15. On the Configure Machines screen, create a machine for each host in the topology. Click the UNIX tab if your hosts use a UNIX-based operating system. Otherwise, click the Machines tab. Supply the following information:

    - **Name**: Name of the host. The best practice is to use the DNS name (OAMHOST1)
    - **Node Manager Listen Address**: The DNS name of the machine (OAMHOST1.example.com)
    - **Node Manager Port**: A port for Node Manager to use.

    Repeat the steps for OAMHOST2:

    - **Name**: Name of the host. The best practice is to use the DNS name (OAMHOST2)

- **Node Manager Listen Address**: The DNS name of the machine (OAMHOST2.example.com)
- **Node Manager Port**: A port for Node Manager to use.

Click **Next**.

16. In the Assign Servers to Machines screen, indicate which managed servers will run on the machines just created.

    - Click the machine **OAMHOST1** in the right window.
    - Click the managed server **WLS_OAM1** in the left window.
    - Click the arrow to assign the managed server to the host **OAMHOST1**.
    - Click the machine **OAMHOST2** in the right window.
    - Click the managed server **WLS_OAM2** in the left window.
    - Click the arrow to assign the managed server to the host **OAMHOST2**.

    Click **Next**.

17. On the Configuration Summary screen, click **Create** to begin the creation process.

> **Note:** You cannot run the config.sh script twice to make configuration changes, you must use another tool such as using the MBeans Browser in Fusion Middleware Control

6.3.5 Configuring the Database Security Store

You must configure the database security store after you configure the domain but before you start the Administration Server. See Section 5.3.4, "Configuring the Database Security Store for the Domain" for more information.

6.3.6 Creating boot.properties for the Administration Server on OAMHOST1

This section describes how to create a `boot.properties` file for the Administration Server on OAMHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

To create the `boot.properties` file:

1. On OAMHOST1, go the following directory:

   ```
   MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
   ```

   For example:

   ```
   cd /u01/app/oracle/product/fmw/user_
   projects/domains/IDMDomain/servers/AdminServer/security
   ```

2. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

> **Note:** When you start the Administration Server, the username and password entries in the file get encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

3. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

4. Start the Administration Server on OAMHOST1 using the startWebLogic.sh script located under the `MW_HOME/user_projects/domains/domainName/bin` directory.

5. Validate that the changes were successful by opening a web browser and accessing the following pages:

   - WebLogic Server Administration Console at:

     ```
     http://oamhost1.example.com:7001/console
     ```

   - Oracle Enterprise Manager Fusion Middleware Control at:

     ```
     http://oamhost1.example.com:7001/em
     ```

   Log into these consoles using the `weblogic` user credentials.

6.3.7 Starting OAMHOST1

This section describes the steps for starting OAMHOST1.

6.3.7.1 Create the Node Manager Properties File on OAMHOST1

Before you can start managed servers from the console, you must create a Node Manager property file. You do this by running the script `setNMProps.sh`, which is located in the *MW_HOME*`/oracle_common/common/bin` directory. For example:

```
OAMHOST1> $MW_HOME/oracle_common/common/bin/setNMProps.sh
```

6.3.7.2 Start Node Manager

Start Node Manager by issuing the following command:

```
OAMHOST1> $MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh
```

6.3.7.3 Start Access Manager on OAMHOST1

To start Access Manager on OAMHOST1, follow these steps:

1. Log into the WebLogic Administration Console using this URL:

   ```
   http://oamhost1.example.com:7001/console
   ```

2. Supply the WebLogic administrator username and password.
3. Select **Environment - Servers** from the **Domain Structure** menu.
4. Click the Control tab.

5. Click the server **WLS_OAM1**.
6. Click **Start**.
7. Click **OK** to confirm that you want to start the server.

6.3.8 Validating OAMHOST1

To validate the implementation, connect to the Oracle Access Management Console at the following URL:

```
http://OAMHOST1.example.com:7001/oamconsole
```

The implementation is valid if the OAM Admin console login page opens and you can login using the WebLogic `administrator` account.

6.3.9 Configuring OAM on OAMHOST2

After the configuration succeeds on OAMHOST1, you can propagate it to OAMHOST2. You do this by packing the domain using the `pack` script on OAMHOST1, and unpacking the domain using the `unpack` script on OAMHOST2.

Both scripts reside in the *MW_HOME*`/oracle_common/common/bin` directory.

On OAMHOST1, enter:

```
pack.sh -domain=$MW_HOME/user_projects/domains/IDM_Domain \
    -template=/tmp/idm_domain.jar -template_name="OAM Domain" -managed=true
```

This creates a file called `idm_domain.jar` in the `/tmp` directory. Copy this file to OAMHOST2.

On OAMHOST2, enter:

```
unpack.sh -domain=$MW_HOME/user_projects/domains/IDM_Domain \
    -template=/tmp/idm_domain.jar
```

6.3.10 Starting OAMHOST2

This section describes the steps for starting OAMHOST2.

6.3.10.1 Create the Node Manager Properties File on OAMHOST2

Before you can start managed servers from the console, you must create a Node Manager property file. You do this by running the script `setNMProps.sh`, which is located in the *MW_HOME*`/oracle_common/common/bin` directory. For example:

```
OAMHOST1> $MW_HOME/oracle_common/common/bin/setNMProps.sh
```

6.3.10.2 Start Node Manager

Start Node Manager by issuing the following command:

```
OAMHOST2> $MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh
```

6.3.10.3 Start Access Manager on OAMHOST2

To start Access Manager on OAMHOST2, follow these steps:

1. Log into the WebLogic Administration Console using this URL:

```
http://oamhost2.example.com:7001/console
```

2. Supply the WebLogic administrator username and password.
3. Select **Environment - Servers** from the **Domain Structure** menu.
4. Click the Control tab.
5. Click the server **WLS_OAM2**.
6. Click **Start**.
7. Click **OK** to confirm that you want to start the server.

6.3.11 Validating OAMHOST2

Validate the implementation by connecting to the OAM server using the following URL:

```
http://OAMHOST2.example.com:14100/oam/server/logout
```

The implementation is valid if an OAM logout successful page opens.

6.3.12 Configure Access Manager to Work with Oracle HTTP Server

This section provides steps for configuring Access Manager to work with the Oracle HTTP Server.

6.3.12.1 Update Oracle HTTP Server Configuration

On WEBHOST1 and WEBHOST2, create a file named `oam.conf` in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

Create the file with the following lines:

```
NameVirtualHost *:7777
<VirtualHost *:7777>

    ServerName sso.example.com:7777
    ServerAdmin you@your.address
    RewriteEngine On
    RewriteOptions inherit

    <Location /oam>
        SetHandler weblogic-handler
        Debug ON
        WLLogFile /tmp/weblogic.log
        WLProxySSL ON
        WLProxySSLPassThrough ON
        WebLogicCluster oamhost1.example.com:14100,oamhost2.example.com:14100
    </Location>

    <Location /oamfed>
        SetHandler weblogic-handler
        Debug ON
        WLLogFile /tmp/weblogic.log
        WLProxySSL ON
        WLProxySSLPassThrough ON
        WebLogicCluster oamhost1.example.com:14100,oamhost2.example.com:14100
    </Location>

</VirtualHost>
```

6.3.12.2 Restart Oracle HTTP Server

Restart the Oracle HTTP Server on WEBHOST1 and WEBHOST2:

```
WEBHOST1>opmnctl stopall
WEBHOST1>opmnctl startall

WEBHOST2>opmnctl stopall
WEBHOST2>opmnctl startall
```

6.3.12.3 Make OAM Server Aware of the Load Balancer

By default, Access Manager sends requests to the login page located on the local server. In a high availability deployment, you must change this setup so that login page requests go to the load balancer.

To make Access Manager aware of the load balancer:

1. Log into the Oracle Access Management Console at this URL as the `weblogic` user:

   ```
   http://OAMHOST1.example.com:7001/oamconsole
   ```

2. Click on the **Configuration** tab.
3. Click the **Access Manager Settings** link.
4. Enter the following information:
   - OAM Server Host: sso.example.com
   - OAM Server Port: 7777
   - OAM Server Protocol: http
5. Click **Apply**.
6. Restart managed servers WLS_OAM1 and WLS_OAM2.

6.3.13 Configuring Access Manager to use an External LDAP Store

By default, Access Manager uses its own in built-in LDAP server. In a highly available configuration, it is recommended that an external LDAP directory be used as the directory store. This section describes how to set up an external LDAP store.

> **Note:** Oracle recommends that you back up the environment and LDAP store before performing the subsequent steps in this section.

6.3.13.1 Extending Directory Schema for Access Manager

Pre-configuring the Identity Store extends the schema in the backend directory regardless of directory type.

To extend the directory schema for Access Manager, perform the following tasks on OAMHOST1:

1. Set the Environment Variables: `MW_HOME`, `JAVA_HOME`, `IDM_HOME` and `ORACLE_HOME`.

   Set `IDM_HOME` to `IDM_ORACLE_HOME`

Set ORACLE_HOME to IAM_ORACLE_HOME

2. Create a properties file extend.props that contains the following:

```
IDSTORE_HOST : idstore.example.com
IDSTORE_PORT : 389
IDSTORE_BINDDN : cn=orcladmin
IDSTORE_USERNAMEATTRIBUTE: cn
IDSTORE_LOGINATTRIBUTE: uid
IDSTORE_USERSEARCHBASE:cn=Users,dc=example,dc=com
IDSTORE_GROUPSEARCHBASE: cn=Groups,dc=us,dc=oracle,dc=com
IDSTORE_SEARCHBASE: dc=example,dc=com
IDSTORE_SYSTEMIDBASE: cn=systemids,dc=example,dc=com
```

Where:

- IDSTORE_HOST and IDSTORE_PORT are, respectively, the host and port of your Identity Store directory. Specify the back-end directory here, rather than OVD.)
- IDSTORE_BINDDN Is an administrative user in the Identity Store Directory
- IDSTORE_USERSEARCHBASE is the location in your Identity Store where users are placed.
- IDSTORE_GROUPSEARCHBASE is the location in your Identity Store where groups are placed.
- IDSTORE_SEARCHBASE is the location in the directory where Users and Groups are stored.
- IDSTORE_SYSTEMIDBASE is the location in your directory where the Oracle Identity Manager reconciliation user are placed.
- IDSTORE_SYSTEMIDBASE is the location of a container in the directory where users can be placed when you do not want them in the main user container. This happens rarely but one example is the Oracle Identity Manager reconciliation user which is also used for the bind DN user in Oracle Virtual Directory adapters.

3. Configure the Identity Store by using the command idmConfigTool, which is located at: IAM_ORACLE_HOME/idmtools/bin.

The command syntax is:

```
idmConfigTool.sh -preConfigIDStore input_file=configfile
```

For example:

```
idmConfigTool.sh -preConfigIDStore input_file=extend.props
```

When the command runs, the system prompts you for the account password with which you are connecting to the Identity Store.

Sample command output:

```
Enter ID Store Bind DN password :
Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
INFO: -> LOADING:

/u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idm_idstore_groups_
template.ldif
Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
INFO: -> LOADING:
```

```
/u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idm_idstore_groups_acl_
template.ldif
Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
INFO: -> LOADING:

/u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/systemid_pwdpolicy.ldif
Apr 5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFileINFO: ->
LOADING:

/u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/idstore_tuning.ldifApr
5, 2011 3:39:25 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
INFO: -> LOADING:

/u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oid_schema_extn.ldif
The tool has completed its operation. Details have been logged to
automation.log
```

4. Check the log file for errors and warnings and correct them.

6.3.13.2 Create Users and Groups in LDAP

To add users that Access Manager requires to the Identity Store, follow these steps:

1. Set the Environment Variables MW_HOME, JAVA_HOME, IDM_HOME, and ORACLE_HOME.
   - Set IDM_HOME to IDM_ORACLE_HOME.
   - Set ORACLE_HOME to IAM_ORACLE_HOME.
2. Create a properties file oam.props that contains the following:

```
IDSTORE_HOST : idstore.example.com
IDSTORE_PORT : 389
IDSTORE_BINDDN : cn=orcladmin
IDSTORE_USERNAMEATTRIBUTE: cn
IDSTORE_LOGINATTRIBUTE: uid
IDSTORE_USERSEARCHBASE: cn=Users,dc=example,dc=com
IDSTORE_GROUPSEARCHBASE: cn=Groups,dc=example,dc=com
IDSTORE_SEARCHBASE: dc=example,dc=com
POLICYSTORE_SHARES_IDSTORE: true
OAM11G_IDSTORE_ROLE_SECURITY_ADMIN:OAMAdministrators
IDSTORE_OAMSOFTWAREUSER:oamLDAP
IDSTORE_OAMADMINUSER:oamadmin
IDSTORE_SYSTEMIDBASE:cn=systemids,dc=example,dc=com
```

   Where:

   - IDSTORE_HOST and IDSTORE_PORT are, respectively, the host and port of your Identity Store directory.
   - IDSTORE_BINDDN is an administrative user in the Identity Store Directory.
   - IDSTORE_USERSEARCHBASE is the location in the directory where Users are Stored.
   - IDSTORE_GROUPSEARCHBASE is the location in the directory where Groups are Stored.
   - IDSTORE_SEARCHBASE is the location in the directory where Users and Groups are stored.
   - POLICYSTORE_SHARES_IDSTORE is set to true if your Policy and Identity Stores are in the same directory. If not, it is set to false.

- IDSTORE_OAMADMINUSER is the name of the user you want to create as your Access Manager Administrator.
- IDSTORE_OAMSOFTWAREUSER is a user that gets created in LDAP that is used when Access Manager is running to connect to the LDAP server.

3. Configure the Identity Store using the command idmConfigTool which is located at IAM_ORACLE_HOME/idmtools/bin.

   The command syntax is:

   ```
   idmConfigTool.sh -prepareIDStore mode=OAM input_file=configfile
   ```

   For example:

   ```
   idmConfigTool.sh -prepareIDStore mode=OAM input_file=oam.props
   ```

   After the command runs, the system prompts you to enter the password for the account with which you are connecting to the ID Store.

   Sample command output:

   ```
   Enter ID Store Bind DN password :
   Apr 5, 2011 3:53:28 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

    /u01/app/oracle/product/fmw/IAM/oam/server/oim-intg/schema/OID_oblix_schema_
   add.ldif
   Apr 5, 2011 3:54:12 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/oam/server/oim-intg/schema/OID_oblix_schema_
   index_add.ldif
   Apr 5, 2011 3:55:10 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

    /u01/app/oracle/product/fmw/IAM/oam/server/oim-intg/schema/OID_oblix_pwd_
   schema_add.ldif
   Apr 5, 2011 3:55:11 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/oam/server/oim-intg/schema/OID_oim_pwd_schema_
   add.ldif
   *** Creation of Oblix Anonymous User ***
   Apr 5, 2011 3:55:11 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oam_10g_anonymous_user_
   template.ldif
   Enter User Password for oblixanonymous:
   Confirm User Password for oblixanonymous:
   Apr 5, 2011 3:55:53 AM oracle.ldap.util.LDIFLoader loadOneLdifFile
   INFO: -> LOADING:

   /u01/app/oracle/product/fmw/IAM/idmtools/templates/oid/oam_group_member_
   template.ldif
   The tool has completed its operation. Details have been logged to
   automation.log
   ```

4. Check the log file for any errors or warnings and correct them.

See Oracle Fusion Middleware Integration Overview for Oracle Identity Management Suite for more information about the idmConfigTool command.

6.3.13.3 Create a User Identity Store

To create a user identity store:

1. Go to the Oracle Access Management Console at the URL:

   ```
   http://adminvhn.example.com:7001/oamconsole
   ```

2. Log in using the WebLogic administration user.
3. Select **System Configuration** then **Data Sources**.
4. Select **User Identity Stores** then click **Add**. Enter the following information:
   - **Store Name**: LDAP_DIR
   - **Store Type**: OVD
   - **Description**: Enter a description of the Directory Store
   - **Enable SSL**: Select this if you communicate with your directory over SSL
   - **Location**: Enter the location, for example ovd.example.com:389
   - **Bind DN**: Enter the user permitted to search the LDAP store. For example, cn=orcladmin
   - **Password**: Enter the oracleadmin password
   - **User Name Attribute**: For example: uid
   - **User Search Base**: Enter the location of users in the LDAP store. For example, cn=Users,dc=example,dc=com
   - **Group Name Attribute**: For example: orclguid
   - **Group Search Base**: Enter the location of groups in the LDAP store. For example, cn=Groups,dc=example,dc=com
   - **OAM Administrator Role**: OAMAdministrators
5. Click **Apply**.
6. Click **Test Connection** to validate the connection to the LDAP server.

6.3.13.4 Set LDAP to System and Default Store

Now that you have defined the LDAP identity store, you must set it as the primary authentication store. To do this, follow these steps in the Oracle Access Management Console:

1. Click the **System Configuration** tab.
2. Select **Data Sources - User Identity Stores** from the navigation pane.
3. Click **LDAP_DIR**.
4. Select **Open** from the **Actions** menu.
5. Click **Set as Default Store**.
6. Click **Set as System Store**.
7. Click the Add **[+]** icon in **Access System Administrators**.
8. Enter **OAM*** in the search name field and click **Search**.
9. Select **OAMAdministrators** from the search results and click **Add Selected**.
10. Click **Apply**.

11. In the Validate System Administrator window, enter the username and password of the OAM administrator, for example, oamadmin.
12. Click **Validate**.
13. Test the connection by clicking **Test Connection**.

6.3.13.5 Set Authentication to Use External LDAP

By default, Access Manager uses the integrated LDAP store for user validation. You must update the LDAP authentication module so that it can validate users against the new external LDAP store.

To update the LDAP authentication module to use external LDAP:

1. Click the **System Configuration** tab.
2. Select **Access Manager Settings** - **Authentication Modules** - **LDAP Authentication Modules**.
3. Click **LDAP**.
4. Select **Open** from the **Actions** menu.
5. Set **User Identity Store** to `LDAP_DIR`.
6. Click **Apply**.
7. Restart the managed servers Admin Server, WLS_OAM1 and WLS_OAM2.

> **Note:** If you use `oamadmin` to manage OIF, you must add the OAM Administrator Role. For more information, see Section 6.3.13.3, "Create a User Identity Store."

6.3.14 Validating the Access Manager Configuration

Validate the configuration by logging into the Oracle Access Management Console at `http://oamhost1.example.com:7001/oamconsole` as `oamadmin`.

6.3.15 Configuring Oracle Coherence to Keep Configuration Files in Sync

In a highly available environment, Oracle uses Oracle Coherence to keep configuration files in sync. Oracle Coherence uses port 9095 by default, but this can be changed through the Oracle Access Management Console.

Log in to the console using the url *http://admin.example.com/oamconsole*, then follow these steps:

1. Click on the **System Configuration** tab.
2. Expand **Servers** in the navigator.
3. Double-click on the Managed Server whose port you wish to change.
4. Click on the **Coherence** tab.
5. Change the value of **Local Port** to the desired value.
6. Click **Apply**.
7. Restart the Administration Server and all the managed servers residing in the same cluster as the managed server that has been updated.

6.3.16 Scaling Up and Scaling Out the Access Manager Topology

This section describes how to scale up and scale out an Access Manager high availability topology. Perform a scale up operation to add a new Access Manager managed server instance is added to a node already running one or more server instances. Perform a scale out operation to add a new Access Manager managed server instance to a new node.

6.3.16.1 Scaling Up Access Manager

Scale up OAM as follows:

Log in to the Oracle WebLogic Server Administration Console at `http://oamhost1.example.com:7001/console`.

1. From the Domain Structure window of the Administration Console, expand the **Environment** node and then **Servers**. The Summary of Servers page appears.
2. In the Change Center, click **Lock & Edit**.
3. Select a server on the host you want to extend, for example: `WLS_OAM1`.
4. Click **Clone**.
5. Enter the following information:
   - **Server Name**: A new name for the server, for example: `WLS_OAM3`.
   - **Server Listen Address**: The name of the host on which the managed server will run.
   - **Server Listen Port**: The port the new managed server will use, this port must be unique within the host.
6. Click **OK**.
7. Click on the newly created server **WLS_OAM3**
8. Set the SSL listen port. This should be unique on the host that the managed server will be running on.
9. Click **Save**.
10. Disable hostname verification for the new managed server. Before starting and verifying the `WLS_OAM3` managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in `OAMHOST`*`n`*.

    If the source server from which the new one was cloned had already disabled hostname verification, these steps are not required, as the hostname verification settings were propagated to the cloned server. To disable hostname verification:

    a. In Oracle Enterprise Manager Fusion Middleware Control, select **Oracle WebLogic Server Administration Console**.

    b. Expand the **Environment** node in the Domain Structure window.

    c. Click **Servers**. Select **WLS_OAM3** in the Names column of the table.

    d. Click the **SSL** tab. Click **Advanced**.

    e. Set **Hostname Verification** to `None`. Click **Save**.
11. Click **Activate configuration** from the Change Center menu.

Register the new managed server with OAM. You must now configure the new managed server as an OAM server from the Oracle Access Management Console:

1. Log in to the Oracle Access Management Console as the `oamadmin` user at `http://oamhost1.example.com:7001/oamconsole`
2. Click the **System Configuration** tab. Click **Server Instances**.
3. Select **Create** from the Actions menu.
4. Enter the following information:
   - **Server Name**: `WLS_OAM3`
   - **Host**: Host that the server will run on
   - **Port**: Listen port that was assigned when the managed server was created
   - **OAM Proxy Port**: Port you want the OAM proxy to run on. This is unique for the host
   - **Proxy Server ID**: `AccessServerConfigProxy`
   - **Mode**: `Open`
5. Click **Coherence** tab.

   Set **Local Port** to a unique value on the host.

   Click **Apply**.
6. Click **Apply**.

You can now start the Access server. To use the Access server, you must inform any Webgates of its existence. You do that as follows:

1. Log in to the Oracle Access Management Console at `http://oamhost1.example.com:7001/oamconsole` as the `oamadmin` user.
2. Click the **System Configuration** tab.
3. From **Launch Pad**, under **Application Management**, click and open **SSO Agents**. Click Search on the SSO Agents page.
4. Double click the WebGate you want to change.
5. Add the new server to either the primary or secondary server list by clicking the Add + icon.
6. Select the server name from the list.
7. Click **Apply**

6.3.16.2 Scaling Out Access Manager

Scale out is very similar to scale up, but first requires the software to be installed on the new node.

1. Install Oracle WebLogic Server on the new host as described in Section 6.3.3, "Installing Oracle WebLogic Server."
2. Install Identity Management components on the new host. See Section 6.3.4, "Installing and Configuring the Access Manager Application Tier."
3. Log in to the Oracle WebLogic Server Administration Console at `http://oamhost1.example.com:7001/oamconsole`.
4. From the Domain Structure window of the Administration Console, expand the **Environment** node and then **Servers**. The Summary of Servers page appears.

5. In the Change Center, click **Lock & Edit**.
6. Select a server on the host you want to extend, for example: `WLS_OAM1`.
7. Click **Clone**.
8. Enter the following information:
   - **Server Name**: A new name for the server, for example: `WLS_OAM3`.
   - **Server Listen Address**: The name of the host on which the managed server will run.
   - **Server Listen Port**: The port the new managed server will use. This port must be unique within the host.
9. Click **OK**.
10. Click the newly created server **WLS_OAM3**.
11. Set the SSL listen port. This should be unique on the host that the managed server will run on.
12. Click **Save**.
13. Disable hostname verification for the new managed server. Before starting and verifying the `WLS_OAM3` managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in `OAMHOST`*n*.

    If the source server from which the new one was cloned had already disabled hostname verification, these steps are not required, as the hostname verification settings was propagated to the cloned server. To disable hostname verification:

    a. In Oracle Enterprise Manager Fusion Middleware Control, select **Oracle WebLogic Server Administration Console**.

    b. Expand the **Environment** node in the Domain Structure pane.

    c. Click **Servers**. The Summary of Servers page appears.

    d. Select **WLS_OAM3** in the Names column of the table. The Settings page for server appears.

    e. Click the **SSL** tab. Click **Advanced**.

    f. Set **Hostname Verification** to **None**.

    g. Click **Save**.
14. Click **Activate Configuration** from the Change Center menu.
15. Pack the domain on `OAMHOST1` using the command:

    ```
    pack.sh -domain=ORACLE_BASE/admin/IDM_Domain/aserver/IDM_Domain -template
    =/tmp/idm_domain.jar -template_name="OAM Domain" -managed=true
    ```

    The `pack.sh` script is located in *`MW_HOME`*`/oracle_common/common/bin`.
16. Unpack the domain on the new host using the command:

    ```
    unpack.sh -domain=ORACLE_BASE/admin/IDM_Domain/mserver/IDM_Domain
    -template=/tmp/idm_domain.jar -template_name="OAM Domain" -app_dir=ORACLE_
    BASE/admin/IDM_Domain/mserver/applications
    ```

The `unpack.sh` script is located in *`MW_HOME`*`/oracle_common/common/bin`.

17. Before you can start managed servers from the console, you must create a node manager properties file on `OAMHOST3`. You do this by running the script `setNMProps.sh`, which is located in *`MW_HOME`*`/oracle_common/common/bin`. Type:

```
MW_HOME/oracle_common/common/bin/setNMProps.sh
```

Register the new managed server with OAM. You must now configure the new managed server as an OAM server from the Oracle Access Management Console:

1. Log in to the Oracle Access Management Console at `http://oamhost1.example.com:7001/oamconsole` as the `oamadmin` user.
2. Click the **System Configuration** tab. Click **Server Instances**.
3. Select **Create** from the Actions menu.
4. Enter the following information:
   - **Server Name**: `WLS_OAM3`
   - **Host**: Host that the server will be running on, `OAMHOST3`.
   - **Port**: Listen port that was assigned when the managed server was created.
   - **OAM Proxy Port**: Port you want the OAM proxy to run on. This is unique for the host.
   - **Proxy Server ID:** `AccessServerConfigProxy`
   - **Mode**: `Open`
5. Click **Apply**.

Start the Access Server. To use the server, you must inform any Webgates of its existence:

1. Log in to the Oracle Access Management Console at `http://oamhost1.example.com:7001/oamconsole` as the `oamadmin` user.
2. Click the **System Configuration** tab.
3. Expand **Agents** -> **OAM Agents** ->**10g Agents**.
4. Double click the WebGate you want to change.
5. Add the new server to either the primary or secondary server list by clicking the Add [+] icon.
6. Select the server name from the list.
7. Click **Apply**.

Update the Web Tier. Now that the new managed server has been created and started, the web tier will start to direct requests to it. Best practice, however, is to inform the web server that the new managed server has been created.

You do this by updating the file `OAM.conf` on each of the web tiers. This file resides in the directory: *`ORACLE_INSTANCE`*`/config/OHS/`*`component name`*`/moduleconf`.

Add the new server to the `WebLogicCluster` directive in the file, for example, change:

```
<Location /OAM_admin>
    SetHandler weblogic-handler
    WebLogicCluster
 OAMhost1.example.com:14200,OAMhost2.example.com:14200
</Location>
```

to:

```
<Location /OAM_admin>
    SetHandler weblogic-handler
    WebLogicCluster
 OAMhost1.example.com:14200,OAMhost2.example.com:14200,OAMhsot3.example.com:14300
</Location>
```

7

Configuring High Availability for Oracle Adaptive Access Manager Components

This chapter introduces Oracle Adaptive Access Manager and describes how to design and deploy a high availability environment for Oracle Adaptive Access Manager.

Oracle Adaptive Access Manager is built on a J2EE-based, multi-tier deployment architecture that separates the platform's presentation, business logic, and data tiers. Because of this separation of tiers, Oracle Adaptive Access Manager can rapidly scale with the performance needs of the customer. The architecture can leverage the most flexible and supported cross-platform J2EE services available: a combination of Java, XML and object technologies. This architecture makes Oracle Adaptive Access Manager a scalable, fault-tolerant solution.

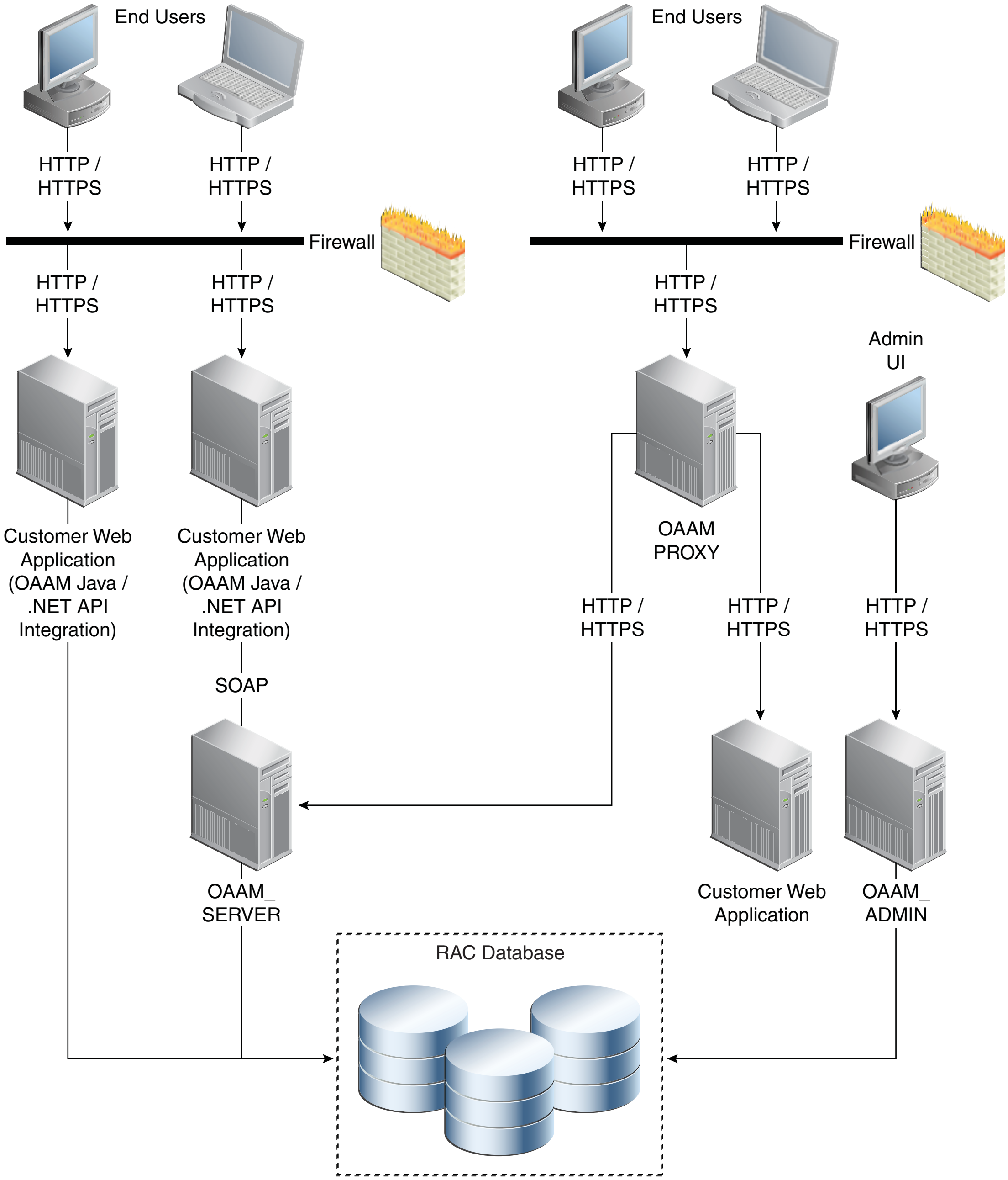
This chapter includes the following topics:

- Section 7.1, "Oracle Adaptive Access Manager Component Architecture"
- Section 7.2, "Oracle Adaptive Access Manager High Availability Concepts"
- Section 7.3, "Oracle Adaptive Access Manager High Availability Configuration Steps"

7.1 Oracle Adaptive Access Manager Component Architecture

Figure 7–1 shows the single instance architecture for Oracle Adaptive Access Manager.

Figure 7–1 Oracle Adaptive Access Manager Single Instance Architecture



In the Oracle Adaptive Access Manager single instance architecture, end users access customer web applications, which communicate with the OAAM Server application and its policies using SOAP. Alternately, an OAAM proxy can be set up so that end users communicate with that machine, which then communicates with the OAAM_ Server application using HTTP(S). Authorized end users can access the customer web application. The OAAM_ADMIN component is used for administration and configuration of the OAAM_SERVER application. The administrator responsible for administering and configuring the OAAM_Server application uses a web browser to access the OAAM_ADMIN application. An Oracle RAC database holds policy and configuration information.

7.1.1 Oracle Adaptive Access Manager Component Characteristics

Oracle Adaptive Access Manager consists of the following two components:

- OAAM_ADMIN: This component is used for administration and configuration of OAAM_SERVER application. This component is developed using the Oracle JAVA ADF Framework the Identity Management shell and deployed as Web applications in a J2EE container. It is packaged as an EAR file.
- OAAM_SERVER: This component contains the OAAM Admin and OAAM Server sub-components within a single web application. The OAAM_SERVER component is packaged as an EAR file and is composed of Servlets and JSPs in addition to Java classes. The subcomponents of OAAM_SERVER are described below by layer:
  - Presentation Layer: typically a Web application serving JSPs, servlets, and so on. The presentation layer provides the strong authenticator functionality; it uses the interfaces provided by the business layer (SOAP or Java native) to access its services.
  - Business Logic Layer: contains the core application logic that implements the risk analyzing engine. This layer provides Java and SOAP interfaces for the presentation layer. When the Java interface is used, the business logic layer and presentation layer can be part of a single web application. With the SOAP interface, these layers are deployed as different applications.
  - Data Access Layer: contains data access components to connect to the supported relational databases. Oracle Adaptive Access Manager uses Oracle's TopLink, which provides a powerful and flexible framework for storing Java objects in a relational database.
- OAAM_OFFLINE: This component is a standalone application. OAAM Offline has its own database. This database has an identical schema to that of the OAAM Online version. It is used to load customer data to perform risk analysis and tune rules. OAAM Offline can support both login and transaction data.

You can also use the following components in an Oracle Adaptive Access Manager 11*g* deployment:

- Fusion Middleware Control / Enterprise Manager Grid Control: Oracle Adaptive Access Manager integrates with the Enterprise Manager Grid Control to display performance metrics and deployment topology. Oracle Adaptive Access Manager uses DMS and Discovery Mbeans to integrate with Enterprise Manager. Enterprise Manager is also used to enhance component tracing and configure auditing.

  Enterprise Manager can also be used to view log files for each Managed Server in the domain and increase the tracing to Debug, Trace, and Info levels.
- Data repositories: Oracle Adaptive Access Manager uses the RDBMS database as its data store. Oracle Adaptive Access Manager supports and works on the following database technologies:
  - Oracle Real Application Clusters
  - Oracle Data Guard
  - Replication Streams
  - Database Partitioning

  Oracle Adaptive Access Manager uses RCU to creates its schema in the database.

7.1.1.1 Oracle Adaptive Access Manager State Information

The OAAM_Server component includes the OAAM Server subcomponent and the OAAM Admin subcomponent.

- OAAM Server is a stateful application that stores the state in HTTP session.
- OAAM Admin is a stateful application that stores its session information in the database.

The OAAM_Admin component is an ADF and Identity Management UI shell-based application. It is a stateless application, and its application state is maintained by the ADF framework.

The OAAM_OFFLINE component is a stateful application that stores the state in HTTP session.

7.1.1.2 Oracle Adaptive Access Manager Runtime Processes

You can perform the following runtime tasks using the Oracle Adaptive Access Manager Administration Console:

- Customer Care application tasks
- System configuration tasks involving policies, groups, and properties
- Viewing session data information
- Viewing the System Statistics dashboard

For example, you can perform the following administration flows:

1. Recent user query:
   a. View recent logins and session details.
   b. Perform a query.
   c. Click **Session Details**.
   d. Log out.
2. Manual CSR and Agent Case creation:
   a. To reset customer care, log in.
   b. Create a case.
   c. Reset the customer.
   d. Log out.

You can also perform runtime processing with the Oracle Adaptive Access Manager Server.

7.1.1.3 Oracle Adaptive Access Manager Process Lifecycle

The following runtime processing occurs with Oracle Adaptive Access Manager Server:

1. Oracle Adaptive Access Manager is deployed and integrated with the customer's application.
2. The user will access the customer's application and enter user credentials.
3. Based on the system and rules configured in OAAM, different login flows will be presented, for example:

   User Registration: Registration Flows

   a. Flow R1: Login (New User), enter password, personalize device, skip Questions Registration, log out.
   b. Flow R2: Login, enter password, skip Registration, log out.
   c. Flow R3: Login, enter password, personalize device, continue Questions Registration, log out.
   d. Flow R4: Login, enter password, personalize device, continue Questions Registration, enter invalid answers, validation, log out
   e. Flow R5: Login (New Device and New User), enter password, personalize device, continue Questions Registration, log out.

   Login Flow:

   a. Flow L1: Login, enter wrong password, Login screen.
   b. Flow L2: Login, enter correct password, Challenge On may be presented, answer correctly, logged in.
   c. Flow L3: Login, enter correct password, Challenge On presented, answer incorrectly, Challenge On presented again, answer correctly, logged in.
   d. Flow L4: Login, enter correct password, Challenge On presented, answer incorrectly, Challenge On presented again, answer incorrectly, Challenge On presented again, answer correctly, logged in.
   e. Flow L5: Login, enter correct password, Challenge On presented, answer incorrectly, Challenge On presented again, answer incorrectly, login blocked.
   f. Flow L6: Login, enter correct password, login blocked (login screen).
   g. Flow L7: Login, enter correct password, logged in.
   h. Flow LNU1Login as a new user: Login, enter correct password, logged in.
   i. Flow LND1: Existing user, login with a new device, enter correct password, logged in.
   j. Flow LNUD1: New user, login with a new device, enter correct password, logged in.

   In Session Transaction Flow: Login, enter password, create transaction, update transaction, log out.

4. OAAM tracks and registers the following data elements:
   a. User login
   b. User names
   c. Devices (cookies, browser headers, and flash data supplied)
   d. IP addresses
   e. Transaction data
5. OAAM will trigger the appropriate policy based on login behavior.

As J2EE applications, you can start the Oracle Adaptive Access Manager Server and Administration Console using the WebLogic Server user interface and command line tools.

The Oracle Adaptive Access Manager Server supports a health check request (a ping request over HTTP) that can be used by a load balancer for death detection.

Oracle Adaptive Access Manager is instrumented for server side metrics (using DMS) and this information is published to the Administration Console. When DMS metrics collection is enabled, monitoring the agent and server component metrics can serve as a proxy for component monitoring. In addition, Oracle Adaptive Access Manager supports fine-grained real time activity tracing, which can also serve as a proxy for component monitoring.

7.1.1.4 Oracle Adaptive Access Manager Configuration Artifacts

Oracle Adaptive Access Manager stores its configuration information in the database. To change these configuration properties, use the Oracle Adaptive Access Manager Administration Console.

Oracle Adaptive Access Manager does not store any configuration information on the file system or in the exploded EAR file.

7.1.1.5 Oracle Adaptive Access Manager Deployment Artifacts

Oracle Adaptive Access Manager supports the no-stage mode of deployment staging. That is, all deployment files are local.

7.1.1.6 Oracle Adaptive Access Manager External Dependencies

Oracle Adaptive Access Manager has an external dependency on the RDBMS database, where it stores its configuration information.

Oracle Adaptive Access Manager uses WebLogic Server multi data sources and Gridlink data source for Oracle RAC databases.

Oracle Adaptive Access Manager uses the standard Oracle TopLink object caching mechanism.

Oracle Adaptive Access Manager follows standard session object serialization to maintain the persistent state of an object.

Oracle Adaptive Access Manager is not dependent on any hostname, IP address, or port. It will work on a container-specific port or hostname.

7.1.1.7 Oracle Adaptive Access Manager Log File Location

Oracle Adaptive Access Manager is a J2EE application deployed on WebLogic Server. All log messages are logged in the server log file of the WebLogic Server that the application is deployed on. The default location of the server log is:

```
WL_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

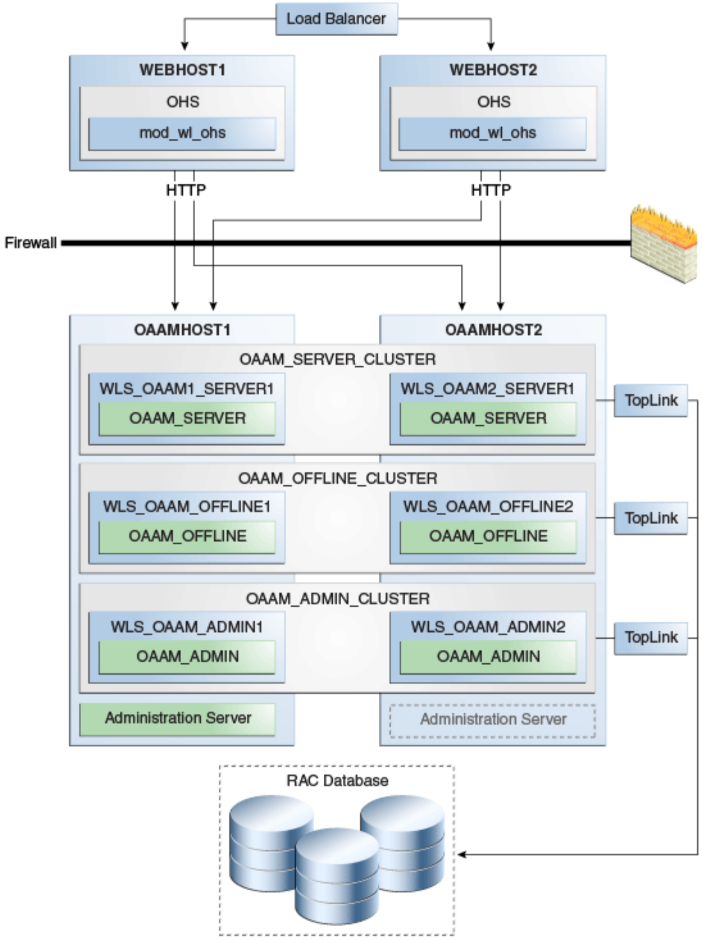
7.2 Oracle Adaptive Access Manager High Availability Concepts

This section provides conceptual information about using Oracle Adaptive Access Manager in a high availability two-node cluster.

7.2.1 Oracle Adaptive Access Manager High Availability Architecture

Figure 7–2 shows the Oracle Adaptive Access Manager high availability architecture:

Figure 7–2 Oracle Adaptive Access Manager High Availability Architecture



In Figure 7–2, the load balancer receives incoming requests and routes them to WEBHOST1 or WEBHOST2 in the web tier. These hosts have Oracle HTTP Server installed. Oracle HTTP Server, using the WebLogic plugin `mod_wl_ohs.conf`, then forwards requests to the WebLogic managed servers on OAAMHOST1 and OAAMHOST2.

OAAMHOST1 and OAAMHOST2 contain managed servers that host the Oracle Adaptive Access Manager Server, the Oracle Adaptive Access Manager Admin, and the Oracle Adaptive Access Manager Offline applications. The managed servers are configured in a cluster which enables the Access Servers to work in an active-active manner.

The WebLogic Administration Server runs on OAAMHOST1 and contains the WebLogic Administration Console and the Oracle Enterprise Manager Fusion Middleware Control. The Administration Server can be configured to run in active-passive mode on OAAMHOST2, which means that if OAAMHOST1 becomes unavailable, then Administration Server can be manually started on OAAMHOST2.

An Oracle RAC database provides high availability in the data tier.

Only one OAAM Server cluster, one OAAM Offline cluster, and one OAAM Administration cluster are supported per WebLogic Server domain. In addition, Oracle Adaptive Access Manager-related clusters cannot span WebLogic Server domains.

A single instance Oracle Adaptive Access Manager deployment satisfies the following high availability requirements:

- Load handling
- External connection management and monitoring
- Recovery
- Fault containment
- Fault diagnostics
- Oracle Adaptive Access Manager Admin / Server offline

A multiple instance Oracle Adaptive Access Manager deployment satisfies the following additional high availability requirements:

- Redundancy
- Client connection failover / continuity
- Client load balancing
- State management

Oracle recommends using an external load balancing router for inbound HTTP connections.

Web sessions open persistent TCP connections to the Oracle Adaptive Access Manager Administration Console and servers. This requires that load balancing router and firewall connection timeouts are sufficiently large to avoid premature termination of TCP connections.

7.2.1.1 Starting and Stopping the Cluster

Each Oracle Adaptive Access Manager Administration Console and Server instance is a peer of other instances. Because all initialization happens before the Server is ready to receive requests and because of built in throttling capabilities, surge conditions are dealt with gracefully without any significant impact of the performance characteristics of the Oracle Adaptive Access Manager 11*g*R2 Access Server.

When the cluster stop, new requests are denied and existing requests can complete before the Oracle Adaptive Access Manager Administration Console and Server application shuts down. If a forced shutdown occurs, Oracle Adaptive Access Manager 11*g*R1 recovers any corrupted or invalid data that the shutdown causes.

Oracle Adaptive Access Manager components are pure J2EE applications and do not have any start or stop functionality of their own. Instead, they rely on container-specific startup and shutdown functionality.

Oracle Adaptive Access Manager components deploy to WebLogic Server Managed Server nodes. You restart the components using Node Manager.

7.2.1.2 Cluster-Wide Configuration Changes

Since Oracle Adaptive Access Manager stores the entire configuration in database, the propagation of configuration changes to all the cluster members transparent. All Oracle Adaptive Access Manager components are notified of change events from the internal layer, which are then taken up by the components. To ensure atomicity of the change, Oracle Adaptive Access Manager components reload their entire configuration every time a change happens.

Oracle Adaptive Access Manager configuration applies to every instance in a cluster.

Adding and removing Oracle Adaptive Access Manager Administration Console and Server instances is transparent to other Oracle Adaptive Access Manager instances in the cluster.

An Oracle Adaptive Access Manager cluster can have any number of instances. There is no restriction on the number of instances per cluster.

Online application redeployment does not cause any problems.

7.2.2 Protection from Failures and Expected Behaviors

The following features protect an Oracle Adaptive Access Manager high availability configuration from failure:

- Session state for the cluster is maintained in memory, which provides replication and failover for all session state information.
- Oracle Adaptive Access Manager Servers support a heartbeat check - a ping request over HTTP. In addition, the WebLogic Node Manager on the Managed Server can monitor the application and restart it if it is not running. Restarting an Oracle Adaptive Access Manager Server has no impact on any other running components or cluster members.
- When a WebLogic Server node fails, external connection failover is based on the configuration and is based on the retry timeout as well as the number of retries.
- When the load balancing router or proxy server detects a WebLogic Server node failure, subsequent client connections are routed to the active instance, which picks up the session state for further processing.
- An Oracle Adaptive Access Manager session does not have a direct impact on an Oracle RAC database node failure, because WebLogic Server maintains the state of its database connections.

7.3 Oracle Adaptive Access Manager High Availability Configuration Steps

This section provides high-level instructions for setting up a maximum high availability deployment for Oracle Adaptive Access Manager. This deployment includes two Oracle HTTP Servers, which distribute requests to two OAAM servers. These OAAM servers interact with an Oracle Real Application Clusters (Oracle RAC) database. If any single component fails, then the remaining components will continue to function.

This section includes the following topics:

- Section 7.3.1, "Prerequisites for Oracle Adaptive Access Manager Configuration"
- Section 7.3.2, "Run the Repository Creation Utility to Create the OAAM Schemas in a Database"
- Section 7.3.3, "Installing Oracle WebLogic Server"
- Section 7.3.4, "Installing and Configuring the Oracle Adaptive Access Manager Application Tier"
- Section 7.3.5, "Configuring the Database Security Store for the Domain"
- Section 7.3.6, "Creating boot.properties for the Administration Server on OAAMHOST1"
- Section 7.3.7, "Create the Oracle Adaptive Access Manager Administration User"
- Section 7.3.8, "Start OAAMHOST1"
- Section 7.3.9, "Validating OAAMHOST1"

- Section 7.3.10, "Configure Oracle Adaptive Access Manager on OAAMHOST2"
- Section 7.3.11, "Start OAAMHOST2"
- Section 7.3.12, "Validating OAAMHOST2"
- Section 7.3.13, "Configure Oracle Adaptive Access Manager to Work with Oracle HTTP Server"
- Section 7.3.14, "Validating the Oracle Adaptive Access Manager Configuration"
- Section 7.3.15, "Scaling Up and Scaling Out the Oracle Adaptive Access Manager Topology"

7.3.1 Prerequisites for Oracle Adaptive Access Manager Configuration

Before you configure Oracle Adaptive Access Manager for high availability, you must:

- Run the Repository Creation Utility to create the OAAM schemas in a database. See Section 7.3.2, "Run the Repository Creation Utility to Create the OAAM Schemas in a Database".
- Install Oracle WebLogic Server on OAAMHOST1 and OAAMHOST2. See Section 7.3.3, "Installing Oracle WebLogic Server"
- Install and configure the Oracle Identity Management executables on OAAMHOST1 and OAAMHOST2. See Follow the steps in Section 7.3.4, "Installing and Configuring the Oracle Adaptive Access Manager Application Tier".

7.3.2 Run the Repository Creation Utility to Create the OAAM Schemas in a Database

The schemas you create depend on the products you want to install and configure. Use the Repository Creation Utility (RCU) that is version compatible with the product you are installing. See the *Oracle Fusion Middleware Installation Planning Guide for Oracle Identity and Access Management* and *Oracle Fusion Middleware Repository Creation Utility User's Guide* to run RCU.

7.3.3 Installing Oracle WebLogic Server

Before you install the Oracle WebLogic Server, ensure that your machines meet the system, patch, kernel, and other requirements as specified in *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

To install Oracle WebLogic Server on OAAMHOST1 and OAAMHOST2, start the installer then complete these steps:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, select **Create a New Middleware Home**.

   For Middleware Home Directory, enter:

   ORACLE_BASE`/product/fmw`

   > **Note:** *ORACLE_BASE* is the base directory under which Oracle products are installed. The recommended value is `/u01/app/oracle`.

   Click **Next**.

3. On the Register for Security Updates screen, enter your contact information so that you can be notified of security updates.

   Click **Next**.

4. On the Choose Install Type screen, select **Custom**.

   Click **Next**.

5. On the Choose Products and Components screen, select only **Oracle JRockit SDK**, and click **Next**.

6. On the Choose Product Installation Directories screen, accept the directory *ORACLE_BASE*`/product/fmw/wlserver_10.3`.

   Click **Next**.

7. On the Installation Summary screen, click **Next**.

8. On the Installation Complete screen, deselect **Run Quickstart**.

   Click **Done**.

7.3.4 Installing and Configuring the Oracle Adaptive Access Manager Application Tier

This section describes how to install Oracle Fusion Middleware components on OAAMHOST1 and OAAMHOST2.

7.3.4.1 Install Oracle Fusion Middleware for Identity Management

Perform these steps to install Oracle Fusion Middleware components on OAAMHOST1 and OAAMHOST2.

If the `/etc/oraInst.loc` file exists, verify that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory. If the `/etc/oraInst.loc` file does not exist, you can skip this step.

Start the installer for Oracle Fusion Middleware as follows:

```
OAAMHOST1> runInstaller
```

When the installer prompts you for a JRE/JDK location, enter the Oracle SDK location created in the WebLogic Server installation, for example, *ORACLE_BASE*`/product/fmw/jrockit_160_`*<version>*, then proceed as follows:

1. On the Welcome screen, click **Next**.

2. On the Prerequisite Checks screen, verify that the checks complete successfully, then click **Next**.

3. On the Specify Installation Location screen, enter the following values:

   - Oracle Middleware Home: Select the previously installed Middleware home from the list for `MW_HOME`, for example:

     ```
     /u01/app/oracle/product/fmw
     ```

   - Oracle Home Directory:

     – Enter `iam` as the Oracle Home directory name when installing the Oracle Identity and Access Management Suite in the `IAM_ORACLE_HOME`.

   Click **Next**.

4. On the Installation Summary screen, click **Install**.

5. On the Installation Complete screen, click **Finish**.

7.3.4.1.1 Configure Oracle Access Manager on OAAMHOST1 This section creates the domain on OAAMHOST1.

Start the configuration wizard by running the command:

ORACLE_HOME`/oracle_common/common/bin/config.sh`

Then continue with the following steps:

1. In the Welcome screen, select **Create a New WebLogic Domain** then click **Next**.
2. In the Select Domain Source Screen:

   Select **Generate a domain configured automatically to support the following products**:

   And select the following products:

   - **Oracle Enterprise Manager**
   - **Oracle JRF** (selected by default)
   - **Oracle Adaptive Access Manager - Server**
   - **Oracle Adaptive Access Manager Admin Server**

   Click **Next**.
3. In the Specify Domain and Location screen enter:

   - **Domain name**: IDM_Domain
   - **Domain Directory**: Accept the default.
   - **Application Directory**: Accept the default.

   Click **Next**.
4. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator and click **Next**.
5. In the Configure Server Start Mode and JDK screen, make the following selections:

   - **WebLogic Domain Startup Mode**: Select **Production Mode**.
   - **JDK Selection**: Select **JROCKIT SDK1.6.0_**<*version*>.
6. In the Configure JDBC Component Schema screen, select all of the data sources, then select **Configure selected data sources as RAC multi data sources**.

   Click **Next**.
7. In the Configure RAC Multi Data Source Component Schema screen, select the first data source, **OAAM Admin Server**, and enter the following:

   - **Data source**: OAAM Admin Server
   - **Service Name**: `oaam.example.com`
   - **User Name**: OAAM_OAAM (assuming OAAM was used as the RCU prefix)
   - **Password**: The password for above account

   In the top right box, click **Add** to add an Oracle RAC host. Enter the following information:

   - **Host Name**: OAAMDBHOST1
   - **Instance Name**: oaamdb1
   - **Port**: 1521

Click **Add** again to add the second database host and enter the following information:

- **Host Name**: OAAMDBHOST2
- **Instance Name**: oaamdb2
- **Port**: 1521

Deselect this data source. Select the next data source, **OAAM Admin MDS Schema**, and enter the following information:

- **Data Source**: OAAM Admin MDS Schema
- **Service Name**: `oaam.example.com`
- **User Name**: OAAM_MDS (assuming OAAM was used as the RCU prefix)
- **Password**: Password for the OAAM_MDS account.

In the top right box, click **Add** to add an Oracle RAC host. Enter the following information:

- **Host Name**: OAAMDBHOST1
- **Instance Name**: oaamdb1
- **Port**: 1521

Click **Add** again to add the second database host and enter the following information:

- **Host Name**: OAAMDBHOST2
- **Instance Name**: oaamdb2
- **Port**: 1521

Deselect this data source. Select the next data source, **OAAM Server**, and enter the following information:

- **Data Source**: OAAM Server
- **Service Name**: `oaam.example.com`
- **User Name**: OAAM_OAAM (assuming OAAM was used as the RCU prefix)
- **Password**: Password for the OAAM_OAAM account.

In the top right box, click **Add** to add an Oracle RAC host. Enter the following information:

- **Host Name**: OAAMDBHOST1
- **Instance Name**: oaamdb1
- **Port**: 1521

Click **Add** again to add the second database host and enter the following information:

- **Host Name**: OAAMDBHOST2
- **Instance Name**: oaamdb2
- **Port**: 1521

Deselect this data source. Select the next data source, **OWSM MDS Schema**, and enter the following information:

- **Data Source**: OWSM MDS Schema

- **Service Name**: `oaam.example.com`
- **User Name**: OAAM_MDS (assuming OAAM was used as the RCU prefix)
- **Password**: Password for the OAAM_MDS account.

In the top right box, click **Add** to add an Oracle RAC host. Enter the following information:

- **Host Name**: INFRADBHOST1
- **Instance Name**: oaamdb1
- **Port**: 1521

Click **Add** again to add the second database host and enter the following information:

- **Host Name**: INFRADBHOST2
- **Instance Name**: oaamdb2
- **Port**: 1521

Deselect this data source.

Click **Next**.

8. In the Test Component Schema screen, the configuration wizard attempts to validate the data source.

   If the data source validation succeeds, click **Next**.

   If it fails, click **Previous**, correct the issue, and try again.

9. In the Select Optional Configuration screen, select:

   - **Administration Server**
   - **Managed Server Clusters and Machines**

   Click **Next**.

10. In the Customize Server and Cluster Configuration screen, select **Yes**, and click **Next**.

11. In the Configure the Administration Server screen, enter the following values:

    - **Name**: AdminServer
    - **Listen Address**: Enter the virtual hostname of the Administration Server, for example: OAAMHOST1.example.com
    - **Listen Port**: 7001
    - **SSL listen port**: Not applicable
    - **SSL enabled**: leave unchecked

    Click **Next**.

12. On the Configure Managed Servers screen, create three entries for each OAAMHOST in the topology: one for OAAM Server, OAAM Admin Server, and OAAM Offline server.

    Select the OAAM_SERVER entry and change the entry to the following values:

    - **Name**: WLS_OAAM1_SERVER1
    - **Listen Address**: OAAMHOST1.*example*.com

- **Listen Port**: 14300
- **SSL Listen Port**: 14301
- **SSL Enabled**: Selected

For the second OAAM_SERVER, click **Add** and supply the following information:

- **Name**: WLS_OAAM2_SERVER2
- **Listen Address**: OAAMHOST2.*example*.com
- **Listen Port**: 14300
- **SSL Listen Port**: 14301
- **SSL Enabled**: Selected

Select the OAAM Offline entry and change the entry to the following values:

- **Name**: WLS_OAAM_OFFLINE1
- **Listen Address**: OAAMHOST1.*example*.com
- **Listen Port**: 14400
- **SSL Listen Port**: 14401
- **SSL Enabled**: Selected

For the second OAAM Offline, click **Add** and supply the following information:

- **Name**: WLS_OAAM_OFFLINE2
- **Listen Address**: OAAMHOST2.*example*.com
- **Listen Port**: 14400
- **SSL Listen Port**: 14401
- **SSL Enabled**: Selected

Select the OAAM_ADMIN_SERVER entry and change the entry to the following values:

- **Name**: WLS_OAAM_ADMIN1
- **Listen Address**: OAAMHOST1.*example*.com
- **Listen Port**: 14200
- **SSL Listen Port**: 14201
- **SSL Enabled**: Selected

For the second OAAM_ADMIN_SERVER, click **Add** and supply the following information:

- **Name**: WLS_OAAM_ADMIN2
- **Listen Address**: OAAMHOST2.example.com
- **Listen Port**: 14200
- **SSL Listen Port**: 14201
- **SSL Enabled**: Selected

Leave all other fields at the default settings.

Click **Next**.

13. In the Configure Clusters screen, create a cluster by clicking **Add**.

Enter name: OAAM_SERVER_CLUSTER

Create a second cluster by clicking **Add**.

Enter name: OAAM_Offline_Cluster

Create a third cluster by clicking **Add**.

Enter name: OAAM_Admin_Cluster

Leave all other fields at the default settings.

Click **Next**.

14. On the Assign Servers to Clusters screen, associate the managed servers with the cluster, as follows:

    - Click the cluster name **OAAM_SERVER_CLUSTER** in the right window.
    - Click the managed server **WLS_OAAM1_SERVER1**, then click the arrow to assign it to the cluster.
    - Repeat for managed server **WLS_OAAM2_SERVER2**.

    Then:

    - Click the cluster name **OAAM_OFFLINE_Cluster** in the right window.
    - Click the managed server **WLS_OAAM_OFFLINE1**, then click the arrow to assign it to the cluster.
    - Repeat for managed server **WLS_OAAM_OFFLINE2**.

    Then:

    - Click the cluster name **OAAM_Admin_Cluster** in the right window.
    - Click the managed server **WLS_OAAM_ADMIN1**, then click the arrow to assign it to the cluster.
    - Repeat for managed server **WLS_OAAM_ADMIN2**.

    Click **Next**.

15. On the Configure Machines screen, create a machine for each host in the topology. Click the UNIX tab if your hosts use a UNIX-based operating system. Otherwise, click the Machines tab. Supply the following information:

    - **Name**: Name of the host. The best practice is to use the DNS name (OAAMHOST1)
    - **Node Manager Listen Address**: The DNS name of the machine (OAAMHOST1.example.com)
    - **Node Manager Port**: A port for Node Manager to use.

    Click **Next**.

16. In the Assign Servers to Machines screen, indicate which managed servers will run on the machines just created.

    For OAAMHOST1:

    - Click the machine **OAAMHOST1** in the right window.
    - Click the managed server **WLS_OAAM1_SERVER1** in the left window.
    - Click the arrow to assign the managed server to the host **OAAMHOST1**.
    - Click the managed server **WLS_OAAM_OFFLINE1** in the left window.

- Click the arrow to assign the managed server to the host **OAAMHOST1**.
- Click the managed server **WLS_OAAM_ADMIN1** in the left window.
- Click the arrow to assign the managed server to the host **OAAMHOST1**.

For OAAMHOST2:

- Click the machine **OAAMHOST2** in the right window.
- Click the managed server **WLS_OAAM2_SERVER2** in the left window.
- Click the arrow to assign the managed server to the host **OAAMHOST2**.
- Click the managed server **WLS_OAAM_OFFLINE2** in the left window.
- Click the arrow to assign the managed server to the host **OAAMHOST2**.
- Click the managed server **WLS_OAAM_ADMIN2** in the left window.
- Click the arrow to assign the managed server to the host **OAAMHOST2**.

Click **Next**.

17. On the Configuration Summary screen, click **Create** to begin the creation process.

7.3.5 Configuring the Database Security Store for the Domain

You must configure the database security store after you configure the domain but before you start the Administration Server. See Section 7.3.5, "Configuring the Database Security Store for the Domain" for more information.

7.3.6 Creating boot.properties for the Administration Server on OAAMHOST1

This section describes how to create a `boot.properties` file for the Administration Server on OAAMHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

To create the `boot.properties` file:

1. Start the Admin Server.
2. On OAAMHOST1, go the following directory:

   ```
   MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
   ```

   For example:

   ```
   cd /u01/app/oracle/product/fmw/user_
   projects/domains/IDMDomain/servers/AdminServer/security
   ```

3. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

> **Note:** When you start the Administration Server, the username and password entries in the file get encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

4. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

5. Start the Administration Server on OAAMHOST1 using the startWebLogic.sh script located under the `MW_HOME/user_projects/domains/domainName/bin` directory.

6. Validate that the changes were successful by opening a web browser and accessing the following pages:

   - WebLogic Server Administration Console at:

     ```
     http://oaamhost1.example.com:7001/console
     ```

   - Oracle Enterprise Manager Fusion Middleware Control at:

     ```
     http://oaamhost1.example.com:7001/em
     ```

   Log into these consoles using the `weblogic` user credentials.

7.3.7 Create the Oracle Adaptive Access Manager Administration User

To create an Administrative user, which will be used to access the OAAM console:

1. Log into the WebLogic Administration Console using this URL:

   ```
   http://oaamhost1.example.com:7001/console
   ```

2. From the **Domain Structure** menu, click **Security Realms** and then **myrealm**.
3. Click the Users and Groups tab.
4. Click the Users sub tab.
5. Click **New**.
6. Enter these values:

   - **Name**: Name for the user, for example: `oaamadmin`
   - **Provider**: Default Authenticator
   - **Password/Confirmation**: Password for user

   Click **OK**.

7. After the user is created, click the user name.
8. Click on the Groups sub tab.
9. Assign the user to the following groups:

   - OAAMCSRGroup
   - OAAMCSRManagerGroup
   - OAAMInvestigatorGroup
   - OAAMInvestigationManagerGroup
   - OAAMRuleAdministratorGroup
   - OAAMEnvAdminGroup
   - OAAMSOAPServicesGroup

Click **Save**.

7.3.8 Start OAAMHOST1

This section describes the steps for starting OAAMHOST1.

7.3.8.1 Create the Node Manager Properties File on OAAMHOST1

Before you can start managed servers from the console, you must create a Node Manager property file. You do this by running the script `setNMProps.sh`, which is located in the *MW_HOME*`/oracle_common/common/bin` directory. For example:

```
OAAMHOST1> $MW_HOME/oracle_common/common/bin/setNMProps.sh
```

7.3.8.2 Start Node Manager

Start Node Manager by issuing the following command:

```
OAAMHOST1> $MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh
```

7.3.8.3 Start Oracle Adaptive Access Manager on OAAMHOST1

To start Oracle Adaptive Access Manager on OAAMHOST1, follow these steps:

1. Log into the WebLogic Administration Console using this URL:

   ```
   http://oaamhost1.example.com:7001/console
   ```

2. Supply the WebLogic administrator username and password.
3. Select **Environment - Servers** from the **Domain Structure** menu.
4. Click the Control tab.
5. Click the servers **WLS_OAAM1_SERVER1**, **WLS_OAAM_OFFLINE1**, and **WLS_OAAM_ADMIN1**.
6. Click **Start**.
7. Click **OK** to confirm that you want to start the servers.

7.3.9 Validating OAAMHOST1

Validate the implementation by connecting to the OAAM Administration Server at the following URL:

```
http://OAAMHOST1.example.com:14200/oaam_admin
```

The implementation is valid if the OAAM Admin console login page is displayed and you can login using the `oaamadmin` account you created in Section 7.3.7, "Create the Oracle Adaptive Access Manager Administration User."

Validate the implementation by connecting to the OAAM Server at:

```
http://OAAMHOST1.example.com:14300/oaam_server
```

The implementation is valid if the OAAM Server login page appears.

7.3.10 Configure Oracle Adaptive Access Manager on OAAMHOST2

Once the configuration has succeeded on OAAMHOST1, you can propagate it to OAAMHOST2. You do this by packing the domain using the pack script on OAAMHOST1, and unpacking the domain using the unpack script on OAAMHOST2.

Both scripts reside in the *MW_HOME*/oracle_common/common/bin directory.

On OAAMHOST1, enter:

```
pack.sh -domain=$MW_HOME/user_projects/domains/IDM_Domain \
    -template=/tmp/idm_domain.jar -template_name="OAAM Domain" -managed=true
```

This creates a file called idm_domain.jar in the /tmp directory. Copy this file to OAAMHOST2.

On OAAMHOST2, enter:

```
unpack.sh -domain=$MW_HOME/user_projects/domains/IDM_Domain \
    -template=/tmp/idm_domain.jar
```

7.3.11 Start OAAMHOST2

Now you will start OAAMHOST2.

This section describes the steps for starting OAAMHOST2.

7.3.11.1 Create the Node Manager Properties File on OAAMHOST2

Before you can start managed servers from the console, you must create a Node Manager property file. You do this by running the script setNMProps.sh, which is located in the *MW_HOME*/oracle_common/common/bin directory. For example:

```
OAAMHOST1> $MW_HOME/oracle_common/common/bin/setNMProps.sh
```

7.3.11.2 Start Node Manager

Start Node Manager by issuing the following command:

```
OAAMHOST2> $MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh
```

7.3.11.3 Start Oracle Adaptive Access Manager on OAAMHOST2

To start Oracle Adaptive Access Manager on OAAMHOST2, follow these steps:

1. Log into the WebLogic Administration Console using this URL:

   ```
   http://oaamhost2.example.com:7001/console
   ```

2. Supply the WebLogic administrator username and password.
3. Select **Environment - Servers** from the **Domain Structure** menu.
4. Click the Control tab.
5. Click the servers **WLS_OAAM2_SERVER2**, **WLS_OAAM_OFFLINE2**,and **WLS_OAAM_ADMIN2**.
6. Click **Start**.
7. Click **OK** to confirm that you want to start the servers.

7.3.12 Validating OAAMHOST2

Validate the implementation by connecting to the OAAM Administration Server at the following URL:

```
http://OAAMHOST2.example.com:14200/oaam_admin
```

The implementation is valid if OAAM Admin console login page is displayed and you can login using the oaamadmin account you created in Section 7.3.7, "Create the Oracle Adaptive Access Manager Administration User."

Validate the implementation by connecting to the OAAM Server at:

```
http://OAAMHOST2.example.com:14400/oaam_server
```

The implementation is valid if the OAAM Server login page is displayed.

7.3.13 Configure Oracle Adaptive Access Manager to Work with Oracle HTTP Server

This section provides steps for configuring Oracle Adaptive Access Manager to work with the Oracle HTTP Server.

7.3.13.1 Update Oracle HTTP Server Configuration

On WEBHOST1 and WEBHOST2, create a file named `oaam.conf` in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

Create the file with the following lines:

```
NameVirtualHost *:7777
<VirtualHost *:7777>

    ServerName oaam.example.com:7777
    ServerAdmin you@your.address
    RewriteEngine On
    RewriteOptions inherit

    <Location /oaam_server>
       SetHandler weblogic-handler
       WebLogicCluster oaamhost1.example.com:14300,oaamhost2.example.com:14300
    </Location>

    <Location /oaam_offline>
       SetHandler weblogic-handler
       WebLogicCluster oaamhost1.example.com:14400,oaamhost2.example.com:14400
       WebLogicPort 7001
    </Location>

    <Location /oaam_admin>
       SetHandler weblogic-handler
       WebLogicCluster oaamhost1.example.com:14200,oaamhost2.example.com:14200
       WebLogicPort 7001
    </Location>

</VirtualHost>
```

7.3.13.2 Restart Oracle HTTP Server

Restart the Oracle HTTP Server on WEBHOST1 and WEBHOST2:

```
WEBHOST1>opmnctl stopall
WEBHOST1>opmnctl startall

WEBHOST2>opmnctl stopall
WEBHOST2>opmnctl startall
```

7.3.13.3 Change Host Assertion in WebLogic

Because the Oracle HTTP Server acts as a proxy for WebLogic, by default certain CGI environment variables are not passed through to WebLogic. These include the host and port. You must tell WebLogic that it is using a virtual site name and port so that it can generate internal URLs appropriately.

To do this, log into the WebLogic Administration Console at:

```
http://oaamhost1.example.com:7001/console
```

Then perform these steps:

1. Select **Clusters** from the home page, or select **Environment > Clusters** from the **Domain Structure** menu.
2. Click **Lock and Edit** in the Change Center window to enable editing.
3. Click the Cluster Name (**OAAM_SERVER_CLUSTER**).
4. In the General tab, set WebLogic Plug in to **Enabled** by checking the box in the Advanced Properties section.
5. Click **Save**.
6. Select HTTP and enter the following values:
   - **Frontend Host**: `oaam.example.com`
   - **Frontend HTTP Port**: 7777
   - **Frontend HTTPS Port**: Set to SSL port on the load balancer or the Oracle HTTP Server SSL port if using SSL communication.

   This ensures that any HTTPS URLs created from within WebLogic are directed to port 443 or 80 on the load balancer.
7. Click **Save**.
8. Select **Clusters** from the home page, or select **Environment > Clusters** from the **Domain Structure** menu.
9. Click the Cluster Name (**oaam_admin_cluster**).
10. In the General tab, set WebLogic Plug in to **Enabled** by checking the box in the Advanced Properties section.
11. Click **Save**.
12. Select HTTP and enter the following values:
    - **Frontend Host**: `oaam.example.com`
    - **Frontend HTTP Port**: 7777
    - **Frontend HTTPS Port**: Set to SSL port on the load balancer or the Oracle HTTP Server SSL port if using SSL communication.
13. Click **Save**.
14. Click **Activate Changes** in the Change Center window to save the changes.

15. Restart managed servers WLS_OAAM1_SERVER1, WLS_OAAM2_SERVER2, WLS_OAAM_OFFLINE1, WLS_OAAM_OFFLINE2, WLS_OAAM_ADMIN1 and WLS_OAAM_ADMIN2 as follows:

    a. Log into the WebLogic Administration Console using this URL:

    ```
    http://oaamhost1.example.com:7001/console
    ```

    b. Supply the WebLogic administrator username and password.

    c. Select **Environment - Servers** from the **Domain Structure** menu.

    d. Click the Control tab.

    e. Click the servers **WLS_OAAM1_SERVER1**, **WLS_OAAM2_SERVER2**, **WLS_OAAM_OFFLINE1**, **WLS_OAAM_OFFLINE2**, **WLS_OAAM_ADMIN1**, and **WLS_OAAM_ADMIN2**.

    f. Click **Shutdown - Force shutdown now**.

    g. Click **Yes** to confirm that you want to stop the servers.

    h. Select **Environment - Servers** from the **Domain Structure** menu.

    i. Click the Control tab.

    j. Click the servers **WLS_OAAM1_SERVER1**, **WLS_OAAM2_SERVER2**, **WLS_OAAM_OFFLINE1**, **WLS_OAAM_OFFLINE2**, **WLS_OAAM_ADMIN1**, and **WLS_OAAM_ADMIN2**.

    k. Click **Start**.

    l. Click **Yes** to confirm that you want to start the servers.

16. Restart the Administration Server.

7.3.14 Validating the Oracle Adaptive Access Manager Configuration

Log into the Oracle Adaptive Access Manager Administration Console at `http://oaam.example.com:7777/oaam_admin` using the `oaamadmin` account you created.

Log into the Oracle Adaptive Access Manager server at `http://oaam.example.com:7777/oaam_offline` using the `oaamadmin` account and the password test.

Also, log into the Oracle Adaptive Access Manager server at `http://oaam.example.com:7777/oaam_server` using the `oaamadmin` account and the password test.

Complete the steps that the 7.7 Post-Installation Steps section of the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* describes.

7.3.15 Scaling Up and Scaling Out the Oracle Adaptive Access Manager Topology

This section describes how to scale up and scale out an Oracle Adaptive Access Manager high availability topology. Perform a scale up operation to add a new Oracle Adaptive Access Manager managed server instance is added to a node already running one or more server instances. Perform a scale out operation to add a new Oracle Adaptive Access Manager managed server instance to a new node.

7.3.15.1 Scaling Up Oracle Adaptive Access Manager

To scale up OAAM, use the same procedure for both the OAAM server and the OAAM Administration Server.

Log in to the Oracle WebLogic Server console at: `http://oaamhost1.example.com:7001/console`. Then proceed as follows:

1. From the Domain Structure window of the Oracle WebLogic Server Administration Console, expand the **Environment** node and then **Servers**. The Summary of Servers page appears.
2. Click **Lock & Edit** from the Change Center menu.
3. Select an existing server on the host that you want to extend, for example: `WLS_OAAM1_SERVER1` or `WLS_OAAM_ADMIN1`.
4. Click **Clone**.
5. Enter the following information:
   - **Server Name**: A new name for the server, for example: `WLS_OAAM3`.
   - **Server Listen Address**: The name of the host on which the managed server will run.
   - **Server Listen Port**: The port the new managed server will use. This port must be unique within the host.
6. Click **OK**.
7. Click the newly-created server **WLS_OAAM3**.
8. Set the SSL listen port. This should be unique on the host that the managed server will be running on.
9. Click **Save**.
10. Disable hostname verification for the new managed server. Before starting and verifying the `WLS_OAAM3` managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in `OAAMHOST`*n*.

    If the source server from which the new one was cloned had already disabled hostname verification, these steps are not required, as the hostname verification settings were propagated to the cloned server. To disable hostname verification:

    a. In the Oracle Fusion Middleware Enterprise Manager Console, select **Oracle WebLogic Server Administration Console**.

    b. Expand the **Environment** node in the Domain Structure pane.

    c. Click **Servers**. The Summary of Servers page appears.

    d. Select **WLS_OAAM3** in the Names column of the table. The Settings page for server appears.

    e. Click the **SSL** tab.

    f. Click **Advanced**.

    g. Set **Hostname Verification** to `None`.

    h. Click **Save**.
11. Click **Activate configuration** from the Change Center menu.

7.3.15.2 Scaling Out Oracle Adaptive Access Manager

Scale out is very similar to scale up, but first requires the software to be installed on the new node. Proceed as follows:

1. Install Oracle WebLogic Server on the new host as described in Section 5.3.1.1.1, "Installing Oracle WebLogic Server."
2. Install Oracle Fusion Middleware Identity Management components on the new host. See Section 7.3.4, "Installing and Configuring the Oracle Adaptive Access Manager Application Tier."
3. Log in to the WebLogic console at `http://oaamhost1.example.com:7001/console`.
4. From the Domain Structure pane of the Administration Console, expand the **Environment** node and then **Servers**. The Summary of Servers page appears.
5. Click **Lock & Edit** from the Change Center menu.
6. Select an existing server on the host you want to extend, for example: `WLS_OAAM1_SERVER1` or `WLS_OAAM_ADMIN1`.
7. Click **Clone**.
8. Enter the following information:
   - **Server Name**: A new name for the server, for example: `WLS_OAAM3`
   - **Server Listen Address**: The name of the host on which the managed server will run.
   - **Server Listen Port**: The port the new managed server will use. This port must be unique within the host.
9. Click **OK**.
10. Click the newly-created server **WLS_OAAM3**.
11. Set the SSL listen port. This should be unique on the host that the managed server will be running on.
12. Click **Save**.
13. Disable hostname verification for the new managed server. Before starting and verifying the `WLS_OAAM3` managed server, you must disable hostname verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in `OAAMHOST`*n*.

    If the source server from which the new one was cloned had already disabled hostname verification, these steps are not required, as the hostname verification settings were propagated to the cloned server. To disable hostname verification:

    a. In the Oracle Fusion Middleware Enterprise Manager Console, select **Oracle WebLogic Server Administration Console**.

    b. Expand the **Environment** node in the Domain Structure pane.

    c. Click **Servers**. The Summary of Servers page appears.

    d. Select **WLS_OAAM3** in the Names column of the table. The Settings page for server appears.

    e. Click the **SSL** tab.

    f. Click **Advanced**.

    g. Set **Hostname Verification** to `None`.

h. Click **Save**.

14. Click **Activate configuration** from the Change Center menu.

15. Pack the domain on OAAMHOST1 using the command:

    ```
    pack.sh -domain=ORACLE_BASE/admin/IDM_Domain/aserver/IDM_Domain -template
    =/tmp/idm_domain.jar -template_name="OAAM Domain" -managed=true
    ```

    The pack.sh script is located in *MW_HOME*/oracle_common/common/bin.

16. Unpack the domain on the new host using the command:

    ```
    unpack.sh -domain=ORACLE_BASE/admin/IDM_Domain/mserver/IDM_Domain
    -template=/tmp/idm_domain.jar -template_name="OAAM Domain" -app_dir=ORACLE_
    BASE/admin/IDM_Domain/mserver/applications
    ```

    The unpack.sh script is located in *MW_HOME*/oracle_common/common/bin.

17. Before you can start managed servers from the console, you must create a node manager properties file on OAAMHOST2 by running the script setNMProps.sh. The setNMProps.sh script is located in *MW_HOME*/oracle_common/common/bin. Type:

    ```
    OAAMHOST2> $MW_HOME/oracle_common/common/bin/setNMProps.sh
    ```

18. Start Node Manager and the new managed server on the new host

19. Now that the new managed server has been created and started, the web tier will start to direct requests to it. Best practice, however, is to inform the web server that the new managed server has been created.

    You do this by updating the file oaam.conf on each of the web tiers. This file resides in the directory: *ORACLE_INSTANCE*/config/OHS/*component_name*/moduleconf.

    Add the new server to the WebLogicCluster directive in the file. For example, change:

    ```
    <Location /oaam_admin>
        SetHandler weblogic-handler
        WebLogicCluster oaamhost1.example.com:14200,oaamhost2.example.com:14200
    </Location>
    ```

    to:

    ```
    <Location /oaam_admin>
        SetHandler weblogic-handler
        WebLogicCluster
    oaamhost1.example.com:14200,oaamhost2.example.com:14200,oaamhost3.example.com:1
    4300
    </Location>
    ```

8

Configuring High Availability for Oracle Access Management Security Token Service

This chapter describes Oracle Access Management Security Token Service high availability.

Security Token Service is a shared Web Service (JAX-WS) that provides a standards-based consolidated mechanism of trust brokerage between different identity domains and infrastructure tiers. Security Token Service brokers trust between a Web Service Consumer (WSC) and a Web Service Provider (WSP) and provides security token lifecycle management services to providers and consumers. Security Token Service can help simplify the effort needed to bridge access to various systems using a standardized set of interfaces.

A Security Token Service high availability deployment depends on Oracle Access Management Access Manager; the Access Manager application contains the Security Token Service server runtime. A Security Token Service high availability deployment cannot occur independent of Access Manager. Access Manager and Security Token Service are bundled together in the OAM J2EE Application EAR file, installed together, and deployed on the same managed server in a WebLogic domain. Security Token Service is also integrated with the Oracle Access Management Console.

For more details on administering and configuring Security Token Service, see one of the following topics in *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management*.

For more details on administering and configuring Security Token Service, see one of the following topics in *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management*.

- Security Token Service Key Terms and Concepts
- About Security Token Service
- About Security Token Service Deployments
- About Security Token Service Administration
- Security Token Service Implementation Scenarios
- Managing Security Token Service Settings and Set Up
- Managing Security Token Service Certificates and Keys
- Managing Templates, Endpoints, and Policies
- Managing Token Service Partners and Partner Profiles

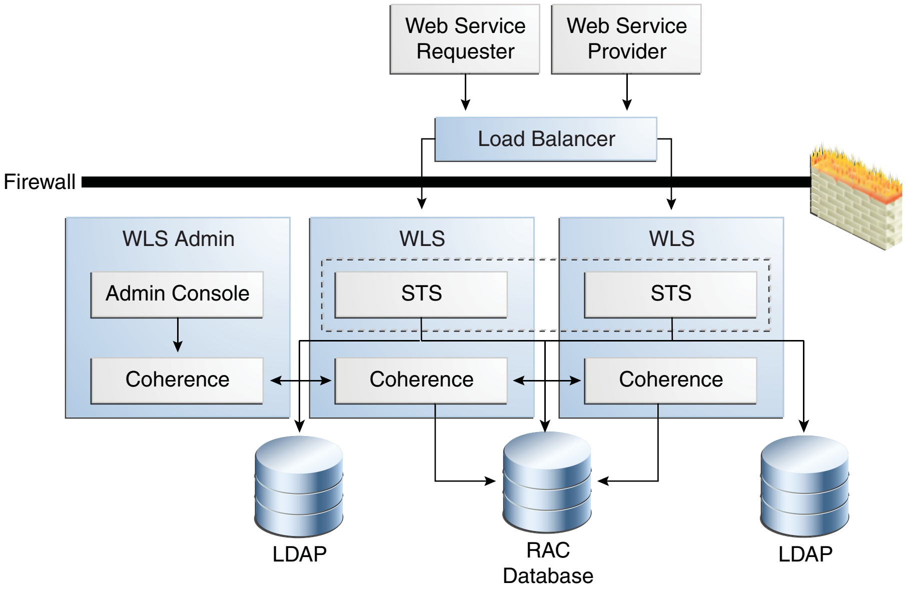
For information on patching, see Migrating Oracle Access Manager 11.1.1.3.0 to 11.1.1.6.0

This section includes the following topics:

8.1 Security Token Service High Availability Architecture

The following figure shows Security Token Service in a high availability architecture:

Figure 8–1 Security Token Service High Availability Architecture



This figure shows a two-node deployment of Access Manager/Security Token Service components. This section provides details about an Security Token Service high availability deployment. For more details about the overall Access Manager high availability architecture, see Section 6.2.1, "Access Manager High Availability Architecture".

Security Token Service is the server side component that implements the WS-Trust protocol support.

The load balancer receives token requests and routes them to the Security Token Service (STS).

The RAC Database is the same database that Access Manager uses as the Coherence backend store.

The Oracle Access Management Console is a J2EE application that provides services to manage the Security Token Service deployment. As part of the OAM deployment, Administration Console must deploy to the Weblogic AdminServer.

In Security Token Service, each WebLogic Server domain supports one Security Token Service cluster. OAM/Security Token Service clusters cannot span WebLogic Server domains.

Security Token Service is primarily based on the OASIS WS-Trust protocol. However, Security Token Service delegates the processing of other WS-* protocols in the SOAP message to the JAX-WS stack.

Oracle recommends using external load balancers for inbound HTTP/SOAP connections. Outbound external connections to LDAP servers are load balanced with support for connection failover.

8.1.1 Clients and Client Connections

Web Service clients that implement the WS-Trust protocol interact with Security Token Service to issue or validate tokens. Clients designed to interact with an STS server, such as OWSM Client, as part of a Web Service call to a Relying Party can invoke Security Token Service.

The client connection process is as follows:

1. The Web Service client sends a SOAP message over http or https.

   The WSS protocol protects the message. The payload contains a WS-Trust request (RST) indicating the operation to perform, which kind of token to issue or validate, and additional information about the token characteristics.

2. The server processes the request and sends a response over the same channel the server received it on.

   The WSS protocol protects the message. The payload contains a WS-Trust response (RSTRC) if the processing was successful or a SOAP fault if an error occurs.

8.1.2 Cluster Wide Configuration Changes

Each Security Token Service Access Server instance is a peer of other instances with no inter-instance communication. Because all initialization happens before the Server is ready to receive requests combined with built in throttling capabilities, the WebLogic Server handles surge conditions gracefully without any significant effect on Security Token Service Access Server performance characteristics.

When the cluster stops, the Security Token Service denies new requests and permits existing requests to complete before the Access Server application shuts down. If a forced shutdown occurs, Security Token Service can recover for any corrupted/invalid data that the shutdown causes.

Propagation of configuration changes to all the cluster members is based on a distribution mechanism that leverages the Coherence distributed object cache. The coherence layer notifies all Security Token Service components of change events. The components then uptake these change events. Access Manager components reload their entire configuration every time a change happens.

Note: The addition/removal of Access Server instance(s) is transparent to other Security Token Service instances in the cluster. Verify that removing a specific Security Token Service server does not affect the load.

8.2 Security Token Service Component Characteristics

This section describes Security Token Service component characteristics and includes the following topics:

- Section 8.2.1, "Security Token Service Component Lifecycle"
- Section 8.2.2, "Runtime Processes"
- Section 8.2.3, "Configuration Artifacts"
- Section 8.2.4, "External Dependencies"

8.2.1 Security Token Service Component Lifecycle

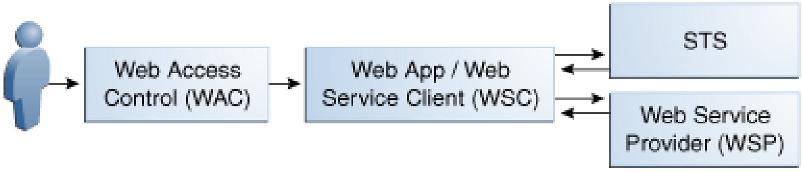
On startup, the Access Manager/Security Token Service Server initializes connections to the backend repositories. If the repository is not reachable, the Access Manager/Security Token Service server retries the connections to the repositories using a timeout that grows exponentially with a configurable upper limit.

The Access Manager/Security Token Service Server provides continuity of service based on locally cached data if the backend connections go down. Service continues until the caches grow stale or the backend connections come up again.

8.2.2 Runtime Processes

The following graphic shows the Security Token Service runtime process.

Figure 8–2 Security Token Service Runtime Process


Web Access Control (WAC)
Web App / Web Service Client (WSC)
STS
Web Service Provider (WSP)

The Security Token Service runtime process works as described below:

1. A Web Service Consumer (WSC) sends a Web Services-Trust Request Security Token (RST)) message for a security token type that the WSP requires. Authentication of the client occurs by using the transport layer authentication, or by binding the WSS Token to the RST message.
2. The Security Token Service (STS) validates the RST message, authenticates the request, then authorizes the requested operation.
3. The appropriate security token is generated in accordance with the metadata that the RST message specifies. For the policy driven exchange use-case, the STS looks up the appropriate token generation policy to generate the appropriate security token.
4. STS generates an RST message that contains the generated security token; it sends the message to the WSC as a response.

> **Note:** WSP validation of the security token depends on the token type. When STS acts as a trust intermediary only, validation is performed against the underlying security infrastructure, such as Kerberos.

8.2.2.1 Starting and Stopping Security Token Service

Because they are J2EE applications, you can start the Access Server (where Security Token Service is deployed) and the Administration Console from the user interface and Command Line tool that the Application Server provides.

8.2.2.2 J2EE Components and Subcomponents

J2EE Components and sub-components include the following:

- STS - An event based design pattern that implements the core Security Token Service 11g-PS1. It is packaged as a WAR application in the Access Manager EAR file and comprises a WS Provider Servlet and Java classes. The STS Web Application is bound to the /sts root path
- Admin Console - A stand-alone console based on ADF/IDM Shell, and packaged as an .EAR file.
- JMX Mbeans - Packaged with the Access Server package. Config Mbeans are packaged as standalone JAR files.
- WSLT Command - Consists of Java classes that are in the Access Manager/Security Token Service package.
- OWSM Agent - Web Service interceptor providing support for WSS protocol, part of JRF.
- ORAProvider - JRF Web Service Provider

8.2.2.3 Session State Information

Security Token Service is a stateless J2EE application with the exception of the Nonce caching for Username Tokens, where Security Token Service keeps track of presented username tokens when the nonce is present, to prevent replay attacks.

8.2.3 Configuration Artifacts

Access Manager and Security Token Service are built together and use the same modules for configuration, logging, and other processes. The Security Token Service configuration artifacts include the following files.

- DOMAIN_HOME/config/fmwconfig/oam-config.xml — Configuration file, which contains instance-specific information.
- DOMAIN_HOME/config/fmwconfig/oam-policy.xml — Present only when OES Micro SM is not being used.
- DOMAIN_HOME/config/fmwconfig/servers/instanceName/logging.xml — Logging config
- DOMAIN_HOME/config/fmwconfig/cwallet.sso — stores passwords that are used to connect to identity stores, database, and other entities. This is not for end user passwords.
- DOMAIN_HOME/config/fmwconfig/.oamkeystore — keystore containing keys and certificates Access Manager/Security Token Service owns

- DOMAIN_HOME/config/fmwconfig/amtruststore — keystore containing the trust anchors used for X509 cert validation
- DOMAIN_HOME/config/fmwconfig/amcrl.jar — zip file containing CRLs used for certificate revocation
- DOMAIN_HOME/config/fmwconfig/default-keystore.jks — OWSM keystore used to store keys and certificates used by the OWSM Agent, as well as trusted anchors used to validate X.509 certificates for WSS operations
- DOMAIN_HOME/config/fmwconfig/.cohstore.jks - trust store that Coherence SSL communication uses

8.2.4 External Dependencies

Security Token Service has external dependencies on the:

- LDAP based Identity Store
  - User Identity Repository
  - LDAP access abstracted by User/Role API.

    Note: Access Manager always connects to one Identity store, which can be a physical server or a load balancer IP. If the primary is down, Access Manager reconnects and expects the load balancer to connect it to the secondary.

- OCSP Responder Service
  - Real-time X.509 Certification Validation
- RDBMS Policy Store/Coherence Store
  - Policy (Authentication and Authorization) Repository
  - RDBMS access abstracted by the OAM policy engine
  - OPSS Security Store

8.3 Security Token Service High Availability Configuration Steps

Security Token Service High Availability is configured as part of Access Manager. All Security Token Service system configuration is done using the Oracle Access Management Console. See Section 6.3, "Access Manager High Availability Configuration Steps" for more information.

8.4 Validating Security Token Service High Availability

You can verify that Security Token Service endpoints are up and running on the different Security Token Service servers. To do so, access the WSDL document of an Security Token Service endpoint directly:
http(s)://[*hostname:port*]/sts/[*ENDPOINT*]/WSDL

Replace [*ENDPOINT*] with the existing published endpoint.

8.5 Security Token Service Failover and Expected Behavior

This section describes Security Token Service failover characteristics in a high availability environment.

Access Manager Access Servers support a heartbeat check--a ping request over HTTP. In addition, the WebLogic Node Manager on the Managed Server can monitor the application and restart it if necessary.

If a WebLogic Server node fails, external connection failover is based on the configuration, the retry timeout, and the number of retries. Access Manager Agent-Access Server failover is based on a timeout.

When the load balancing router or proxy server detects a WebLogic Server node failure, subsequent client connections route to the active instance, which picks up the session state from the Coherence Distributed Object Cache and continues processing.

Front Channel HTTP bindings use a load balancing router for failover.

When it receives an exception from another service, Access Manager retries external connection requests. The number of retries is configurable and includes a `no retries` option.

See the following topics for more information:

- Section 8.5.1, "Death Detection and Restart"
- Section 8.5.2, "Node Failure"

8.5.1 Death Detection and Restart

Access Manager/Security Token Service Access Servers support a heartbeat check in the form of a ping request sent over HTTP. Also, the WebLogic Node Manager on the managed server can monitor the application and restart if the event isn't running. Restarting an Access Manager Access Server does not affect any other cluster components or members.

8.5.2 Node Failure

External Connection failover is based on the configuration, retry timeout, and the number of retries. The load balancer or Proxy Server detects node failure and subsequent client connections are routed to the active instance, which picks up the session state from the Coherence DOC and continues with the processing.

8.6 Disabling and Enabling Security Token Service

Security Token Service is enabled by default. To disable Security Token Service, you use the Oracle Access Management Console. See Enabling or Disabling Available Services in the *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management*.

8.7 Troubleshooting Security Token Service

Security Token Service logs are logged to the Managed Servers log files. However, you can edit the logging.xml so that it logs Security Token Service information to a separate log file, `diagnostic.log`, in the folder `<DomainHome>/config/fmwconfig/servers/<servername>/sts/log/`.

To create an Security Token Service log file to troubleshoot Security Token Service:

1. Open the file *DomainHome*/`config/fmwconfig/servers/`*servername*/`logging.xml`
2. Add the following in the appropriate sections:

```
<log_handler name='sts-handler'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
      <property name='path' value='sts/log'/>
      <property name='maxFileSize' value='10485760'/>
      <property name='maxLogSize' value='104857600'/>
    </log_handler>

<logger name='oracle.security.fed' level='TRACE:32'>
      <handler name='sts-handler'/>
    </logger>
```

8.8 Log File Location

All log messages go to the server log file of the WebLogic Server that the application is deployed on. The default location of the server log is:

WL_HOME`/user_projects/domains/`*domainName*`/servers/`*serverName*`/logs/`
serverName`-diagnostic.log`

8.9 Additional Considerations

The Security Token Service server can detect fake requests, such as replay attacks, that can occur if a user tries to steal token data from a request and send another request with the same token. In this case, the server detects the second fake request. The second issuance request with the same token in `<Env: Body>` goes to the Security Token Service server. The server denies the request after checking its UNT token cache, which indicates a replay attack.

9

Configuring High Availability for Identity Federation Components

This chapter describes Oracle Access Management Identity Federation 11*g*R2 high availability. See Integration with Access Manager 11gR2 in the *Oracle Fusion Middleware Integration Guide for Oracle Identity Management Suite* for additional details on Identity Federation.

This section includes the following topics:

- Section 9.1, "Identity Federation Component Architecture"
- Section 9.2, "Identity Federation High Availability Concepts"
- Section 9.3, "Identity Federation High Availability Configuration"
- Section 9.4, "Identity Federation Failover and Expected Behavior"
- Section 9.5, "Troubleshooting Identity Federation High Availability"

9.1 Identity Federation Component Architecture

Identity Federation service provides single sign-on capabilities to Oracle Access Management Access Manager in a multiple-domain identity network. It supports the broadest set of federation standards, enabling users to federate in heterogeneous environments and business associations, whether or not they implement other Oracle Identity Management products in their solution set.

Only Identity Federation 11*g* Release 2 supports the Service Provider (SP) functionality. For Identity Provider (IDP) support, use Identity Federation 11*g* Release 1.

Acting as an SP, Identity Federation enables you to manage your resources while offloading user authentication to an IDP, without having to synchronize users across security domains out of band. Once authenticated at the IDP, the SP can enable or deny access to users for the SP's applications, depending on local access policies.

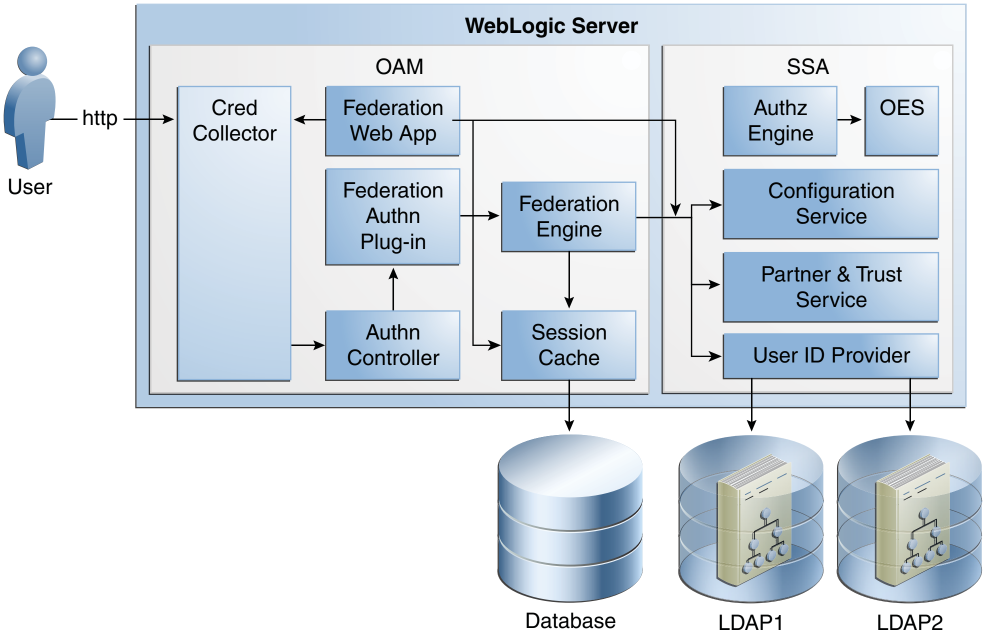
If a user no longer has an account with the IDP, the federation is terminated and cross-domain single sign-on for that user is automatically disabled.

Key features of Identity Federation include support for:

- Multiple leading federation protocols such as SAML 1.x and SAML 2.0
- Cross-protocol single sign-on and sign-out
- X.509 certificate validation
- Native Integration with Access Manager

- Integration with any LDAP Directory supported by Access Manager

Figure 9–1 Identity Federation Architecture



For more details about how Identity Federation integrates with Access Manager, see Integration with Access Manager 11*g*R2 in *Oracle Fusion Middleware Integration Guide for Oracle Identity Management Suite*.

9.1.1 Identity Federation Component Characteristics

Identity Federation is an Oracle Access Management component providing SP functionality for Cross Domain Single Sign-On. It enables Oracle Access Management to delegate user authentication to a remote Identity Provider partner. It supports a broad set of federation standards such as SAML 1.x, SAML 2.0.

- Section 9.1.1.1, "Runtime Processes"
- Section 9.1.1.2, "Process Lifecycle"
- Section 9.1.1.3, "Request Flow"
- Section 9.1.1.4, "Configuration Artifacts"
- Section 9.1.1.5, "External Dependencies"
- Section 9.1.1.6, "Identity Federation Log File Location"

9.1.1.1 Runtime Processes

Identity Federation is part of the Access Manager J2EE application deployed on the WebLogic Server.

Because Identity Federation runs within the Access Server, it has the same runtime processes as Access Manager. For more information, see Section 6.1.1.2.1, "Access Manager Process Lifecycle".

9.1.1.2 Process Lifecycle

Identity Federation follows the same process lifecycle as Access Manager. See Section 6.1.1.2.1, "Access Manager Process Lifecycle" for more information.

9.1.1.3 Request Flow

See Architecture in *Oracle Fusion Middleware Integration Guide for Oracle Identity Management Suite* for information on the Identity Federation request flow.

9.1.1.4 Configuration Artifacts

The Identity Federation configuration artifacts include the following files.

- DOMAIN_HOME/config/fmwconfig/oam-config.xml — configuration file containing instance-specific information.
- DOMAIN_HOME/config/fmwconfig/oam-policy.xml — present only when OES Micro SM is not being used.
- DOMAIN_HOME/config/fmwconfig/servers/instanceName/logging.xml — Logging config
- DOMAIN_HOME/config/fmwconfig/cwallet.sso — stores passwords that are used to connect to identity stores, database, and other entities. This is not for end user passwords.
- DOMAIN_HOME/config/fmwconfig/.oamkeystore — keystore containing keys and certificates OAM/Identity Federation owns
- DOMAIN_HOME/config/fmwconfig/amtruststore — keystore containing the trust anchors used for X509 cert validation
- DOMAIN_HOME/config/fmwconfig/amcrl.jar — zip file containing CRLs used for certificate revocation
- DOMAIN_HOME/config/fmwconfig/default-keystore.jks — OWSM keystore used to store keys and certificates used by the OWSM Agent, as well as trusted anchors used to validate X.509 certificates for WSS operations
- DOMAIN_HOME/config/fmwconfig/servers/*servername*/dms_config.xml — eventing configuration file
- DOMAIN_HOME/config/fmwconfig/component_events.xml — audit definition
- DOMAIN_HOME/config/fmwconfig/system-jazn-data.xml — Administration Console permissions
- DOMAIN_HOME/config/fmwconfig/cacerts.jks — keystore containing Certificate Authority certificates.

9.1.1.5 External Dependencies

The following external components are required for Identity Federation server to function:

- WebLogic Server
  - Administration Server
  - Managed Server
- LDAP based Identity Store
  - User Identity Repository

  - LDAP access abstracted by User/Role API.

> **Note:** Access Manager always connects to one Identity store, which can be a physical server or a load balancer IP. If the primary is down, Access Manager reconnects and expects the load balancer to connect it to the secondary.

- OCSP Responder Service
  - Real-time X.509 Certification Validation
- RDBMS Policy Store/Coherence Store
  - Policy (Authentication and Authorization) Repository
  - RDBMS access abstracted by the Access Manager policy engine
- Oracle Entitlements Server (though OAM)
- Federation Data Cache
  - For session and runtime information. You can configure Identity Federation to use the memory store or RDBMS store for this. However, in a high availability environment you must use a RDBMS store.

Data Repositories

The session information related to federation, partners where the user is signed in, and protocol data is stored in the session and cache. You can configure Identify Federation to use either a memory store or an RDBMS store for this data. For high availability environments, you must use a RDBMS store.

9.1.1.6 Identity Federation Log File Location

The default location of the WebLogic Server log file is:

```
WEBLOGIC_SERVER_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

Use the Oracle Enterprise Manager Fusion Middleware Control to view the log files.

9.2 Identity Federation High Availability Concepts

This section provides conceptual information about using Identity Federation in a high availability two-node cluster.

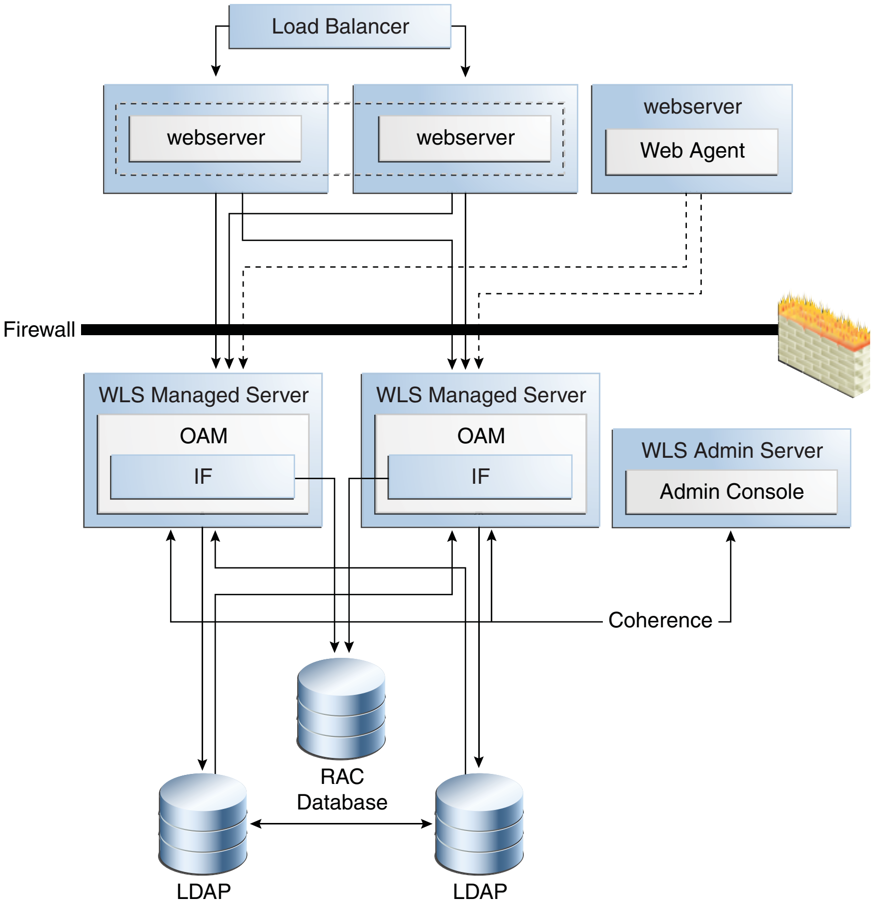
This section includes the following topics:

- Section 9.2.1, "Identity Federation High Availability Architecture"
- Section 9.2.2, "Identity Federation Prerequisites"

9.2.1 Identity Federation High Availability Architecture

Figure 9–2 shows the Identity Federation high availability architecture in an active-active configuration.

Figure 9–2 Identity Federation in a High Availability Architecture



In Figure 9–2,the hardware load balancer receives incoming authentication requests and routes them a web server in the web tier. These hosts have Oracle HTTP Server installed. The Oracle HTTP Server forwards requests to the WebLogic managed servers using `WebLogic plugin mod_wl_ohs.conf`.

The load balancing router should use session stickiness for HTTP traffic only.

The two managed servers that host the Oracle Access Server application are configured in a cluster which enables the Access Servers to work in active-active mode.

If a federation scheme protects the resource being accessed, OAM uses the Federation Engine to authenticate the user.

The LDAP directories that the Federation Engine uses are deployed in a cluster.

An Oracle RAC database provides high availability in the data tier.

The WebLogic Administration Server runs on the same host as one of the managed servers and deploys the WebLogic Administration Console, Oracle Enterprise Manager Fusion Middleware Control, and the Oracle Access Management Console. You can configure the Administration Server to run in active-passive mode on the host machine that runs the second managed server. In this configuration, if the first Administration Server becomes unavailable, you can start the Administration Server manually on the second host.

9.2.1.1 Starting and Stopping the Cluster

Identity Federation clusters start and stop in the same manner as OAM clusters. For more information, see Section 9.2.1.1, "Starting and Stopping the Cluster."

9.2.1.2 Cluster-Wide Configuration Changes

Configuration changes made through one cluster member propagate automatically to all others because the configuration is stored in the shared database. See Section 9.2.1.2, "Cluster-Wide Configuration Changes" for additional information.

9.2.2 Identity Federation Prerequisites

Because Identity Federation is part of OAM, it has the same prerequisites as OAM. See Section 9.2.1.2, "Cluster-Wide Configuration Changes" for more information.

Note: Oracle requires that you synchronize the system clocks on the cluster nodes when you are using Identity Federation in a high availability deployment.

9.3 Identity Federation High Availability Configuration

As a feature that runs on OAM servers, you configure Identity Federation high availability as part of OAM high availability. To configure OAM high availability, see Section 6.3, "Access Manager High Availability Configuration Steps". Note the following special considerations as you configure Identity Federation for high availability.

- Section 9.3.1, "Setting the Hostname and Port"
- Section 9.3.2, "Changing the ProviderID Value"
- Section 9.3.3, "Tuning Identity Federation Parameters"

9.3.1 Setting the Hostname and Port

Oracle recommends that you set the host name and port in the OAM/Identity Federation configuration to the load balancer host and port. If you do not, errors occur during Single Sign-On/Logout operation. Oracle also recommends that you use virtualized hostnames in OAM configuration so that, after a restore on a different host, the corresponding agent configuration does not require updates.

9.3.2 Changing the ProviderID Value

The ProviderId is a string that uniquely identifies the SP. The ProviderId for all servers in a cluster must be identical. The ProviderId defaults to `http://host:port/oam/fed/` at installation. If necessary, change or set this value after installation; do not change it during operation.

9.3.3 Tuning Identity Federation Parameters

You can tune the connection to the RAC database that stores session data.

If you use the artifact profile, use WLST to tune SOAP connection details.

Identity Federation parameters that you can set include the following:

- "Outgoing SOAP connection"

- "RDBMS Transient Store Asynchronous Settings"
- "RDBMS Artifact memory cache settings"
- "RDBMS Memory cache settings"
- "Interval for the RDBMS cleanup thread"

Outgoing SOAP connection

Outgoing SOAP connection parameters that you can tune include:

- Max connections total
- Max connections per host

RDBMS Transient Store Asynchronous Settings

Table 9–1 describes RDBMS transient store asynchronous settings.

Table 9–1 RDBMS Transient Store Asynchronous Settings

Setting	Description
rdbmsasynchronousmanagerinterval	Execution interval for the asynchronous thread manager
rdbmsasynchronousmanagersleep	Sleep interval for the asynchronous thread manager, to check if execution should occur
rdbmsasynchronousqueuesize	Size of the queue containing RDBMS operations of the same type (create session, create artifact)
rdbmsasynchronousqueuesleep	Sleep time before the calling thread can retry adding an operation to a queue if the queue is full
rdbmsasynchronousqueueretries	Number of retries when attempting to add an operation to the queue if the queue is full
rdbmsasynchronousthreadcore	Number of default threads in the RDBMS thread executor module for RDBMS asynchronous operations
rdbmsasynchronousthreadkeepalive	Maximum amount of time to keep the extra threads in the RDBMS thread executor module for RDBMS asynchronous operation
rdbmsasynchronousthreadmax	Maximum number of threads in the RDBMS thread executor module for RDBMS asynchronous operation
rdbmsasynchronousthreadpolicy	Thread policy of the RDBMS thread executor module for RDBMS asynchronous operation
rdbmsasynchronousthreadqueuesize	Thread queue size of the of the RDBMS thread executor module for RDBMS asynchronous operation

RDBMS Artifact memory cache settings

Table 9–2 describes RDBMS Artifact memory cache settings, used in conjunction with the RDBMS asynchronous module.

Table 9–2 RDBMS Artifact memory cache settings

Setting	Description
artifactrdbmscachetimeout	Time to live in the memory cache
artifactrdbmsretries	Maximum number of time to retry to locate an entry in RDBMS before returning a failure
artifactrdbmssleep	Sleeping time between retrying lookup operations

RDBMS Memory cache settings

Table 9–3 describes RDBMS Memory cache setting, with the exception of artifact settings (see Table 9–2).

Table 9–3 RDBMS Memory Cache Settings

Setting	Description
transientrdbmscachesize	Cache size
transientrdbmscachetimeout	Time to live for cache objects before becoming invalid and forcing an RDBMS lookup operation when an object is searched

Interval for the RDBMS cleanup thread

The setting for the RDBMS cleanup thread interval is `rdbmscleanupinterval`, which indicates the sleep interval of the thread that removes expired entries from Identity Federation database tables.

9.4 Identity Federation Failover and Expected Behavior

This section describes steps for performing failover operations on Identity Federation instances deployed in a high availability environment and their expected behavior.

To perform a test of a failover of an Identity Federation instance and to check the status of Identity Federation:

1. Log in to the Administration Server console. Go to **Home**, **Summary of Security Realms**, **myrealm**, **Users and Groups**, **Realm Roles**, then **Edit Global Role**. Add the group `OAMAdminstrators`.
2. Log in to the OAM Administration Console. Go to **System Configuration**, **Common Settings**, **Available Services** then enable Identity Federation.
3. Set up Identity Federation between an IDP instance (which can be an Oracle Identity Federation 11*g* Release 1 instance) and this Identity Federation 11*g*Release 2 instance that is functioning as an SP.
4. Integrate with OAM 11gR2 and protect a resource with OIFScheme. Follow the steps in Integration with Access Manager 11gR2 in *Oracle Fusion Middleware Integration Guide for Oracle Identity Management Suite*.
5. Shut down one of the two managed servers and try to access the protected resource.
6. When the IDP login page appears, restart the managed server that you stopped in the previous step, shut down the managed server that was running, and try to complete the operation.

9.5 Troubleshooting Identity Federation High Availability

Use the following tips to troubleshoot Identity Federation issues:

- Identity Federation logs its messages in the Managed Server log files, for example:

```
WEBLOGIC_SERVER_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

- Verify that the hostname and port in the Identity Federation server configuration are set to the load balancer host and port, otherwise errors occur during Single Sign-On operation.

- If system clocks on the computers which IDP and SP run on have different times, errors occur during Single Sign-On. Fix this by setting the system clocks to have the same time or by adjusting the server drift using the Server Properties page in Oracle Enterprise Manager Fusion Middleware Control.
- The `ProviderId` string uniquely identifies the IDP/SP and must be identical for all servers in a cluster. The `ProviderId` string defaults to: `http://`*`host`*`:`*`port`*`/oam/fed` at installation. If you must change `ProviderId`, change it after installation, not during an operation.

10 Configuring High Availability for Oracle Entitlements Server

This chapter introduces Oracle Entitlements Server and describes how to set up a high availability environment for Oracle Entitlements Server components.

Oracle Entitlements Server is a fine-grained authorization product that allows an organization to protect its resources by defining and managing policies that control access to, and usage of, these resources. A policy defines access privileges by specifying who can do what to which resource, when it can be done, and how. The policy can enforce controls on all types of resources including software components and business objects.

- See "Overview of the Oracle Entitlements Server Architecture" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for an illustration and description of the Oracle Entitlements Server component architecture.
- See "Features of Oracle Entitlements Server 11gR2" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information on Oracle Entitlements Server features.

This chapter includes the following topics:

- Section 10.1, "Oracle Entitlements Server High Availability Concepts"
- Section 10.2, "Configuring Oracle Entitlements Server High Availability"

10.1 Oracle Entitlements Server High Availability Concepts

This section provides conceptual information about using Oracle Entitlements Server in a high availability two-node cluster.

This section includes the following topics:

- Section 10.1.1, "Oracle Entitlements Server High Availability Architecture"
- Section 10.1.2, "Oracle Entitlements Server Security Module High Availability"
- Section 10.1.3, "Load Balancing"
- Section 10.1.4, "Failover Considerations"
- Section 10.1.5, "Protection from Failures and Expected Behaviors"
- Section 10.1.6, "Starting and Stopping the Oracle Entitlements Server Cluster"
- Section 10.1.7, "Cluster-Wide Configuration Changes"
- Section 10.1.8, "Considerations for Synchronizing with LDAP"

10.1.1 Oracle Entitlements Server High Availability Architecture

This section describes the following high availability architecture scenarios for Oracle Entitlements Server components.

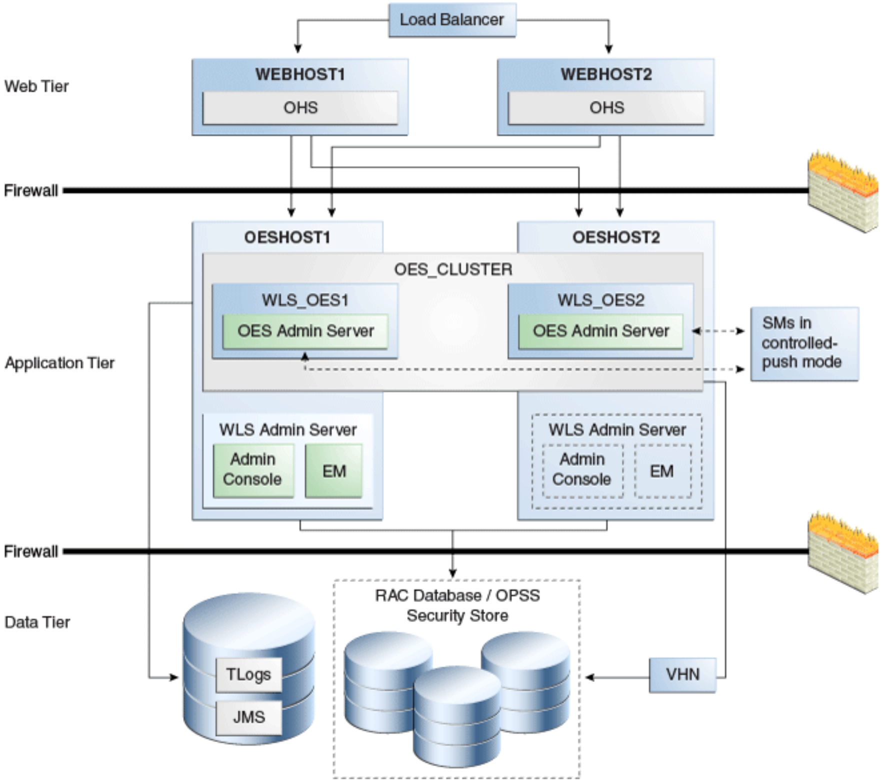
This section includes the following topics:

- Section 10.1.1.1, "Oracle Entitlements Server Administration Server High Availability"
- Section 10.1.1.2, "Security Module (OES Client)/Policy Information Point High Availability"
- Section 10.1.1.3, "Security Module in Proxy Mode Working Against Web Service / RMI Security Module in Controlled-Push Mode High Availability"
- Section 10.1.1.4, "Security Module in Proxy Mode Working Against Web Service / RMI Security Module in Controlled Pull Mode High Availability"
- Section 10.1.1.5, "Oracle Entitlements Server WebLogic Server Security Module High Availability"

10.1.1.1 Oracle Entitlements Server Administration Server High Availability

Figure 10–1 shows the Oracle Entitlements Server Administration Server deployed in a high availability architecture in an active-active configuration.

Figure 10–1 Oracle Entitlements Server Administration Server High Availability Architecture



On OESHOST1, you see the following installations:

- An Oracle Entitlements Server instance is installed in the WLS_OES1 Managed Server and a APM instance is installed in the WLS_OES1 Managed Server.
- The Oracle RAC database is configured in a JDBC multi data source or GridLink Data source to protect the instance from Oracle RAC node failure.
- A WebLogic Server Administration Server is installed. Under normal operations, this is the active Administration Server.

On OESHOST2, you see the following installations:

- An Oracle Entitlements Server instance is installed in the WLS_OES2 Managed Server and an APM instance is installed in the WLS_OES2 Managed Server.
- The Oracle RAC database is configured in a JDBC multi data source to protect the instance from Oracle RAC node failure.
- The instances in the WLS_OES1 and WLS_OES2 Managed Servers on OESHOST1 and OESHOST2 are configured as the OES_CLUSTER cluster.
- A WebLogic Server Administration Server is installed. Under normal operations, this is the passive Administration Server. You make this Administration Server active if the Administration Server on OESHOST1 becomes unavailable.

You can configure Oracle Entitlements Server Security Modules in controlled-push mode so that two Oracle Entitlements Server Administration Servers function as a registration server and backup registration server. The Oracle Entitlements Server Security Modules can switch to a backup server and get distributed policy from the Oracle Entitlements Server Administration Server if the registration server is down. See Section 10.1.4, "Failover Considerations" for information about failover scenarios and behavior.

10.1.1.2 Security Module (OES Client)/Policy Information Point High Availability

You can deploy the Security Module so that it is embedded and configure it to failover between different Policy Information Points (PIP). A PIP is a data repository, a source from which information can be retrieved for use when evaluating policies for an authorization decision. For more information on PIP, see The Policy Information Point in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server*.

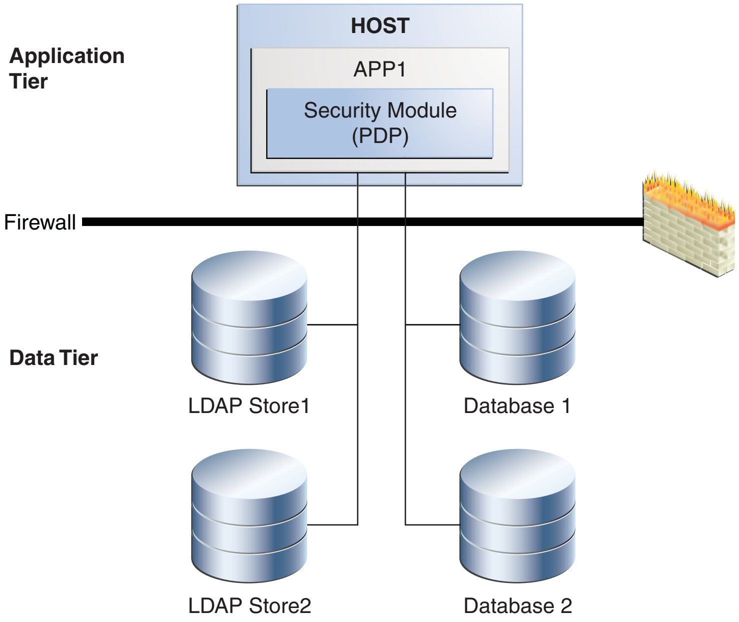
See the following topics for deployment options:

- "Oracle Entitlements Server PIP with Multiple LDAP/JDBC URLs"
- "Oracle Entitlements Server PIP with RAC and Load Balancer"

Oracle Entitlements Server PIP with Multiple LDAP/JDBC URLs

Figure 10–2 shows an embedded Security Module instance in a high availability deployment. With both LDAP and DB-based PIP, you can configure multiple endpoints for external sources to failover between them. For DB-based PIP, you can also configure a multi-source datasource.

Figure 10–2 Security Module / Policy Information Point Configuration

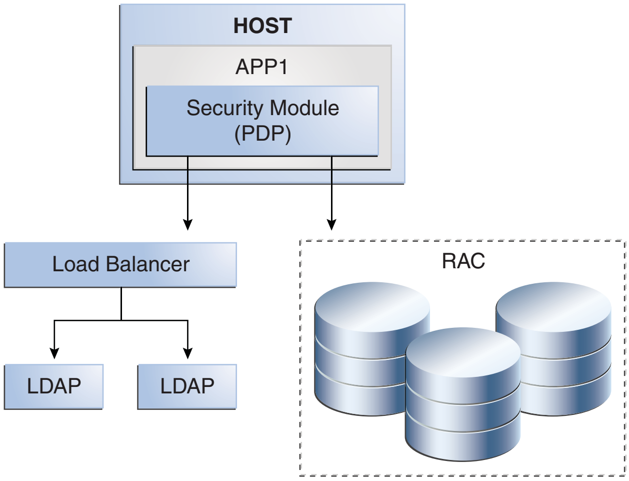


In Figure 10–2, the Security Module (PDP) uses LDAP 1 or Database 1 as its primary PIP. In the case of failover, the Security Module fails over to LDAP2 and Database 2.

Oracle Entitlements Server PIP with RAC and Load Balancer

Another high availability deployment option for Oracle Entitlements Server is one in which the Security Module (PDP) uses the RAC database or LDAP servers with a load balancer. In the case of failover, the Security Module fails over to the RAC, as Figure 10–3 shows.

Figure 10–3 Oracle Entitlements Server PIP with RAC and Load Balancer



10.1.1.3 Security Module in Proxy Mode Working Against Web Service / RMI Security Module in Controlled-Push Mode High Availability

Oracle Entitlements Server supports a proxy mode that allows clients to invoke authorization services remotely. See Using the Security Module Proxy Mode of the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server*.

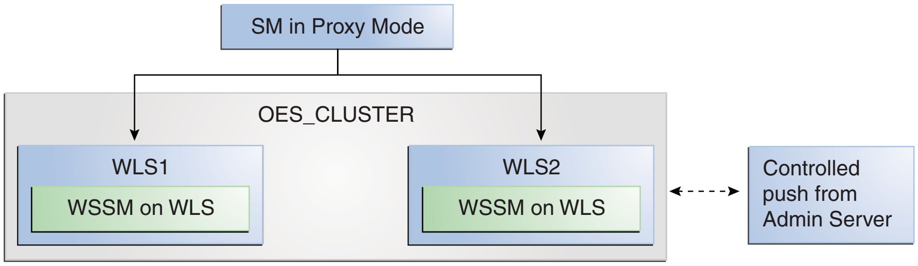
There are three deployment options for deploying Security Module in proxy mode:

- "Web Service Security Module on WebLogic Server Deployment"
- "Standalone Web Service Security Module Deployment"
- "RMI Security Module Deployment"

Web Service Security Module on WebLogic Server Deployment

Figure 10–4 shows a Web Service Security Module on WebLogic Server.

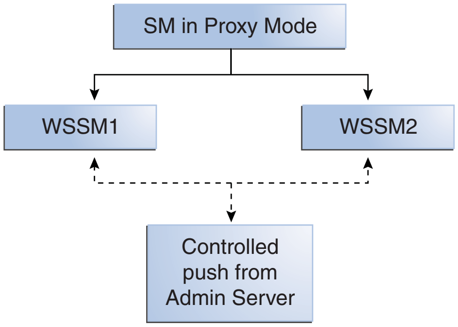
Figure 10–4 Web Service Security Module on WebLogic Server Deployment



Standalone Web Service Security Module Deployment

Figure 10–5 shows a standalone Web Service Security Module deployment.

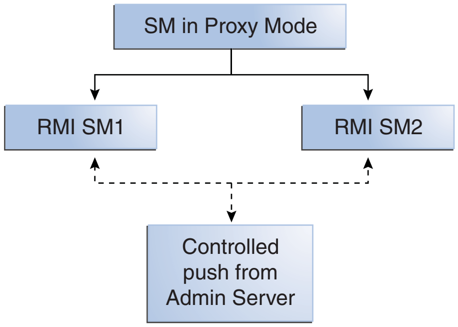
Figure 10–5 Standalone Web Service Security Module Deployment



RMI Security Module Deployment

Figure 10–6 shows a RMI Security Module deployment.

Figure 10–6 RMI Security Module Deployment



10.1.1.4 Security Module in Proxy Mode Working Against Web Service / RMI Security Module in Controlled Pull Mode High Availability

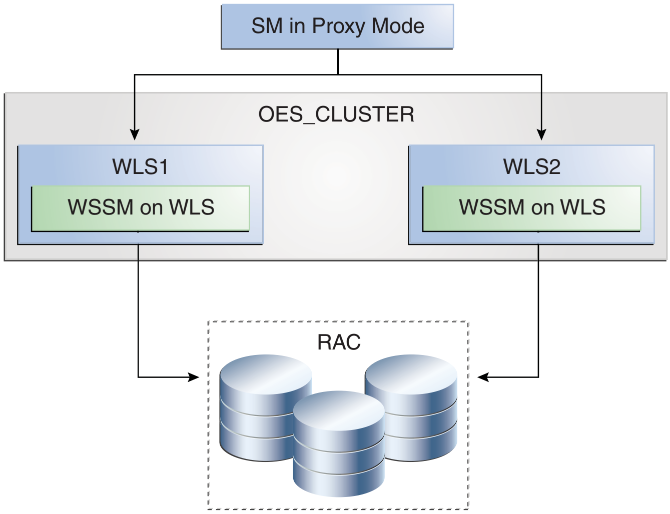
Options to deploy Security Module in proxy mode working against Web Service/RMI Security Modules in controlled-pull mode include the following:

- "Web Service Security Module on WebLogic Server"
- "Standalone Web Service Security Module"
- "RMI Security Module"

Web Service Security Module on WebLogic Server

Figure 10–7 shows Web Service Security Module on WebLogic Server.

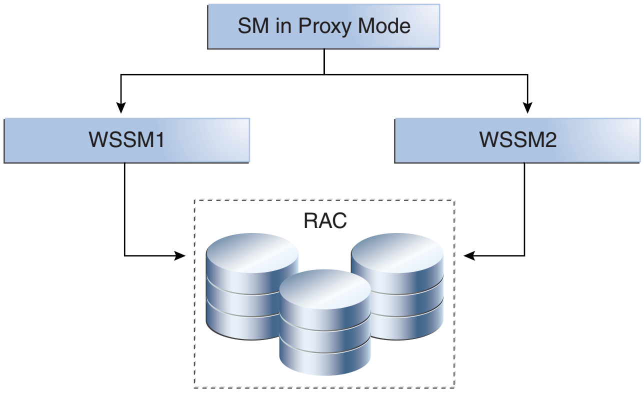
Figure 10–7 Web Service Security Module on WebLogic Server



Standalone Web Service Security Module

Figure 10–8 shows a standalone Web Service Security Module deployment.

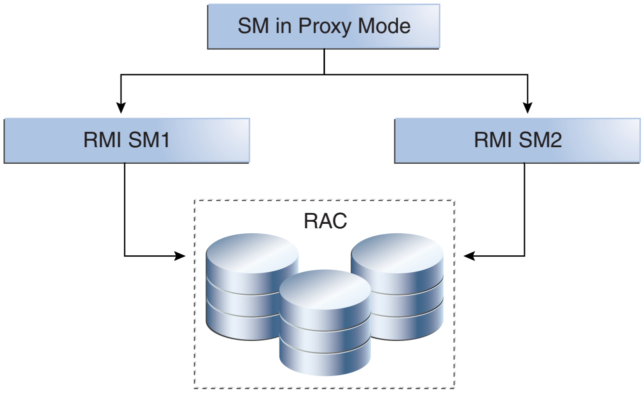
Figure 10–8 Standalone Web Service Security Module



RMI Security Module

Figure 10–9 shows a RMI Security Module deployment.

Figure 10–9 RMI Security Module



See PDP Proxy in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information on configuring the Web Services Security Module proxy client and the RMI Security Module proxy client.

10.1.1.5 Oracle Entitlements Server WebLogic Server Security Module High Availability

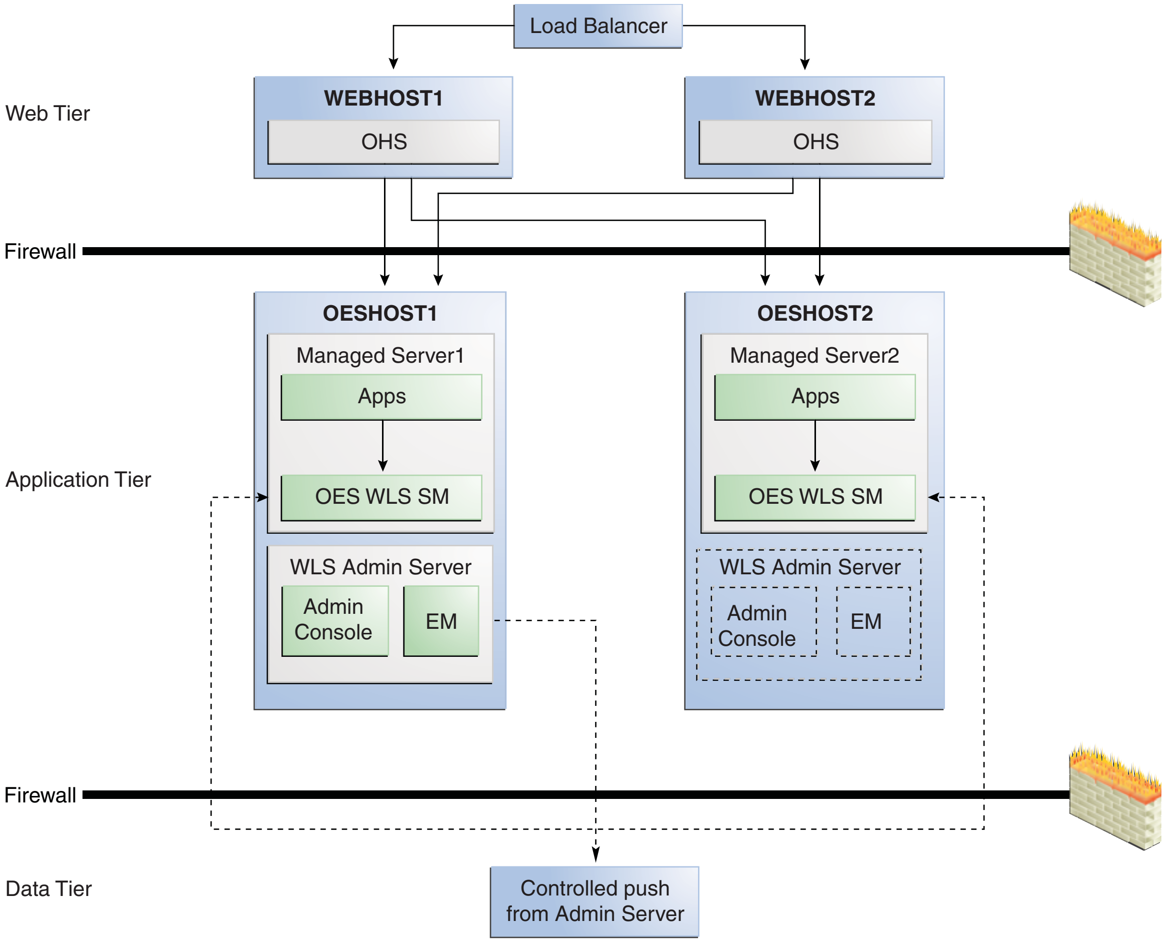
There are two deployment options for OES WebLogic Server high availability:

- "Oracle Entitlements Server WebLogic Server Security Module, Controlled-push Mode"
- "Oracle Entitlements Server WebLogic Server Security Module High Availability, Controlled-pull or Non-Controlled Mode"

Oracle Entitlements Server WebLogic Server Security Module, Controlled-push Mode

The following graphic shows Oracle Entitlements Server WebLogic Server Security Module in controlled-push mode.

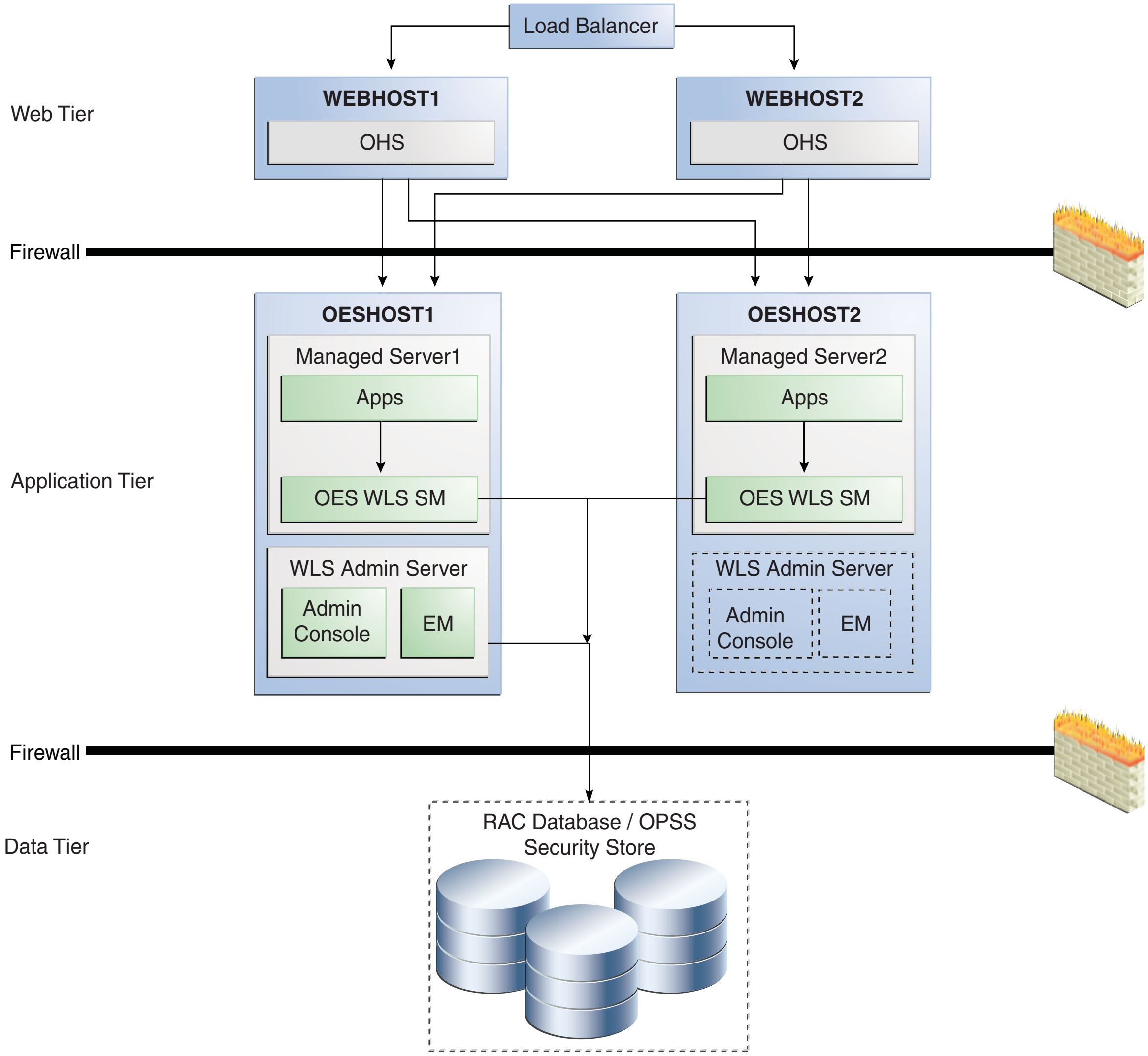
Figure 10–10 Oracle Entitlements Server WebLogic Server Security Module, Controlled-push Mode



Oracle Entitlements Server WebLogic Server Security Module High Availability, Controlled-pull or Non-Controlled Mode

The following figure shows an Oracle Entitlements Server with WebLogic Server Security Module in controlled-pull/non-controlled mode.

Figure 10–11 Oracle Entitlements Server WebLogic Server Security Module Controlled-pull/Non-Controlled Mode



10.1.2 Oracle Entitlements Server Security Module High Availability

When the Security Module reads policy from the OPSS security store for controlled-pull or non-controlled distribution, use Oracle-recommended high availability methods for an application accessing a database.

10.1.3 Load Balancing

For all high availability scenarios, you can deploy the load balancer:

- In front of Authorization Policy Manager (APM) for user-to-APM communication. Oracle recommends a sticky connection to avoid losing data that does not persist to the policy store.
- In front of the Web Service Security Module for client-to-Security Module communication. Oracle recommends a sticky connection to maximize cache usage.

> **Note:** Oracle Entitlements Server does not have any timeout requirements for the load balancer.

10.1.4 Failover Considerations

This section describes Oracle Entitlements Server failover considerations.

Table 10–1 Oracle Entitlements Server Failover Scenarios and Behavior

Failover Scenario	Failover Behavior
OES Policy Store fails	APM and Security Module in controlled-pull and uncontrolled mode switch to a working instance if multi-source data source is used. If the policy store instance is lost while the transaction is being processed: ■ APM returns an error to the user, who can repeat the request. ■ Security Module retries the transaction. Security Module uses only read operations. If Security Module is in controlled-pull mode, it uses the locally-persisted copy of the policy store.
OES Admin Server fails	■ User-to-APM communication - If a load balancer is present, it redirects the user to another APM instance. All unsaved data in the user session is lost, however, the user has full access to persisted policy data. ■ Security Module-to-APM communication - In controlled push distribution, Security Module registers with OES Admin on startup and retries the request to back-up instance if primary one is down. Retries continues until working instance is detected. While trying to reach OES Admin, Security Module uses a locally-persisted copy of the policy store.
Web Service Security Module or RMI Security Module fails	Security Module Proxy retries requests until it reaches the configured number of retries.
DB or LDAP attribute source fails	Security Module (OES Client) continues to try to read data until it reaches the configured number of retries.

10.1.5 Protection from Failures and Expected Behaviors

This section describes protection from different types of failure in an Oracle Entitlements Server active-active cluster.

- Section 10.1.5.1, "Expected Client Application Behavior When Failure Occurs"
- Section 10.1.5.2, "Node failure"
- Section 10.1.5.3, "Database failure"

10.1.5.1 Expected Client Application Behavior When Failure Occurs

Oracle Entitlements Server failover is not transparent. You must reestablish the connection during a WebLogic Server instance failover using Oracle Entitlements Server.

10.1.5.2 Node failure

Node failures are treated in the same way as WebLogic Server crashes.

10.1.5.3 Database failure

Oracle Entitlements Server is protected against failures in the database by using multi data sources, which you configure during the initial system set up. The multi data sources guarantee that when an Oracle RAC database instance fails, the connections reestablish with available database instances. The multi data source allows you to configure connections to multiple instances in an Oracle RAC database.

10.1.6 Starting and Stopping the Oracle Entitlements Server Cluster

In a high availability architecture, you deploy Oracle Entitlements Server on a WebLogic cluster, which has at least two servers as part of the cluster.

By default, WebLogic Server starts, stops, monitors, and manages the various lifecycle events for the application. The Oracle Identity Manager application leverages the high availability features of the underlying Oracle WebLogic clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

Use one or more of the following command line tools and consoles to manage Oracle Entitlements Server lifecycle events:

- WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle WebLogic Scripting Tool (WLST)

10.1.7 Cluster-Wide Configuration Changes

For high availability environments, changing the configuration of one Oracle Entitlements Server instance changes the configuration of all the other instances, because all the Oracle Entitlements Server instances share the same configuration repository. Nearly all Oracle Entitlements Server deployments use cluster configurations. The only exception is Oracle Entitlements Server Administration Server, which is usually not clustered.

10.1.8 Considerations for Synchronizing with LDAP

Synchronization between LDAP and the Oracle Entitlements Server database is handled by a process called Reconciliation, which is a scheduled process that runs in the background primarily. You can also run this process manually.

If an LDAP outage occurs during the Synchronization process, the data which did not get into Oracle Entitlements Server is picked up during the next run of the reconciliation task.

10.2 Configuring Oracle Entitlements Server High Availability

This section provides high-level instructions for setting up a high availability deployment for Oracle Entitlements Server.

The Oracle Entitlements Server Administration Server high availability deployment is the same as a typical Oracle application.

To set up high availability for users accessing the Oracle Entitlements Server Administration Server user interface, use a WebLogic cluster.

To set up a high availability database for Administration Server user interface, you use multi source data source, Oracle RAC, and other typical elements.

This section includes the following topics:

- Section 10.2.1, "Prerequisites for Oracle Entitlements Server Configuration"
- Section 10.2.2, "Configure Weblogic Domain for OES Administration Server on OESHOST1"
- Section 10.2.3, "Post-Configuration and Verification"
- Section 10.2.4, "Configure OES Security Module in Controlled-push Mode with Oracle Entitlements Server Administration Server High Availability"
- Section 10.2.5, "Configure Oracle Entitlements Server Security Module in Proxy Mode with PDP High Availability"
- Section 10.2.6, "Configure Oracle Entitlements Server Policy Information Point with High Availability"
- Section 10.2.7, "Configuring Oracle Entitlements Server Web Service Security Module on WebLogic High Availability"
- Section 10.2.8, "Configuring Oracle Entitlements Server WebLogic Security Module High Availability"
- Section 10.2.9, "Using RAC Datasource for Security Module in Controlled-pull Mode and Non-controlled Mode"
- Section 10.2.10, "Configuring Oracle Entitlements Server to Work with the Web Tier"

10.2.1 Prerequisites for Oracle Entitlements Server Configuration

Complete the following steps before you configure Oracle Entitlements Server high availability:

1. Use the Repository Creation Utility to create the Oracle Entitlements Server schemas in the Oracle RAC database. See Installation and Configuration Roadmap for Oracle Entitlements Server in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for information on creating schemas.
2. Install Weblogic Server on OESHOST1 and OESHOST2. See Section 5.3.1.1.1, "Installing Oracle WebLogic Server" for more information.
3. Install the Oracle Entitlements Server Administration software on OESHOST1 and OESHOST2. See Installing Oracle Entitlements Server Administration Server in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for more information.
4. Install the Oracle Entitlements Server Client. See Installing Oracle Entitlements Client in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for more information.

10.2.2 Configure Weblogic Domain for OES Administration Server on OESHOST1

To configure a WebLogic domain for the OES Administration Server on OESHOST1, perform these steps:

1. Run the `<MW_HOME>/oracle_common/common/bin/config.sh` script.
2. On the Welcome screen, select **Create a new WebLogic domain** and click **Next**. The Select Extension Source screen appears.
3. On the Select Extension Source screen, select **Oracle Entitlements Server for Managed Server - 11.1.1.0[Oracle_IDM1**. The Configuration Wizard automatically

selects Oracle JRF, Oracle Platform Security Service, and Basic WebLogic Server Domain.

Click **Next**.

4. In the Specify Domain Name and Location screen, enter the domain name for the domain you are creating and the domain location. Click **Next**. The Configure Administrator User Name and Password screen appears.
5. Configure a user name and a password for the administrator. The default user name is `weblogic`. Click **Next**.
6. Choose the Weblogic domain startup mode and JDK in the Configure Server Start Mode and JDK screen.
7. In the Configure JDBC Component Schema screen, configure JDBC properties for all of the schemas then click **Next**.
8. On the Test JDBC Component Schema screen, click **Select All** then **Test Connections**. Click **Next**.

   If the data source validation succeeds, click **Next**.

   If the data source validation fails, click **Previous**, correct the issue, then try again.
9. On the Select Optional Configuration screen, select **Administration Server and Managed Servers, Clusters and Machines**. Click **Next**.
10. In the Configure the Administration Server screen, enter the following values:
    - **Name:** `AdminServer`
    - **Listen address**: `All Local Addresses`
    - **Listen port:** `7001`
    - **SSL listen port:** `7002`
    - Select **SSL enabled**

    Click **Next**.
11. On the Configure Managed Servers screen, when you first enter the screen, one managed server called `oes_server1` is created automatically. You can rename oes_server1 and update its attributes for this entry.

    For example:
    - **Name**: *oes_server1*
    - **Listen Address**: OESHOST1.*example.com*
    - **Listen Port**: 14600
    - **SSL Port**: 14601

    For the second OES_SERVER, click **Add** and enter the following values:
    - **Name**: *oes_server2*
    - **Listen Address**: OESHOST2.*example.com*
    - **Listen Port**: 14600
    - **SSL Port:** 14601
    - Select **SSL enabled**

    Click **Next**.

12. In the Configure Clusters screen, click **Add** to create a cluster.

    Enter the name `oes_cluster`. Select unicast for Cluster messaging mode, then enter the Cluster address in the format *listen address or DNS name of oes_server1:port,listen address or DNS name of oes_server2:portmanaged server1:port,managed server2: port*.

    Click **Next**.

13. On the Assign Servers to Clusters screen, associate the managed servers with the cluster:

    Click on the cluster name **oes_cluster** in the right window.

    Click on the managed server **oes_server1** then click the arrow to assign it to the cluster.

    Repeat the preceding steps for the managed server **oes_server2**.

    Click **Next**.

14. On the Configure Machines screen, create a machine for each host in the topology.

    Click on the **Unix** tab.

    For Admin Server Host:

    - **Name**: Name of your host. Use the DNS name here.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

    Leave all other values at the default settings.

    Repeat the preceding steps for OESHOST1 and OESHOST2 and enter the following values. Leave all other values at the default settings.

    - **Name**: Name of the host. A good practice is to use the DNS name.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

    For Unix operating systems, delete the default local machine entry under the **Machines** tab.

    Click **Next**.

15. On the Assign Servers to Machines screen, you assign the managed servers that will run on the machines you just created. Follow these steps:

    Click on a machine in the right hand window.

    Click on the managed servers you want to run on that machine in the left window.

    Click on the arrow to assign the managed servers to the machine.

    Repeat these steps until you assign all managed servers to the appropriate machine.

    Assign servers to machines as follows:

    - ADMINHOST: **Admin Server**
    - OESHOST1: **oes_server1**
    - OESHOST2: **oes_server2**

Click **Next**.

16. On the Configuration Summary screen, click **Create**.
17. Verify that the first RAC database instance in the OPSS security store configuration is running.
18. Configure the OPSS Security Store. See Configuring Security Store for OES Administration Server in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management*.

10.2.3 Post-Configuration and Verification

This section includes the following topics:

- Section 10.2.3.1, "Starting Node Manager"
- Section 10.2.3.2, "Validating the WebLogic Administration Server"
- Section 10.2.3.3, "Creating a Separate Domain Directory for Managed Servers in the Same Node as the Administration Server"
- Section 10.2.3.4, "Start Node Manager on Remote Hosts"
- Section 10.2.3.5, "Stop and Start the WebLogic Administration Server and start oes_server1 and oes_server2"

10.2.3.1 Starting Node Manager

Perform these steps to start Node Manager on the administration host, for example, OESHOST1.

1. Run the `startNodeManager.sh` script located in the `MW_HOME/wlserver_10.3/server/bin/` directory.
2. Run the `setNMProps.sh` script to set the StartScriptEnabled property to true:

   ```
   cd MW_HOME/oracle_common/common/bin
   ```

3. Stop the Node Manager by killing the Node Manager process.
4. Start Node Manager.

10.2.3.2 Validating the WebLogic Administration Server

Perform the following steps to verify that the Administration server is configured properly.

1. Start Weblogic Administration Server by using `./startWeblogic.sh` in the new domain.
2. In a browser, enter the URL for the Oracle WebLogic Server Administration Console, for example:

   `http://`*<OESHOST1>*`:7001/console`

3. Log in as the WebLogic administrator, for example, `weblogic`.

10.2.3.3 Creating a Separate Domain Directory for Managed Servers in the Same Node as the Administration Server

Use the pack and unpack commands to separate the domain directory used by the Administration Server from the domain directory used by the managed server in OESHOST1.

To create a separate domain directory on OESHOST1:

1. Run the pack command to create a template pack as follows:

   ```
   cd MW_HOME/oracle_common/common/bin
   ```

   ```
   ./pack.sh -managed=true -domain=ORACLE_BASE/admin/domain_
   name/aserver/domain_name -template=domaintemplate.jar -template_
   name=domain_template
   ```

2. Run the unpack command to unpack the template in the managed server domain directory as follows:

   ```
   cd MW_HOME/oracle_common/common/bin
   ```

   ```
   ./unpack.sh -domain=ORACLE_BASE/admin/domain_name/mserver/domain_name
   -template=domaintemplate.jar
   ```

10.2.3.3.1 Propagate Changes to Remote Server Perform an unpack on remote hosts before you start managed servers on remote hosts, for example, OESHOST2.

1. Copy the file `domaintemplate.jar` created in Section 10.2.3.3, "Creating a Separate Domain Directory for Managed Servers in the Same Node as the Administration Server" to OESHOST2.

2. Run unpack on the host on OESHOST2 using the following commands:

   ```
   cd MW_HOME/oracle_common/common/bin
   ```

   ```
   ./unpack.sh -domain=ORACLE_BASE/admin/domain_name/mserver/domain_name
   -template=domaintemplate.jar
   ```

10.2.3.4 Start Node Manager on Remote Hosts

To start Node Manager on remote hosts, follow these steps:

1. On OESHOST2, start the Node Manager to create the `nodemanager.properties` file by using the `startNodemanager.sh` script located in the `MW_HOME/wlserver_10.3/server/bin` directory.

2. Run the `setNMProps.sh` script to set the `StartScriptEnabled` property to true:

   ```
   cd MW_HOME/oracle_common/common/bin
   ```

   ```
   ./setNMProps.sh
   ```

3. Stop and start the Node Manager.

10.2.3.5 Stop and Start the WebLogic Administration Server and start oes_server1 and oes_server2

1. Restart the WebLogic Administration Server.

2. In a browser, enter the URL for the Oracle WebLogic Server Administration Console, for example:

   `http://`*<OESHOST1>*`:7001/console`

3. Log in as the WebLogic administrator, for example, `weblogic`.

4. Start oes_server1 and oes_server2 managed servers from the WebLogic Server Admin console.

Note: You can also start the managed server by using the `startManagedWebLogic.sh` script in the domain directory subfolder bin. For example:

```
./startManagedWebLogic.sh oes_server1 http://localhost:7001
```

---

5. Validate the OES Admin Server instance on OESHOST1 by opening the APM console at the URL `http://`*<OESHOST1>*`:14600/apm`

   Log in with the WebLogic username and password.

6. Validate the OES Admin Server Instance on OESHOST2 by opening up the APM Console in a web browser at `http://`*<OESHOST2>*`:14600/apm`

10.2.4 Configure OES Security Module in Controlled-push Mode with Oracle Entitlements Server Administration Server High Availability

To configure the Oracle Entitlements Server Security Module in controlled-push mode with high availability, you set high availability parameters using the OES Security Module configuration user interface:

1. Change to the bin directory in the appropriate Security Module instance directory and run the following script on the command line.

   ```
   cd $OES_CLIENT_HOME/oes_sm_instances/SM_Name/bin
   ```

2. Run `oessmconfig.sh` to start the SMConfig UI.

   See Starting the SMConfig UI in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information.

3. Set the following parameters in the `jps-config.xml` file:

   - `oracle.security.jps.runtime.pd.client.backupRegistrationServerURL`
   - `oracle.security.jps.runtime.pd.client.registrationRetryInterval`

     The following example shows the `backupRegistrationServerURL` used as a backup when the `RegistrationServerURL` fails.

     ```
     <property
     name="oracle.security.jps.runtime.pd.client.backupRegistrationServerURL"
     value="https://slc00bqz:14601/pd-server"/>
     ```

     See Configuring the Java Security Module in *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information.

10.2.5 Configure Oracle Entitlements Server Security Module in Proxy Mode with PDP High Availability

To configure the Security Module in proxy mode with PDP high availability:

1. See Using the Security Module Proxy Mode in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* to configure the Security Module in proxy mode.

2. Change the PDP address by adding a comma-separated value as `oracle.security.jps.pdp.proxy.PDPAddress`

   For example:

```
oracle.security.jps.pdp.proxy.PDPAddress=http://ws1:9410,http://ws2:941
0
```

See PDP Proxy Client Configuration in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information.

10.2.6 Configure Oracle Entitlements Server Policy Information Point with High Availability

To configure the Policy Information Point high availability:

1. Change to the bin directory in the appropriate Security Module instance directory and run the following script on the command line:

   `cd $OES_CLIENT_HOME/oes_sm_instances/`*SM_Name*/bin

2. Run `oessmconfig.sh` to start the SMConfig UI.

   See Starting the SMConfig UI in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information.

3. Set attribute retriever parameters for Policy Information Point high availability. See Configuring Attribute Retrievers in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* for more information.

> **Note:** You can set multiple values for the `ldap.url` or `jdbc.url` attribute retriever parameter. Separate values with a comma; the first value is treated as the primary value. See Configuring the LDAP Repository Attribute Retriever Parameters in the *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server*.

10.2.7 Configuring Oracle Entitlements Server Web Service Security Module on WebLogic High Availability

You can configure Oracle Entitlements Server Web Service Security Module on WebLogic for high availability by means of a WebLogic cluster.

To configure Oracle Entitlements Server Web Service Security Module on WebLogic:

On OESHOST1

1. Run `OESCLIENT_HOME/oessm/bin/config.sh` to create a Web Service Security Module and a WebLogic Server domain.

   For example:

   ```
   ./config.sh -smType ws -onWLS -smConfigId <ws_name> -serverLocation <wls_home>
   -pdServer <oes_admin_server> -pdPort <oes_admin_ssl_port>
   ```

2. On the Welcome screen, select **Create a WebLogic Domain** then click **Next**.

3. On the Select Domain Source screen, select **Generate a domain configured automatically to support the following added products**. From the list, select **Oracle Entitlements Server Web Service Security Module on Weblogic For Managed Server**.

   Click **Next**.

4. On the Specify Domain Name and Location screen, enter the name and location for the domain and all its applications:

- **Domain Name**: <*domain name*>
- **Domain Location**: Accept the default entry.

5. On the Configure Administration Server Username and Password screen, enter the following:
   - **Name**: `weblogic`
   - **User Password**: Password for the WebLogic user
   - **Confirm User Password**: Password for the WebLogic user
   - **Description**: Description for the WebLogic user.
6. On the Configure Server Start Mode and JDK screen, select **Production Mode and JDK**.
7. On the Select Optional Configuration screen, select **AdminServer and Managed Servers**. Click **Next**.
8. On the Configure Administration Server screen, enter the following:
   - **Name**: `AdminServer`
   - **Listen address**: All Local Addresses
   - **Listen port**: `7001`
   - **SSL Listen port**: `7002`

   Select **SSL Enabled** then click **Next**.
9. On the Configure Managed Servers screen, the default managed server `wsonwls_server1` is created. Change the details of `wsonwls_server1` and then add the second managed server:

   For `wsonwls_server1`, enter these values:
   - **Name**: `wsonwls_server1`
   - **Listen address**: `WSSMHOST1`
   - **Listen port**: `14610`
   - **SSL listen port**: `14611`

   For the second managed server, click **Add** and enter these values:
   - **Name**: `wsonwls_server2`
   - **Listen address**: `WSSMHOST2`
   - **Listen port**: `14610`
   - **SSL listen port**: `14611`
10. In the Configure Clusters screen, click **Add** and enter `wssm_cluster`. Select **unicast** for **Cluster messaging mode** then enter the Cluster address as *managed_server1:port,managed_server2: port*

    Click **Next**.
11. On the Assign Servers to Clusters screen, associate the managed servers with the cluster:
    - Click on the cluster `wssm_cluster` in the right window.
    - Click on the managed server `wsonwls_server1` then click the arrow to assign it to the cluster.

Repeat the preceding steps for the managed server `wsonwls_server2`.

Click **Next**.

12. On the Configure Machines screen, create a machine for each host in the topology.

    Click on the **Unix** tab for a Unix operating system.

    For Admin Server Host:

    - **Name**: Name of your host. A good practice is to use the DNS name here.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

    Leave all other values at the default settings.

    Repeat the preceding steps for WSSMHOST1 and WSSMOESHOST2 and enter the following values. Leave all other values at the default settings.

    - **Name**: Name of the host. A good practice is to use the DNS name.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

    For Unix operating systems, delete the default local machine entry under the **Machines** tab.

    Click **Next**.

13. On the Assign Servers to Machines screen, assign the managed servers that will run on the machines you just created:

    Click on a machine in the right hand window.

    Click on the managed servers you want to run on that machine in the left window.

    Click on the arrow to assign the managed servers to the machine.

    Repeat these steps until you assign all managed servers to the appropriate machine.

    Assign servers to machines as follows:

    - ADMINHOST: **Admin Server**
    - WSSMHOST1: **wsonwls_server1**
    - WSSMHOST2: **wsonwls_server2**

    Click **Next**.

14. On the Configuration Summary screen, click **Create**.

15. Start Weblogic Administration Server by using `./startWeblogic.sh` in the new domain.

16. Start Managed Server. Switch to created domain directory subfolder `bin` and type `./startManagedWebLogic.sh` *managed server name* http://*wlsadminserver host*:*wls_adminserver_port*

    For example:

    ```
    ./startManagedWeblogic.sh wsonwls_server1 http://localhost:7001
    ```

On OESHOST2:

Use the pack and unpack commands to separate the domain directory that the OES WebService SM uses from the domain directory that the managed server in OESHOST1 uses.

See the procedure to separate the domain directory in Section 10.1.1.5, "Oracle Entitlements Server WebLogic Server Security Module High Availability."

10.2.8 Configuring Oracle Entitlements Server WebLogic Security Module High Availability

You can configure Oracle Entitlements Server WebLogic Security Module for high availability by means of a WebLogic cluster.

To configure Oracle Entitlements Server WebLogic Security Module:

On OESHOST1

1. Run `OESCLIENT_HOME/oessm/bin/config.sh` to create a WebLogic Security Module and a WebLogic Server domain.

   For example:

   ```
   ./config.sh -smType wls -smConfigId <wls_name> -serverLocation <wls_home>
   -pdServer <oes_admin_server> -pdPort <oes_admin_ssl_port>
   ```

2. On the Welcome screen, select **Create a WebLogic Domain** then click **Next**.
3. On the Select Domain Source screen, select **Generate a domain configured automatically to support the following added products**. From the list, select **Oracle Entitlements Server WebLogic Security Module on Weblogic For Managed Server**.

   Click **Next**.
4. On the Specify Domain Name and Location screen, enter the name and location for the domain and all its applications:
   - **Domain Name**: <*domain name*>
   - **Domain Location**: Accept the default entry.
5. On the Configure Administration Server Username and Password screen, enter the following:
   - **Name**: `weblogic`
   - **User Password**: Password for the WebLogic user
   - **Confirm User Password**: Password for the WebLogic user
   - **Description**: Description for the WebLogic user.
6. On the Configure Server Start Mode and JDK screen, select **Production Mode and JDK**.
7. On the Select Optional Configuration screen, select **AdminServer and Managed Servers**. Click **Next**.
8. On the Configure Administration Server screen, enter the following:
   - **Name**: `AdminServer`
   - **Listen address**: All Local Addresses
   - **Listen port**: `7001`

- **SSL listen port**: 7002

Select **SSL Enabled** then click **Next**.

9. On the Configure Managed Servers screen, the default managed server wlssm_server1 is created. Change the default managed server details and then add the second managed server:

   For the default managed server, enter these values:

   - **Name**: wlssm_server1
   - **Listen address**: WLSSMHOST1
   - **Listen port**: 14610
   - **SSL listen port**: 14611

   For the second managed server, click **Add** and enter these values:

   - **Name**: wlssm_server2
   - **Listen address**: WLSSMHOST2
   - **Listen port**: 14610
   - **SSL listen port**: 14611

10. In the Configure Clusters screen, click **Add** and enter wlssm_cluster. Select **unicast** for **Cluster messaging mode** then enter the Cluster address as *managed_server1:port,managed_server2: port*

    Click **Next**.

11. On the Assign Servers to Clusters screen, associate the managed servers with the cluster:

    - Click on the cluster wlssm_cluster in the right window.
    - Click on the managed server wlssm_server1 then click the arrow to assign it to the cluster.

      Repeat the preceding steps for the managed server wlssm_server2.

      Click **Next**.

12. On the Configure Machines screen, create a machine for each host in the topology.

    Click on the **Unix** tab for a host that uses a Unix operating system.

    For Admin Server Host:

    - **Name**: Name of your host. A good practice is to use the DNS name here.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

    Leave all other values at the default settings.

    Repeat the preceding steps for WLSSHOST1 and WLSSMOESHOST2 and enter the following values. Leave all other values at the default settings.

    - **Name**: Name of the host. A good practice is to use the DNS name.
    - **Node Manager Listen Address**: Oracle recommends that the machine IP address be identical to the DNS name of the machine.
    - **Node Manager Listen Port**: Enter a port for Node Manager to use.

For Unix operating systems, delete the default local machine entry under the **Machines** tab.

Click **Next**.

13. On the Assign Servers to Machines screen, you assign the managed servers that will run on the machines you just created. Follow these steps:

    Click on a machine in the right hand window.

    Click on the managed servers you want to run on that machine in the left window.

    Click on the arrow to assign the managed servers to the machine.

    Repeat these steps until you assign all managed servers to the appropriate machine.

    Assign servers to machines as follows:

    - ADMINHOST: **Admin Server**
    - WLSSMHOST1: **wlssm_server1**
    - W:SSMHOST2: **wlssm_server2**

    Click **Next**.

14. On the Configuration Summary screen, click **Create**.
15. Start Weblogic Administration Server by using `./startWeblogic.sh` in the new domain.
16. Start Managed Server. Switch to created domain directory subfolder `bin` and type `./startManagedWebLogic.sh` *managed server name* http://*wlsadminserver host*:*wls_adminserver_port*

    For example:

    ```
    ./startManagedWeblogic.sh wlssm_server1 http://localhost:7001
    ```

Use the pack and unpack commands to separate the domain directory that WebLogic Server Security Module uses from the one that the managed server in OESHOST1 uses.

To create a separate domain directory on OESHOST1:

1. Run the pack command to create a template pack as follows:

    ```
    cd MW_HOME/oracle_common/common/bin
    ```

    ```
    ./pack.sh -managed=true -domain=domain_path
    -template==domaintemplate.jar -template_name=domain_template
    ```

2. Run the unpack command to unpack the template in the managed server domain directory as follows:

    ```
    cd MW_HOME/oracle_common/common/bin
    ```

    ```
    ./unpack.sh -domain=new_domain_path -template=domaintemplate.jar
    ```

    Run the unpack operation on the remote hosts before you start the managed server, for example, OESHOST2.

3. Copy the file `domaintemplate.jar` from step 1. to OESHOST2.
4. Run unpack on the host on OESHOST2 using these commands:

    ```
    cd MW_HOME/oracle_common/common/bin
    ```

    ```
    ./unpack.sh -domain=domain_path -template==domaintemplate.jar
    ```

5. Start the managed server then switch to the domain directory subfolder `bin` that you created. Enter `./startManagedWebLogic.sh` *managed_server_name http://wlsadminserver host:wls_adminserver_port*

   For example:

   ```
   ./startManagedWeblogic.sh wlssm_server2 http://localhost:7001
   ```

10.2.9 Using RAC Datasource for Security Module in Controlled-pull Mode and Non-controlled Mode

Connection to policy store is used for Oracle Entitlements Server Security Modules in controlled-pull mode and non-controlled mode. Due to an SMConfig UI limitation, you must configure JDBC properties at the time that you create Security Module instances.

To use a RAC datasource in WebLogic Server Security Modules or Web Service Security Modules on WebLogic Server, run the following steps after you create a Security Module instance:

1. Log in to Weblogic Administrator Console of the domain that Security Module is deployed in. Configure the RAC datasource with database information identical to that of the Oracle Entitlements Server Administration Server.
2. Edit the Security Module configuration with the SMConfig UI:
   - Run `OES_CLIENT_HOME/oes_sm_instances/`*SM_Name*`/bin`
   - Run `oessmconfig.sh`.
   - Select **Database Configuration through JNDI Name** and enter the RAC datasource JNDI name into the **Data source JNDI Name** field. Click **Save & Close**.

10.2.10 Configuring Oracle Entitlements Server to Work with the Web Tier

This section describes how to configure Oracle Entitlements Server to work with the Oracle Web Tier and includes the following topics:

- Section 10.2.10.1, "Prerequisites"
- Section 10.2.10.2, "Configuring Oracle HTTP Servers to Front End the OES Managed Servers"
- Section 10.2.10.3, "Validate the Oracle HTTP Server Configuration"

10.2.10.1 Prerequisites

Verify that the following tasks have been performed:

1. Oracle Web Tier has been installed on WEBHOST1 and WEBHOST2.

   For instructions on installing Oracle HTTP Server on WEBHOST1 and WEBHOST2, see Section 8.5.3.5.1, "Installing Oracle HTTP Server for the Web Tier."
2. Oracle Entitlements Server has been installed and configured on OESHOST1 and OESHOST2.
3. The load balancer has been configured with a virtual hostname (`sso.`*example*`.com`) pointing to the web servers on WEBHOST1 and WEBHOST2.
4. The load balancer has been configured with a virtual hostname (`oesinternal.`*example*`.com`) pointing to web servers WEBHOST1 and WEBHOST2.

10.2.10.2 Configuring Oracle HTTP Servers to Front End the OES Managed Servers

1. On each of the web servers on WEBHOST1 and WEBHOST2, create a file `oes.conf` in the directory `ORACLE_INSTANCE/config/OHS/component/moduleconf`. This file must contain the following information:

   ```
   NameVirtualHost *:7777
   <VirtualHost *:7777>
   ServerName http://sso.example.com:7777
   RewriteEngine On
   RewriteOptions inherit
   UseCanonicalName On

   # OES admin console
    <Location /apm>
   SetHandler weblogic-handler
   WebLogicCluster oeshost1.example.com:14600,
   oeshost2.example.com:14600
   </Location>
   ```

2. Save the file on both WEBHOST1 and WEBHOST2.
3. Stop and start the Oracle HTTP Server instances on both WEBHOST1 and WEBHOST2.

10.2.10.3 Validate the Oracle HTTP Server Configuration

To validate that Oracle HTTP Server is configured properly:

1. In a web browser, enter the following URL for the Oracle Identity Manager Console:

   `http://sso.example.com:7777/apm`

2. In the APM login page, use weblogic user credentials to log in.

11

Configuring High Availability for Mobile and Social

This chapter describes the Oracle Access Management Mobile and Social high availability framework. Topics include the following:

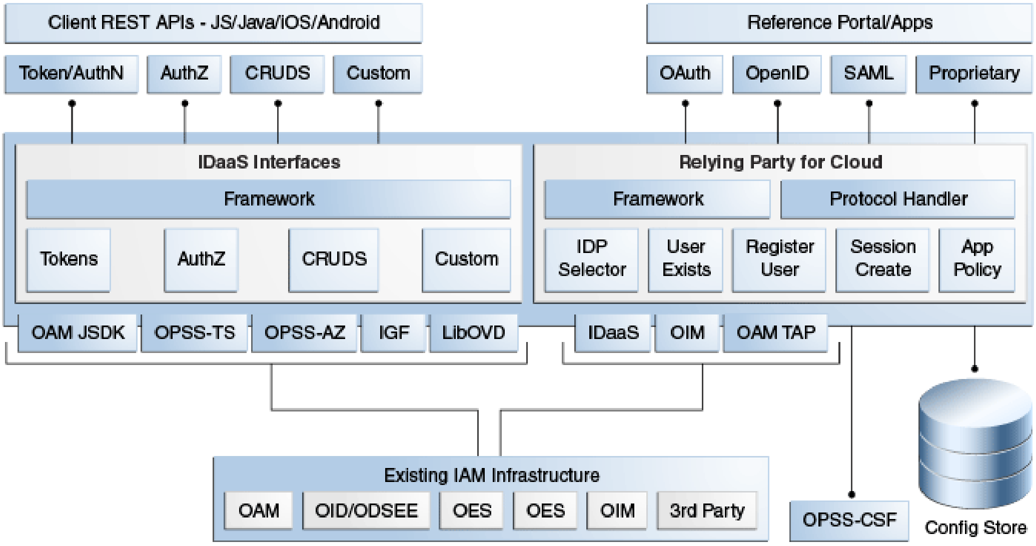
- Section 11.1, "Oracle Access Management Mobile and Social Component Architecture"
- Section 11.2, "Mobile and Social Component Characteristics"
- Section 11.3, "Mobile and Social High Availability Concepts"
- Section 11.4, "Configuring Mobile and Social High Availability"

11.1 Oracle Access Management Mobile and Social Component Architecture

Oracle Access Management Mobile and Social (Mobile and Social) is a lightweight security and identity solution that opens your existing Identity Management infrastructure to mobile and social networks. With Mobile and Social, you can securely expose a controlled view to the external world. Mobile and Social provides mechanisms for enterprise applications and portals to consume social media user identities inside the managed enterprise perimeter. Mobile and Social integrates with existing IDM products including OAM and IGF. See Oracle Access Management Mobile and Social in *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management* for more information on Mobile and Social.

Figure 11–1 shows the Mobile and Social component architecture.

Figure 11–1 Mobile and Social Component Architecture



The Oracle Access Management Mobile and Social service acts as an intermediary between a user who wants to access protected resources, and the back-end Access Management and Identity Management services that protect the resources. Mobile and Social provides simplified client libraries that allow developers to quickly add feature-rich authentication, authorization, and Identity capabilities to registered applications. On the back-end, the Mobile and Social server's pluggable architecture lets system administrators add, modify, and remove Identity and Access Management services without having to update user installed software.

11.1.1 Session State Information

No session state is recorded for the Mobile Services component. For Internet Identity Services, short-lived tokens are kept in memory and discarded as soon as they expire.

11.1.2 Component Lifecycle

Mobile and Social is a component in Access Suite J2EE application. You deploy and configure Mobile and Social as part of the Access Suite; the install, configuration, server instances are associated with the Access Suite.

As part of Mobile Services and Internet Identity Services, Mobile and Social provides HTTP interfaces for the clients to invoke. Mobile and Social processes those requests and returns a response, which the client may use to make additional requests. Mobile and Social is stateless, it can handle all requests sent by the client independently. Mobile and Social provides mobile device registration service and user authentication services using products like OAM, or by using social networks authentication and user profile services using Directory Servers.

Mobile and Social starts up as part of Access Suite server startup. MBean Server loads the Mobile and Social configuration.

11.1.3 Component Configuration Artifacts

Use the Administration Console or WLST commands to edit configuration files. Table 11–1 shows Mobile and Social configuration files and their location.

Table 11–1 Mobile and Social Component Configuration Files

File	Location
`Idaas.xml`	`<DOMAIN_HOME>/config/fmwconfig`
`oic_rp.xml`	`<DOMAIN_HOME>/config/fmwconfig`

11.1.4 Mobile and Social Deployment Artifacts

A Mobile and Social installation deploys the following components as part of `oam-server.ear`:

- `oic_rest.war`
- `oic_rp.war`

Table 11–2 shows Mobile and Social services deployment locations:

Table 11–2 Mobile and Social Product Deployment Locations

Mobile and Social Product	Deployment Location
Administration Server	Administration Server user interface deploys as part of the OAM Admin .ear, `oam-admin.ear`
Managed Server, REST, and Internet Identity Services runtime	Deploy as part of the OAM Server .ear file, `oam-server.ear`

11.2 Mobile and Social Component Characteristics

Mobile and Social services consists of two subcomponents, Mobile Services and Internet Identity Services. Mobile Services provide Representational state transfer (REST) interfaces for authentication, user profile, and authorization services. Internet Identity Services provides integration with social network authentication.

See Introduction to Mobile Services in *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management* for more information on Mobile and Social.

11.3 Mobile and Social High Availability Concepts

This section describes the Mobile and Social high availability architecture and its elements. Topics include the following:

- Section 11.3.1, "Mobile and Social High Availability Architecture"
- Section 11.3.1, "Mobile and Social High Availability Architecture"

11.3.1 Mobile and Social High Availability Architecture

Figure 11–1 shows Mobile and Social deployed in a high availability architecture in an active-active configuration.

Figure 11–2 Mobile and Social High Availability Architecture

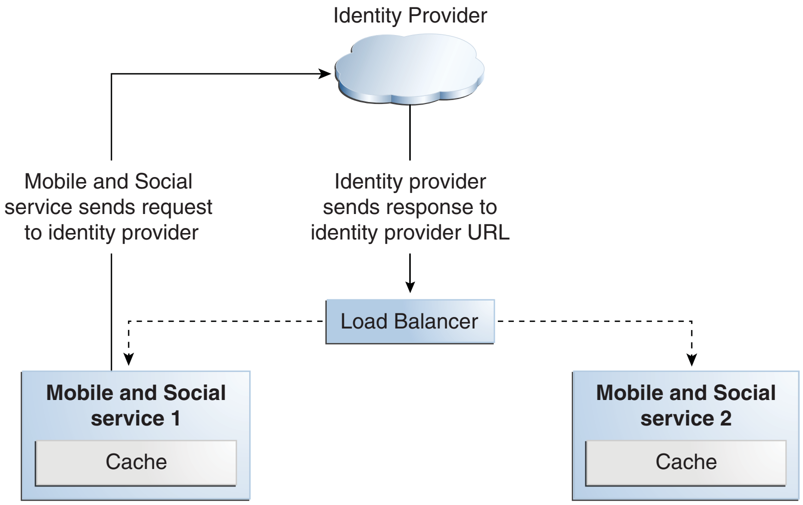

Identity Provider
Mobile and Social service sends request to identity provider
Identity provider sends response to identity provider URL
Load Balancer
Mobile and Social service 1
Cache
Mobile and Social service 2
Cache

Figure 11–2 shows a typical client server architecture that supports multi-instance deployments. In most cases, requests are stateless, requiring no persistence. Mobile and Social services relies on other products, such as OAM/OID, that may have their own high availability capability. For the cases where state is maintained (Internet Identity authentication requests), high availability is achieved through sticky load balancing.

Requests can go to either server because there is no database persistence for sessions or runtime data. The load balancer returns requests to either Mobile and Social service 1 or 2 based on the policy set, such as Round Robin.

When an application invokes Internet Identity Services, control goes to a social network site such as Google, Facebook, or an Internet identity provider to process the request. When the response returns from the identity provider, it must return to the same server that initiates the request. With multiple Mobile and Social node deployments and when access is through the load balancer, requests to Mobile and Social must be pinned to the same server using the load balancer sticky sessions feature. Mobile and Social can further process the request after the Mobile and Social server that initiates the request to an IDP receives the IDP response.

You deploy Mobile and Social applications to all members in a cluster. The Mobile and Social application does not notify other cluster members when it successfully deploys on a cluster.

11.3.2 Mobile and Social High Availability and Node Failover

This section describes elements of the Mobile and Social architecture that provide protection from node failure.

Note that if a node failover occurs, Mobile and Social follows standard WebLogic Server failover procedures. If node failure occurs before Mobile and Social completes a request that it receives from a client, the client receives an error through HTTP connections timeouts.

This section includes the following topics:

- Section 11.3.2.1, "Load Balancing Requirements and Characteristics"
- Section 11.3.2.2, "Session State Replication and Failover"

- Section 11.3.2.3, "Client Application Startup"
- Section 11.3.2.3.1, "Death Detection / Restart"

11.3.2.1 Load Balancing Requirements and Characteristics

Mobile and Social components are standard J2EE applications deployed on WebLogic Server 10.3, and conform to the standard for load balancing. Note the following:

- You must use sticky session routing for Mobile and Social/RP requests, however, external load balancers are supported.
- There is no intra-component load balancing.
- There are no timeout requirements because there are no persistent connections.

11.3.2.2 Session State Replication and Failover

The Mobile and Social high availability architecture relies on standard WebLogic Server clustering support for failover requirements. This architecture does not replicate the session state.

11.3.2.3 Client Application Startup

When an Mobile and Social instance starts up, it does not affect the running system state. The Mobile and Social instance does not affect other running components or cluster members other than becoming a participant in WebLogic Server clustering scenarios.

11.3.2.3.1 Death Detection / Restart Use the Node Manager for Java EE components and OPMN for system components for death detection and component restart.

11.4 Configuring Mobile and Social High Availability

Mobile and Social is part of the same managed server as Oracle Access Manager.

You can deploy Mobile and Social independently or with other components such as OAM, STS, and Identity Federation.

Note the following Mobile and Social configuration prerequisites:

- You must enable Mobile and Social if it is not enabled. Log into the Oracle Access Management Console, select the **System configuration** tab, open **Available Services**, and verify that the Mobile and Social status shows a green check.
- You must configure OHS for `oic_rest` and `oic_rp` by adding the following mapping to the `oam.conf` file in *ORACLE_INSTANCE*`/config/OHS/ohs1/moduleconf`:

```
<Location /oic_rest>
  SetHandler weblogic-handler
  WebLogicCluster
  oamhost1.example.com:14100,oamhost2.example.com:14100
</Location>

<Location /oic_rp>
  SetHandler weblogic-handler
  WebLogicCluster
  oamhost1.example.com:14100,oamhost2.example.com:14100
</Location>
```

12

Configuring High Availability for Oracle Privileged Account Manager Components

This chapter describes the Oracle Privileged Account Manager high availability framework. Topics include the following:

Oracle Privileged Account Manager manages *privileged* accounts that are not being managed by any other Oracle Identity Management components. Accounts are considered privileged if they can access sensitive data, can grant access to sensitive data, or can both access and grant access to that data.

For a more detailed overview of Oracle Privileged Account Manager and its features, see "Understanding Oracle Privileged Account Manager" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Privileged Account Manager*.

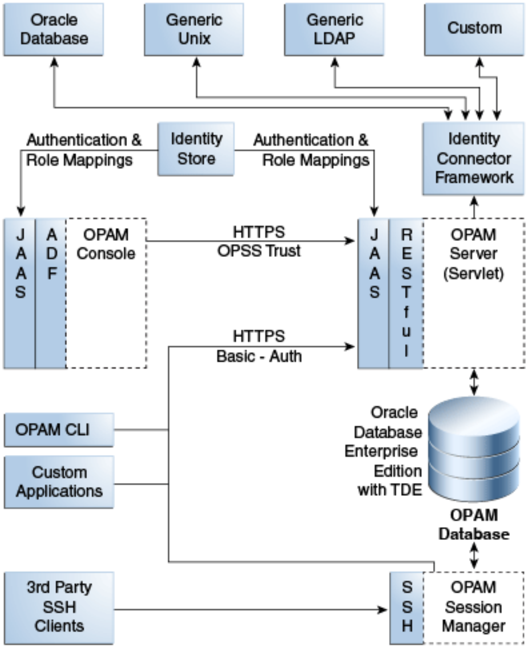
12.1 Oracle Privileged Account Manager Component Architecture

Oracle Privileged Account Manager (OPAM) consists of a server, user interface, and session manager component that run as web applications. The server and user interface are web applications. Session Manager is a standard J2EE application with no web interface at this time. The OPAM server runs in a WebLogic Managed Server. The OPAM user interface is part of Oracle Identity Navigator (OIN) and also runs on the OPAM WebLogic Managed Server. OPAM Session Manager also runs on the Managed Server

The default and most simple configuration of OPAM deployment involves running a single OPAM managed server in a WebLogic domain. OPAM uses its own schema to store passwords to targets, accounts, and other items.

The following diagram illustrates Oracle Privileged Account Manager's architecture and topology:

Figure 12–1 Oracle Privileged Account Manager Component Architecture



Note the following features of this architecture:

- All of Oracle Privileged Account Manager's core logic resides on the Oracle Privileged Account Manager Server. This functionality is exposed through a Representational state transfer (REST or RESTful) service, where the data is encoded as JavaScript Object Notation (JSON).

  Note: Oracle Privileged Account Manager provides a Web-based user interface in Oracle Identity Navigator and an Oracle Privileged Account Manager Command Line Tool. Both interfaces are essentially clients of the Oracle Privileged Account Manager Server.

  However, third parties can write their own clients, such as custom applications, by leveraging the open RESTful service. For more information, refer to Working with Oracle Privileged Account Manager's RESTful Interface in *Oracle Fusion Middleware Administrator's Guide for Oracle Privileged Account Manager*.

- OPAM Session Manager is an OPAM sub-component that empowers OPAM Session Management capabilities. It is a J2EE application that interacts with the OPAM server by means of the OPAM RESTful interfaces, and shares the same database that OPAM Server uses. In addition, the OPAM Session Manager listens and responds to SSH traffic to establish privileged sessions against SSH capable OPAM targets.
- Oracle Privileged Account Manager authentication relies on Java Authentication & Authorization Service (JAAS) support in WebLogic. Refer to "WebLogic Security Service Architecture" in *Oracle® Fusion Middleware Understanding Security for Oracle WebLogic Server* for more information about JAAS support in WebLogic.
- All communication with, and between, Oracle Privileged Account Manager-related components (including Oracle Privileged Account Manager's Web-based user interface, command-line interface, and server) all occur over SSL

- Oracle Privileged Account Manager relies on and transparently uses the ID Store and Policy Store configured for the WebLogic domain in which Oracle Privileged Account Manager is deployed.

  See How Oracle Privileged Account Manager is Deployed in Oracle Fusion Middleware in *Oracle Fusion Middleware Administrator's Guide for Oracle Privileged Account Manager* for more information.

- The Oracle Privileged Account Manager Web-based user interface leverages, and is rendered by, Oracle Application Development Framework (ADF).
- Oracle Privileged Account Manager connects to targets by using Identity Connector Framework (ICF) connectors. For additional information, see Understanding the Identity Connector Framework in the *Oracle® Fusion Middleware Developer's Guide for Oracle Identity Manager*.

This section includes the following topics:

- Section 12.1.1, "Runtime Processes"
- Section 12.1.2, "Process Lifecycle"
- Section 12.1.3, "Session State"
- Section 12.1.4, "External Dependencies"
- Section 12.1.5, "Deployment Artifacts"
- Section 12.1.6, "Log File Locations"

12.1.1 Runtime Processes

OPAM server is the server component that runs WebLogic Managed Server. The OPAM Session Manager is a component that also runs on the OPAM Weblogic Managed Server. The OPAM user interface is a component that runs on the OPAM WebLogic Managed Server. OPAM server uses the REST protocol to communicate with clients including OPAM user interface and command line client. OPAM server uses the OPAM database as its data store. You can deploy OPAM server and the user interface in a WebLogic cluster with multiple servers and use WebLogic Console to manage the processes.

12.1.2 Process Lifecycle

OPAM server and OPAM Session Manager are J2EE applications deployed in WebLogic server.

During startup, the OPAM Lifecycle Listener checks whether the keystore is present for that domain.

- If the keystore is not present, OPAM Lifecycle Listener creates a keystore using a randomly-generated keystore password, creates a keystore entry with domain name as alias, and updates the CSF with the keystore password.
- If the keystore is present, it retrieves the keystore password from CSF and checks whether an entry is present with domain name as alias.

Because there is no operation if the keystore and required keystore CSF entry are present, a high availability environment for OPAM does not affect the process lifecycle.

OPAM Server uses a servlets model in which RESTful interfaces service clients. Therefore, the component lifecycle does not affect other running instances.

For more information on CSF, see Oracle Privileged Account Manager-Managed CSF Credentials in the *Oracle Fusion Middleware Administrator's Guide for Oracle Privileged Account Manager*.

12.1.3 Session State

OPAM server uses the REST-based communication; clients use HTTPS URLs. Session state information is not stored. Each communication completes an operation; sessions are not preserved.

OPAM Session Manager maintains the session state for each privileged session that it facilitates. After a session is established, the OPAM Session Manager instance used at session initiation time must continue it. If the OPAM Session Manager fails mid-session, a background thread cleans up orphaned sessions periodically.

12.1.4 External Dependencies

OPAM server dependencies include the following:

- ICF Connectors, for communication with target systems
- External ID stores, for authentication (configurable in WebLogic)

The OPAM user interface and OPAM Command Line tool depend on the OPAM server. Client connections, which are REST using HTTPS URL, are short lived.

12.1.5 Deployment Artifacts

When you deploy OPAM to a managed server, Oracle Identity Navigator deploys to the Administration Server. The Oracle JRF deploys to both the Administration Server and Managed Server. Table 12–1 describes where OPAM product artifacts deploy.

Table 12–1 OPAM Deployment Artifacts

Location	Product Artifact
Administration Server	Oracle JRF
Managed Server	OPAM server .ear file, OPAM session manager .ear file, Oracle Identity Navigator .ear file, Oracle JRF

12.1.6 Log File Locations

WebLogic Server Logs and Audit Logs save to the DOMAIN_HOME.

Audit Logs, if configured in WebLogic, save to the Audit database.

12.2 Oracle Privileged Account Manager High Availability Concepts

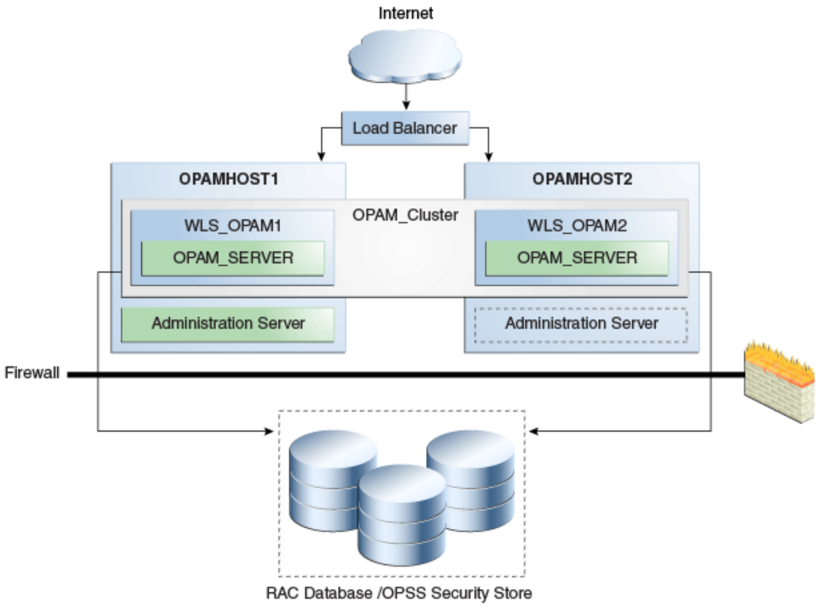
At the cluster level, you can change OPAM properties such as connector directory and time limits.

All servers connect to a common domain OPAM data store, where configuration is maintained. Therefore, all connected servers pick up the shared data.

12.3 Oracle Privileged Account Manager High Availability Architecture

Figure 12–2 shows Oracle Privileged Account Manager deployed in a high availability architecture in an active-active configuration.

Figure 12–2 Oracle Privileged Account Manager in a High Availability Architecture



In this configuration, the Oracle Privileged Account Manager Servers are part of the same domain and use the same OPSS security store. Because multiple servers interact with the OPSS security store, Oracle recommends using the database as the OPSS backend.

In this cluster configuration, the same data is available from the different servers in the cluster. Each managed server has its own port configuration. The configuration includes a load balancer.

OPAMHOST1 has the following installations

- An Oracle Privileged Account Manager instance in the WLS_OPAM1 Managed Server. The Oracle RAC database has been configured in a JDBC multi data source or GridLink data source to protect the instance from Oracle RAC node failure.
- A WebLogic Server Administration Server. Under normal operations, this is the active Administration Server.

OPAMHOST2 has the following installations:

- An Oracle Privileged Account Manager in the WLS_OPAM2 Managed Server. The Oracle RAC database is configured in a JDBC multi data source tor GridLink data source to protect the instance from Oracle RAC node failure.

  The instances in the WLS_OPAM2 Managed Server and the instances in the WLS_OPAM1 Managed Server are configured as the CLUSTER_OPAM cluster.
- A WebLogic Server Administration Server. Under normal operations, this is the passive Administration Server. You make this Administration Server active if the Administration Server on HOST1 becomes unavailable. See Section 15, "Active-Passive Topologies for Oracle Fusion Middleware High Availability" for more information on the active-passive Administration Server configuration.

12.3.1 Starting and Stopping the Cluster

By default, WebLogic Server starts stops, monitors, and manages lifecycle events for the application. The Oracle Privileged Account Manager application leverages the high availability features of the underlying Oracle WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic Servers. In case of failure, Node Manager restarts the WebLogic Server.

In high availability environments, the state and configuration information is stored a database that is shared by all the members of the cluster.

12.4 Oracle Privileged Account Manager High Availability and Node Failure

Note the following characteristics about Oracle Privileged Account Manager node failure in a high availability configuration:

- Node failure does not affect clients.
- If a server fails, ongoing REST-based operations stop; they do not fail over. Clients can connect to available servers and retry or continue with their requests. If you include a load balancer in the deployment, it directs requests to available servers.

12.5 Oracle Privileged Account Manager High Availability Configuration

This section provides high-level instructions for setting up a maximum high availability deployment for Oracle Privileged Account Manager High Availability. Topics in this section include the following:

- Section 12.5.1, "Appropriate Development Environment"
- Section 12.5.2, "Components Deployed"
- Section 12.5.3, "Dependencies"
- Section 12.5.4, "High Availability Configuration Procedure"
- Section 12.5.5, "OHS Load Balancer Configuration"

12.5.1 Appropriate Development Environment

Perform the configuration in this topic if you want to configure Oracle Privileged Account Manager with Oracle Identity Navigator in a new WebLogic domain and then run the Oracle Identity Navigator discovery feature. This feature populates links to the product consoles for Oracle Identity Manager, Oracle Access Management, Oracle Adaptive Access Manager, and Oracle Privileged Account Manager. You can then access those product consoles from within the Oracle Identity Navigator interface, without having to remember the individual console URLs.

12.5.2 Components Deployed

Performing the configuration in this section deploys the Oracle Privileged Account Manager and Oracle Identity Navigator applications on a new WebLogic Administration Server.

12.5.3 Dependencies

The configuration in this section depends on the following:

- Oracle WebLogic Server

- Installation of the Oracle Identity and Access Management 11g software

For more information, see Installing and Configuring Oracle Identity and Access Management (11.1.2.0.0) in the *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management*.

12.5.4 High Availability Configuration Procedure

This section includes the following steps:

- Section 12.5.4.1, "Configuring Oracle Identity and Access Management on OPAMHOST1"
- Section 12.5.4.2, "Starting OPAMHOST1"
- Section 12.5.4.3, "Configuring OPAM on OPAMHOST2"
- Section 12.5.4.4, "Starting OPAMHOST2"

12.5.4.1 Configuring Oracle Identity and Access Management on OPAMHOST1

To configure Oracle Privileged Account Manager and Oracle Identity Navigator for maximum high availability:

1. Install Oracle WebLogic Server and create a Middleware Home. See WebLogic Server and Middleware Home Requirements in *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for more information.
2. Install the Oracle Identity and Access Management 11*g* software. See Installing Oracle Identity and Access Management (11.1.2.0.0) in *Oracle Fusion Middleware Installation Guide for Oracle Identity and Access Management* for more information.
3. Start the Configuration Wizard by running this command:

   ```
   ORACLE_HOME/oracle_common/common/bin/config.sh
   ```

   > **Note:** You must run the config.sh script from your Oracle Identity and Access Management Home directory that contains Oracle Identity Manager, Access Manager, Oracle Adaptive Access Manager, Oracle Entitlements Server, Oracle Privileged Account Manager, Mobile and Social, and Oracle Identity Navigator.

4. On the Welcome screen, select **Create a new WebLogic domain** and click **Next**. The Select Domain Source screen appears.
5. On the Select Domain Source screen, ensure that the **Generate a domain configured automatically to support the following products:** option is selected. Select **Oracle Privileged Account Manager - 11.1.2.0.0 [IAM_Home]**.

   > **Note:** When you select the **Oracle Privileged Account Manager - 11.1.2.0.0 [IAM_Home]** option, the Oracle Identity Navigator, Oracle JRF, and Oracle Platform Security Service options are also selected by default.

   Click **Next**. The Specify Domain Name and Location screen appears.
6. Enter the domain name **IDM_Domain**. Keep the default **Domain Location** and **Application Directory**. Click **Next**. The Configure Administrator User Name and Password screen appears.

7. Configure a user name and a password for the administrator. The default user name is **weblogic**. Click **Next**.

8. Choose `JRockit SDK 1.6.0_`<*version*> and Production Mode in the Configure Server Start Mode and JDK screen of the Oracle Fusion Middleware Configuration Wizard.

9. In the Configure JDBC Component Schema screen, enter the database schema details for the OPAM and OPSS schema.

   > **Note:** You can also use GridLink data sources. See Section 3.13, "GridLink Data Sources" for more information.

   (Optional) Select an option for RAC configuration for component schemas.

   Click **Next**.

10. On the Test Component Schema screen, the Configuration Wizard tries to validate the data sources. Validate that the test for all schemas completes successfully

    If the data source validation succeeds, click **Next**.

    If the data source validation fails, click **Previous**, correct the issue, then try again.

11. On the Select Optional Configuration screen, select the following:

    - **Administration Server**
    - **Managed Servers, Clusters, and Machines**
    - Deployments and Services

    Click **Next**.

12. In the Customize Server and Cluster configuration screen, select **Yes** then **Next**.

13. In the Configure the Administration Server screen, enter the following values:

    - **Name:** `AdminServer`
    - **Listen address**: `OPAMHOST1.example.com`
    - **Listen port:** `7001`

      Do not set or change the following parameters:

    - **SSL listen port:** Not applicable
    - **SSL enabled** or **disabled**

    Click **Next**.

14. On the Configure Managed Servers screen, create an entry for each OPAMHOST in the topology, that is, one entry for the OPAM server running on OPAMHOST1 and one entry for the OPAM server running on OPAMHOST2.

    Select the OPAM_SERVER entry and change the entry to the following values:

    - **Name**: `opamserver1`
    - **Listen Address**: `OPAMHOST1`*.example.com*
    - **Listen Port**: `18101`
    - **SSL Port**: `18102`

    For the second OPAM_SERVER, click **Add** and enter the following values:

- **Name**: `opam_server2`
- **Listen Address**: `OPAMHOST2`*.example.com*
- **Listen Port**: `18101`
- **SSL Port:** `18102`

Click **Next**.

15. In the Configure Clusters screen, click **Add** to create a cluster.

    Enter a name for the cluster, such as `OPAM_Cluster`. Leave all other fields at their default setting.

    Click **Next**.

16. On the Assign Servers to Clusters screen, associate the managed servers with the cluster:

    Click on the cluster name **OPAM_Cluster** in the right window.

    Click on the managed server **opam_server1** then click the arrow to assign it to the cluster.

    Repeat the preceding steps for the managed server **opam_server2**.

    Click **Next**.

17. On the Configure Machines screen, create a machine for each host in the topology.

    Click on the **Unix** tab if your hosts use a Unix operating system or click **Machines**.

    Provide the following information:

    - **Name**: Name of the host. A good practice is to use the DNS name here.
    - **Node Manager Listen Address**: Enter the DNS name of the machine here.
    - **Node Manager Port**: Enter a port for Node Manager to use.

    Repeat the preceding steps for OPAMHOST2 and enter the following values:

    - **Name**: Name of the host. A good practice is to use the DNS name, `OPAMHOST2`
    - **Node Manager Listen Address**: Enter the DNS name of the machine, `OPAMHOST.example.com`
    - **Node Manager Port**: Enter a port for Node Manager to use.

    Click **Next**.

18. On the Assign Servers to Machines screen, assign the managed servers that will run on the machines you just created. Follow these steps:

    Click on a machine in the right hand window.

    Click on the managed servers you want to run on that machine in the left window.

    Click on the arrow to assign the managed servers to the machine.

    Repeat these steps until all managed servers are assigned to the appropriate machine.

    For example:

    - Host1: **Admin Server, OPAMHOST1, and opam_server1**
    - Host2: **OPAMHOST2 and opam_server2**

    Click **Next**.

19. On the Configuration Summary screen, click **Create**.

12.5.4.1.1 Configuring the Database Security Store You must configure the Database Security Store after you configure the domain but before you start the Administration Server. See Section 12.5.4.1.1, "Configuring the Database Security Store" for more information.

12.5.4.1.2 Starting Administration Server on OPAMHOST1 To start the Administration Server:

1. Go to `DOMAIN_HOME/bin` directory. Run `startWeblogic.sh`.
2. At the prompt, enter the WebLogic administrator username and password.

> **Note:** You perform the following steps when you start the Administration Server for the first time only.

> **Note:** Before proceeding to the next step to run `opam-config.sh`, you must set the `ORACLE_HOME`, `JAVA_HOME` and `ANT_HOME` environment variables. `ANT_HOME` must point to a location that contains Ant and JAR binary files such as *middleware_home*`/modules/org.apache.ant_1.7.1`.

3. After the Administration Server is running, go to `ORACLE_HOME/opam/bin` directory. Run `opam-config.sh`.
4. At the prompt, enter the WebLogic username, password, URL, domain name, and middleware home.
5. Restart the Administration Server after `opam-config.sh`.

12.5.4.2 Starting OPAMHOST1

This section describes the steps for starting OPAMHOST1:

1. Before you can start managed servers from the console, you must create a Node Manager property file. Run `setNMProps.sh` located in the `MW_HOME/oracle_common/common/bin` directory.

   ```
   MW_HOME/oracle_common/common/bin/setNMProps.sh
   ```

2. Start Node Manager with the command `MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh`.

12.5.4.2.1 Starting Oracle Privileged Account Manager on OPAMHOST1 To start Oracle Privileged Account Manager on OPAMHOST1, follow these steps:

1. Log into the WebLogic Administration Console using this URL:

   ```
   http://opamhost1.example.com:7001/console
   ```

2. Enter the WebLogic administrator username and password.
3. Select **Environment - Servers** from the **Domain Structure** menu.
4. Click the Control tab.
5. Click the server **opam_server1**.

6. Click **Start**.
7. Click **OK**.

12.5.4.3 Configuring OPAM on OPAMHOST2

After the configuration succeeds on OPAMHOST1, you can propagate it to OPAMHOST2. You do this by packing the domain using the pack script on OPAMHOST1 and unpacking the domain using the unpack script on OPAMHOST2.

Both scripts reside in the `MW_HOME/oracle_common/common/bin` directory.

1. Verify that `MW_HOME` and `ORACLE_HOME` directory structure is identical to the `OPAMHOST1` directory structure.
2. Enter the following on OPAMHOST1 to create the file `idm_domain.jar` in the `/tmp` directory:

   ```
   pack.sh -domain=$MW_HOME/user_projects/domains/IDM_Domain
    -template=/tmp/idm_domain.jar -template_name="OPAM Domain" -managed=true
   ```

3. The previous step created the `idm_domain.jar` file in the `/tmp` directory.

   Copy the `idm_domain.jar` file from OPAMHOST1 to a temporary directory on OPAMHOST2.
4. To copy `idm_domain.jar` to OPAMHOST2, enter the following:

   ```
   unpack.sh -domain=MW_HOME/user_projects/domains/IDM_Domain
   -template=/tmp/idm_domain.jar
   ```

5. Copy `jps-wls-trustprovider.jar` from `MW_HOME/oracle_common/modules/oracle.jps_11.1.1` to `WLS_SERVER_HOME/server/lib/mbeantypes`.

12.5.4.4 Starting OPAMHOST2

This section describes the steps for starting OPAMHOST2.

1. Before you can start managed servers from the console, you must create a Node Manager property file. Run the script `setNMProps.sh` located in the directory `MW_HOME/oracle_common/common/bin`.
2. Start Node Manager with the command `MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh`
3. Start Oracle Privileged Account Manager on OPAMHOST2. See Section 12.5.4.2.1, "Starting Oracle Privileged Account Manager on OPAMHOST1" and replace HOST1 and server1 with HOST2 and server2.

12.5.5 OHS Load Balancer Configuration

You can load balance OPAM servers using Oracle HTTP Server (OHS). This section provides the steps to configure OPAM to work with OHS. Topics include the following:

- Section 12.5.5.1, "Configure SSL"
- Section 12.5.5.2, "Update the Oracle HTTP Server Configuration"
- Section 12.5.5.3, "Restart the Oracle HTTP Server"

12.5.5.1 Configure SSL

Because OPAM servers use SSL for communication, you must configure SSL options in the OHS Load Balancer. Follow these steps:

1. Enable the outbound SSL connections from OHS so that they can communicate with the OPAM managed/Administration servers. To do this, see Enable SSL for Outbound Requests from Oracle HTTP Server in the *Oracle Fusion Middleware Administrator's Guide*.
2. Enable inbound SSL connections into the OPAM managed/Administration servers. See Inbound SSL to Oracle WebLogic Server in the *Oracle Fusion Middleware Administrator's Guide*.

12.5.5.2 Update the Oracle HTTP Server Configuration

On the host where OHS is installed, create a file named `opam.conf` in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

Create the file with the following lines:

```
NameVirtualHost *:4443
<VirtualHost *:44443>

  ServerName http://opam.example.com:4443
  ServerAdmin user@example.com
  RewriteEngine On
  RewriteOptions inherit

   <Location /opam>
    SetHandler weblogic-handler
    WebLogicCluster opamhost1.example.com:18102,opamhost2.example.com:18102
    WLCookieName OPAMSESSIONID
    WlSSLWallet "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_
    NAME}/keystores/ohs_wallet"
    WLProxySSL ON
    WLProxySSLPassThrough ON
    SecureProxy On
  </Location>

</VirtualHost>
```

To configure OHS for the OPAM console, update the `opam.conf` file with the following lines:

```
NameVirtualHost *:7777
<VirtualHost *:7777>

  ServerName http://opam.example.com:7777
  ServerAdmin user@example.com
  RewriteEngine On
  RewriteOptions inherit

   <Location /oinav>
    SetHandler weblogic-handler
    WebLogicCluster opamhost1.example.com:18101,opamhost2.example.com:18101
    WLCookieName OPAMSESSIONID
   </Location>
```

```
</VirtualHost>
```

12.5.5.3 Restart the Oracle HTTP Server

To restart Oracle HTTP Server, run the following commands from `ORACLE_INSTANCE/bin`:

```
opmnctl stopall
opmnctl startall
```

13

Configuring High Availability for Oracle Identity Navigator

Oracle Identity Navigator is an administrative portal designed to act as a launch pad for Oracle Identity Management products. It does not replace the individual product consoles. Rather, it enables you to access the Oracle Identity Management consoles from one site. For more details on Oracle Identity Navigator, refer to *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Navigator*

Oracle Identity Navigator is deployed on the OPAM managed server (`opam_server1`) in an out-of-the-box scenario.

You can configure Oracle Identity Navigator in a clustered configuration same as Oracle Privileged Account Manager. However, you do not need to run the `opam-config.sh` script when configuration is complete.

To configure OHS for Oracle Identity Navigator, complete the following steps:

- Section 13.1, "Update the Oracle HTTP Server Configuration"
- Section 13.2, "Restart the Oracle HTTP Server"

13.1 Update the Oracle HTTP Server Configuration

On the host where OHS is installed, create a file named `oinav.conf` in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

Create the file with the following lines:

```
NameVirtualHost *:7777
<VirtualHost *:7777>

  ServerName http://oinav.example.com:7777
  ServerAdmin user@example.com
  RewriteEngine On
  RewriteOptions inherit

   <Location /oinav>
    SetHandler weblogic-handler
    WebLogicCluster oinavhost1.example.com:18101,oinavhost2.example.com:18101
    WLCookieName OPAMSESSIONID
  </Location>

</VirtualHost>
```

13.2 Restart the Oracle HTTP Server

To restart Oracle HTTP Server, run the following commands from ORACLE_INSTANCE/bin:

```
opmnctl stopall
opmnctl startall
```

14 Oracle Unified Directory

Oracle Unified Directory is Oracle's comprehensive, next-generation directory service that is designed to address large deployments, provide high performance, and be highly extensive. Oracle Unified Directory is also designed to be easy to deploy, manage, and monitor.

For more information on Oracle Unified Directory High Availability Deployments, see "Understanding Oracle Unified Directory High Availability Deployments" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Unified Directory*.

15

Active-Passive Topologies for Oracle Fusion Middleware High Availability

This chapter describes how to configure and manage active-passive topologies. It contains the following sections:

- Section 15.1, "Oracle Fusion Middleware Cold Failover Cluster Topology Concepts"
- Section 15.2, "Configuring Oracle Fusion Middleware for Active-Passive Deployments"
- Section 15.3, "Oracle Fusion Middleware Cold Failover Cluster Example Topologies"
- Section 15.4, "Transforming the Administration Server in an Existing Domain for Cold Failover Cluster"

15.1 Oracle Fusion Middleware Cold Failover Cluster Topology Concepts

Oracle Fusion Middleware provides an active-passive model for all its components using Oracle Fusion Middleware Cold Failover Cluster (Cold Failover Cluster). In a Cold Failover Cluster configuration, two or more managed server instances are configured to serve the same application workload but only one is active at any particular time.

You can use a two-node Cold Failover Cluster to achieve active-passive availability for middle-tier components. In a Cold Failover Cluster, one node is active while the other is passive, on standby. If the active node fails, the standby node activates and the middle-tier components continue servicing clients from that node. All middle-tier components fail over to the new active node. No middle-tier components run on the failed node after the failover.

The most common properties of a Cold Failover Cluster configuration include:

- **Shared Storage**: Shared storage is a key property of a Cold Failover Cluster. The passive Oracle Fusion Middleware instance in an active-passive configuration has access to the same Oracle binaries, configuration files, domain directory, and data as the active instance. You configure this access by placing these artifacts in storage that all participating nodes in the Cold Failover Cluster configuration can access. Typically the active node has shared storage mounted, while the passive node's is unmounted but accessible if the node becomes active. The shared storage can be a dual-ported disk device accessible to both the nodes or a device-based storage such as a NAS or a SAN. You can install shared storage on a regular file system. With Cold Failover Cluster, you mount the volume on one node at a time.

- **Virtual hostname**: In a Cold Failover Cluster solution, two nodes share a virtual hostname and a virtual IP. (The virtual hostname maps to the virtual IP and is used interchangeably in this guide.) However, only the active node can use this virtual IP at any one time. When the active node fails and the standby node is made active, the virtual IP moves to the new active node. The new active node now services all requests through the virtual IP. The Virtual Hostname provides a single system view of the deployment. A Cold Failover Cluster deployment is configured to listen on this virtual IP. For example, if the two physical hostnames of the hardware cluster are `node1.example.com` and `node2.example.com`, the name `cfcvip.example.com` provides the single view of this cluster. In the DNS, `cfcvip.example.com` maps to the virtual IP, which floats between node1 and node2. When a hardware cluster is used, it manages the failover of the virtual IP without the middle tier clients detecting which physical node is active and actually servicing requests.
- **Hardware Cluster**: Typically, a Cold Failover Cluster deployment uses a hardware cluster. The hardware cluster addresses the management of shared storage and virtual IP in its architecture. It plays a role in reliable failover of these shared resources, providing a robust active-passive solution. Most Cold Failover Cluster deploy to hardware clusters that include the following:
  - Two nodes that are in the same subunit,
  - A high-speed, private interconnect between the two nodes,
  - Public network interfaces, on which the client requests are served and the virtual IP is enabled.
  - Shared storage accessible by the two nodes. This includes shared storage that acts as a quorum device and shared storage for Oracle Fusion Middleware and database installs.
  - Clusterware running to manage node and component failures
- **Planned Switchover and Unplanned Failover**: A typical Cold Failover Cluster deployment is a two-node hardware cluster. To maximize utilization, both nodes typically have some elements of the deployment running, with the other node acting as a backup node for the appropriate element if needed. For example, a deployment may have the application tier (WebLogic container) running on one node and the Web tier (Oracle HTTP Server) running on the other node. If either node is brought down for maintenance or if either node fails, the surviving node hosts the services of the down node while continuing to host its current services.

  The high-level steps for switch-over to the standby node are as follows:

  1. Stop the middle-tier service on the primary node if the node is still available.
  2. Fail over the virtual IP from the current active node to the passive node. Bring it down on the current node then enable it and bring it up on the passive node.
  3. Fail over the shared disk from the current active node to the passive node. This involves unmounting the shared disk from the current node and mounting it on the passive node.
  4. Start the middle-tier service on the passive node, which becomes active.

  To manage failover, you run the planned switchover steps manually; both the detection of the failure and the failover itself is manual.

In active-passive deployments, services are typically down for a short period of time. This is the time taken to either restart the instance on the same node, or to fail over the instance to the passive node.

Active-Passive Topologies: Advantages

- Increased availability

  If the active instance fails or must be taken offline, an identically configured passive instance is ready to take over at any time. This provides an increased level of availability than a normal single instance deployment. Also, active-passive deployments provide availability and protection against planned and unplanned maintenance operation of the hardware. If you must bring down a node for planned maintenance, you can bring up middleware services on the passive node. You switch back to the old node at appropriate times.

- Reduced operating costs

  In an active-passive configuration only one set of processes is up and serving requests. Managing the active instance is generally less costly than managing an array of active instances.

- A Cold Failover Cluster topology is less difficult to implement because it does not require a load balancer, which is required in active-active topologies
- A Cold Failover Cluster topology is less difficult to implement than active-active topologies because you are not required to configure options such as load balancing algorithms, clustering, and replication.
- Active-passive topologies better simulate a one-instance topology than active-active topologies.
- Application independence

  Some applications may not be suited to an active-active configuration. This may include applications which rely heavily on application state or on information stored locally. Singleton applications are more suitable for active-passive deployments. An active-passive configuration has only one instance serving requests at any particular time.

Active-Passive Topologies: Disadvantages

- Active-passive topologies do not scale as well as active-active topologies. You cannot add nodes to the topology to increase capacity.
- State information from HTTP session state and EJB stateful session beans is not replicated and is lost when a node terminates unexpectedly. Such state can be persisted to the database or to the file system residing on a shared storage. However, this requires additional overhead that may affect the single node Cold Failover Cluster deployment performance.
- Active-passive deployments have a shorter downtime than a single node deployment. However, downtime is much shorter in an active-active deployment.

15.2 Configuring Oracle Fusion Middleware for Active-Passive Deployments

Oracle Fusion Middleware components come in a variety of Java EE container-deployed components and non-Java EE components. Oracle Internet Directory, Oracle Virtual Directory, and Oracle Reports are system components. Oracle SOA Suite and Oracle WebCenter Portal are Java EE components that are deployed to Oracle WebLogic Server.

Administration Console and Oracle Enterprise Manager Fusion Middleware Control also deploy to the WebLogic container. You can deploy both Java EE and system components to Cold Failover Cluster environments; they can co-exist on the same system or on different systems. When on the same system, you can configure them to failover as a unit, sharing the same virtual IP, or failover independently using separate virtual IPs. In most Oracle Fusion Middleware deployments, a database is used either for the component metadata created using Repository Creation Utility (RCU), or for application data. In many cases, a Cold Failover Cluster middle tier deployment uses a Cold Failover Cluster database, both deployed to the same cluster. The typical deployment has the two components configured as separate failover units using different VIPs and different shared disks on the same hardware cluster.

To create an active-passive topology for Oracle Fusion Middleware:

1. Install the component as a single instance configuration. If you plan to transform this instance to a Cold Failover Cluster deployment, install it using a shared disk. That is, the Middleware home, the Instance home (for system components) and the domain directory (for a WebLogic deployment on a shared disk). Everything that fails over as a unit should be on a shared disk.
2. After the installation, transform the deployment into a Cold Failover Cluster deployment and configure it to listen on a Virtual IP. The Virtual IP is configured on the current active node. It fails over, along with the Oracle Fusion Middleware deployment, to the passive node when failure occurs.

This general procedure applies to the Cold Failover Cluster Oracle database. For example, the Oracle database instance is installed as a single instance deployment and subsequently transformed for Cold Failover Cluster. A Cold Failover Cluster Oracle Fusion Middleware deployment can also use an Oracle Real Application Clusters (Oracle RAC) database.

The following sections describe the procedures for post-installation configuration to transform a single instance deployment to a Cold Failover Cluster deployment.

The rest of this chapter describes how to transform Cold Failover Cluster for each individual component in the Oracle Fusion Middleware suite. The first section details the procedure for the basic infrastructure components, and the subsequent section does so for the individual Oracle Fusion Middleware component. Any given deployment, for example an Oracle instance or domain, has more than one of these in a machine. To transform the entire instance or domain:

- Decide which components form a unit of failover.
- Deploy them on the same shared disk.

> **Note:** For details about installing and deploying Oracle Fusion Middleware components, see the installation guide for the specific Fusion Middleware component.

- Determine a virtual IP to use for this unit of failover. Typically, a single virtual IP is used for all the components, but separate IPs can be used as long as all of them fail over together.
- Apply the transformation procedure to each of the individual components to transform the deployment as a whole. Since more than one of these sections will apply for Cold Failover Cluster transformation of an installation, the order of transformation should always be as follows:
  - Transform the Administration Server or Enterprise Manager instance (if applicable).

  - Transform all managed servers in the deployment.
  - Transform the Oracle instances (non-Java EE deployments).

This section includes the following topics:

- Section 15.2.1, "Cold Failover Cluster Requirements"
- Section 15.2.2, "Directories and Environment Variables Terminology"
- Section 15.2.3, "Transforming Oracle Fusion Middleware Infrastructure Components"
- Section 15.2.4, "Transforming Oracle Fusion Middleware Components"
- Section 15.2.5, "Additional Actions for Fusion Middleware Failover"
- Section 15.2.6, "Transforming an Oracle Database"

15.2.1 Cold Failover Cluster Requirements

A Cold Failover Cluster deployment has at least two nodes. You install on one of these nodes; the other node is the passive node. Requirements for both nodes are as follows:

- The nodes should be same in all respects at the operating system level. For example, they should be the same operating system, version, and patch level.
- The nodes should have similar hardware characteristics. This ensures predictable performance during normal operations and on failover. Oracle suggests designing each node for capacity to handle both its normal role and the additional load required to handle a failover scenario. This is not required only if the SLA agreement indicates that, during outage scenarios, reduced performance is acceptable.
- The nodes should have the same mount point free so that mounting shared storage can occur to the same node during normal operations and failover conditions.
- The user ID and group ID on the two nodes are similar, and the user ID and group ID of the user owning the instance is the same on both nodes.
- The `oraInventory` location is the same on both nodes and has similar accessibility of the instance or domain owner. The location of the `oraInst.loc` file, as well as the `beahomelist` file should be the same.
- Since a given instance uses the same listen ports irrespective of the machine on which it is currently active, ensure that the ports that the Cold Failover Cluster instance uses are free on both the nodes.

> **Note:** Before you start the transformation, back up the entire domain. Oracle also recommends that you create a local backup file before editing the source files. For more information, see the *Oracle Fusion Middleware Administrator's Guide*. Oracle recommends backing up:
>
> - All domain directories
> - All Instance homes
> - The database repository (optional)
> - Middleware homes (optional)

15.2.2 Directories and Environment Variables Terminology

The following list describes the directories and variables used in this chapter:

- ORACLE_BASE: This environment variable and related directory path refers to the base directory under which Oracle products are installed.
- MW_HOME: This environment variable and related directory path refers to the location where Fusion Middleware (FMW) resides.
- WL_HOME: This environment variable and related directory path contains installed files necessary to host a WebLogic Server.
- ORACLE_HOME: This environment variable and related directory path refers to the location where a specific Oracle FMW Suite, such as SOA, is installed.
- ORACLE_COMMON_HOME: The Oracle home that contains the binary and library files that are common to all the Oracle Fusion Middleware software suites. In particular, the Oracle Common home includes the files required for Oracle Enterprise Manager Fusion Middleware Control (which is used to manage the Fusion Middleware) and the Oracle Java Required Files (JRF).
- DOMAIN_HOME: This directory path refers to the location where the Oracle WebLogic Domain information (configuration artifacts) is stored.
- ORACLE_INSTANCE: An Oracle instance contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. An Oracle instance directory contains updatable files, such as configuration files, log files, and temporary files.
- /localdisk: The root of the directory tree when the FMW install (either MW_HOME or DOMAIN_HOME) is on a local disk. It is used to represent the MW_HOME on local disk.
- /shareddisk: Root of the directory tree when the FMW install (either MW_HOME or DOMAIN_HOME) is on a shared storage system that is mounted by any one node of a CFC configuration. It is used to represent the MW_HOME on shared disk.

The example values this guide uses and that Oracle recommends for consistency are:

- `ORACLE_BASE: /u01/app/oracle`
- `MW_HOME (Apptier): ORACLE_BASE/product/fmw`
- `ORACLE_COMMON_HOME: MW_HOME/oracle_common`
- `WL_HOME: MW_HOME/wlserver_10.3`

The following table includes examples of Oracle home, domain home, and domain directory values used for some of the Oracle Fusion Middleware components:

Component	ORACLE_HOME	DOMAIN_HOME	Domain Directory
Identity Management	MW_HOME/idm	IDMDomain	MW_HOME/user_projects/domains/IDMDomain
Oracle SOA	MW_HOME/soa	SOADomain	MW_HOME/user_projects/domains/SOADomain
WebCenter	MW_HOME/wc	WCDomain	MW_HOME/user_projects/domains/WCDomain
WebCenter Content	MW_HOME/wcc	WCCDomain	MW_HOME/user_projects/domains/WCCDomain

Component	ORACLE_HOME	DOMAIN_HOME	Domain Directory
Oracle Portal	MW_HOME/portal	PortalDomain	MW_HOME/user_projects/domains/PortalDomain
Oracle Forms	MW_HOME/forms	FormsDomain	MW_HOME/user_projects/domains/FormsDomain
Oracle Reports	MW_HOME/reports	ReportsDomain	MW_HOME/user_projects/domains/ReportsDomain
Oracle Discoverer	MW_HOME/disco	DiscoDomain	MW_HOME/user_projects/domains/DiscoDomain
Web Tier	MW_HOME/web	Not applicable	Not applicable
Directory Tier	MW_HOME/idm	Not applicable	Not applicable

Example location for Applications Directory: ORACLE_BASE/admin/*domain_name*/apps

Example location for Oracle Instance: ORACLE_BASE/admin/*instance_name*

Oracle recommends ORACLE_BASE as the shared storage mount point. In most cases, this location helps to ensure that all the persistent bits of a failover unit are on the same shared storage. When more than one Cold Failover Cluster exists on a node, and each one fails over independently, different mount points will exist for each failover unit.

15.2.3 Transforming Oracle Fusion Middleware Infrastructure Components

An Oracle Fusion Middleware deployment comprises basic infrastructure components that are common across all the product sets. This section describes Cold Failover Cluster transformation steps for these components.

There are two Administration Server topologies supported for Cold Failover Cluster configuration. The following sections describe these two topologies and provide installation and configuration steps to prepare the Administration Server for Cold Failover Cluster transformation.

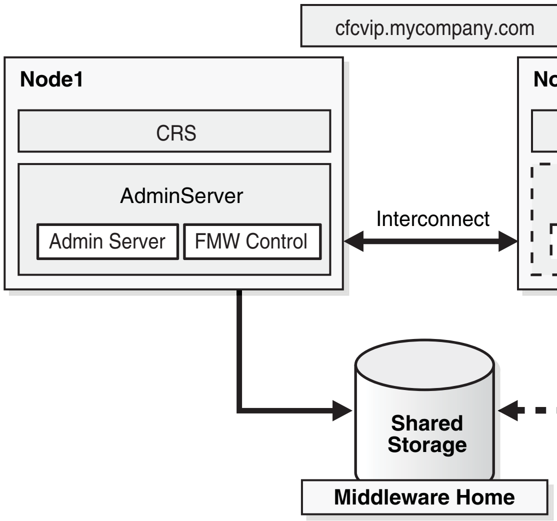
This section includes the following topics:

- Section 15.2.3.1, "Administration Server Topology 1"
- Section 15.2.3.2, "Topology 1 Installation Procedure"
- Section 15.2.3.3, "Administration Server Topology 2"
- Section 15.2.3.4, "Topology 2 Installation Procedure"
- Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster"
- Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers"
- Section 15.2.3.7, "Transforming Node Manager"
- Section 15.2.3.8, "Transforming Oracle Process Management and Notification Server"
- Section 15.2.3.9, "Transforming Oracle Enterprise Manager for an Oracle Instance"
- Section 15.2.3.10, "Transforming Web Tier Components and Clients"

15.2.3.1 Administration Server Topology 1

Figure 15–1 shows the first supported topology for Oracle Cold Failover Cluster.

Figure 15–1 Administration Server Cold Failover Cluster Topology 1



In Figure 15–1, the Administration Server runs on a two-node hardware cluster: Node 1 and Node 2. The Administration Server listens on the Virtual IP or hostname. The Middleware Home and the domain directory are on a shared disk that is mounted on either Node 1 or Node 2 at any given point. Both the Middleware home and the domain directory should be on the same shared disk or shared disks that can fail over together. If an enterprise has multiple Fusion Middleware domains for multiple applications or environments, this topology is well suited for Administration Server high availability. You can deploy a single hardware cluster to host these multiple Administration Servers. Each Administration Server can use its own virtual IP and set of shared disks to provide high availability of domain services.

15.2.3.2 Topology 1 Installation Procedure

To install and configure Cold Failover Cluster for the managed server in this topology:

Install the Middleware Home

This installation includes the Oracle home, WebLogic home, and the Domain home on a shared disk. This disk should be mountable by all the nodes that act as the failover destination for the Administration Server. Depending on the storage sub-system used, the shared disk may be mountable only on one node at a time. This is the preferred configuration, even when the storage sub system enables simultaneous mounts on more than one node. This is done as a regular single-instance installation. See the component chapters to install the Administration Server (and Enterprise Manager) alone. The overall procedure for each suite is as follows:

For Administration Server only:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Invoke the Configuration Wizard and create a domain with just the Administration Server.

   In the Select Domain Source screen, select the following:

- Generate a domain configured automatically to support the following products.
- Select Enterprise Manager and Oracle JRF.

For Oracle SOA, Oracle WebCenter Portal or Oracle WebCenter Content:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Install the Oracle Home for Oracle SOA, Oracle WebCenter Portal or Oracle WebCenter Content.

   See the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*, the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*, or the *Oracle WebCenter Content Installation Guide*.

For Oracle Identity Management:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Using Oracle Identity Management 11g Installer, install and configure the IDM Domain using the create domain option. In the Configure Components Screen, de-select everything except **Enterprise Manager** (selected by default)

   See the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*.

For Oracle Portal, Forms, Reports and Discoverer:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Using Oracle Fusion Middleware 11g Portal, Forms, Reports, and Discoverer Installer, install and configure the Classic Domain using the create domain option. In the Configure Components Screen, ensure that **Enterprise Manager** is selected.

> **Note:** In this case, at least one more Managed Server for the product components is also installed in this process; you cannot install the Administration Server on its own. You must transform this Managed Server to CFC using the specific procedure for the component. It is part of the same failover unit as the Administration Server.

For Oracle Business Intelligence:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Using Oracle Fusion Middleware 11g Business Intelligence Installer, install and configure the Business Intelligence domain using the create new BI system option.

> **Note:** In this case, at least one more Managed Server for the product components is also installed in this process; you cannot install the Administration Server on its own. You must transform this Managed Server to CFC using the specific procedure for the component. It is part of the same failover unit as the Administration Server.

Configuring the Administration Server for Cold Failover Cluster

To configure the Administration Server for Cold Failover Cluster:

1. Provision the Virtual IP using the following commands as root user:

   ```
   /sbin/ifconfig interface:index IP_Address netmask netmask
   /sbin/arping -q -U -c 3 -I interface IP_Address
   ```

   Where *IP_Address* is the virtual IP_address and the *netmask* is the associated netmask. In the following example, the IP_address is being enabled on the interface eth0.

   ```
   /sbin/ifconfig eth0:1 130.35.46.17 netmask 255.255.224.0
   /sbin/arping -q -U -c 3 -I eth0 130.35.46.17
   ```

2. Transform the Administration Server instance to Cold Failover Cluster following the procedure in section Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster."

3. Validate the Administration Server transformation by accessing the consoles on the virtual IP.This virtual IP address is the new IP address; Oracle recommends that you use only this virtual IP address.

   http://cfcvip.example.com:7001/console

   http://cfcvip.example.com:7001/em

4. Failover the Administration Server manually to the second node using the following procedure:

   > **Note:** If the Administration Server is managed by a Node Manager, you must enable the Node Manager for Cold Failover Cluster.

   a. Stop the Administration Server process (and any other process running out of a given Middleware Home)

   b. Unmount the shared storage from Node1 where the Middleware Home and domain directory exists.

   c. Mount the shared storage on Node2, follow storage specific commands.

   d. Disable the Virtual IP on Node1 using the following command as root user:

      ```
      /sbin/ifconfig interface:index down
      ```

      In the example below, the virtual IP_Address is being disabled on the interface eth0.

      ```
      /sbin/ifconfig eth0:1 down
      ```

   e. Enable the virtual IP on Node2 using the same commands as in Step 1.

   f. Start the Administration Server process.

      ```
      DOMAIN_HOME/bin/startWebLogic.sh
      ```

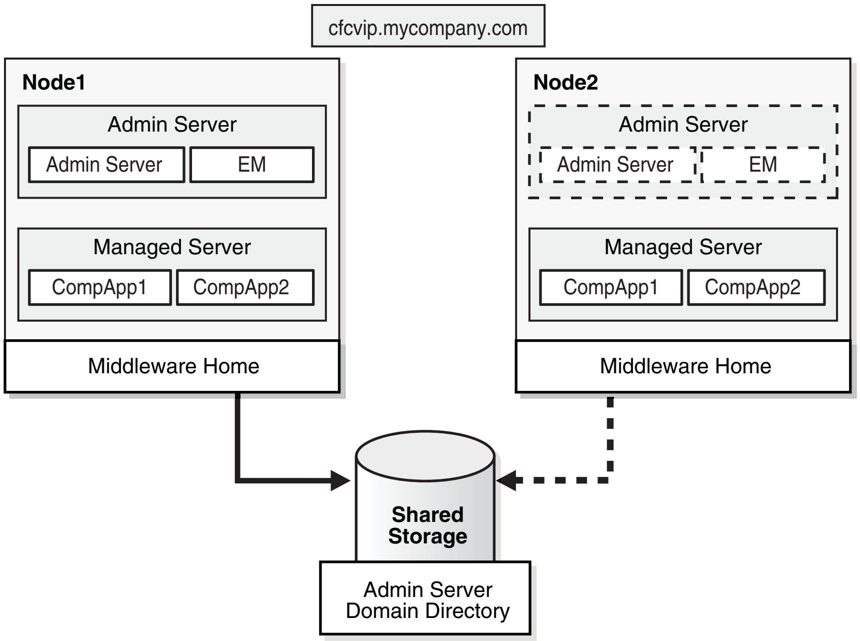
      Where *DOMAIN_HOME* is the location of your domain directory.

   g. Validate access to both the Administration Server and Enterprise Manager console.

15.2.3.3 Administration Server Topology 2

Figure 15–2 shows the second supported Administration Server topology for Oracle Cold Failover Cluster.

Figure 15–2 Administration Server Cold Failover Cluster Topology 2



In Figure 15–2, the Administration Server runs on a two-node hardware cluster: Node 1 and Node 2. The Administration Server is listening on the Virtual IP or hostname. The domain directory used by the Administration Server is on a shared disk. This is mandatory. This shared disk is mounted on Node 1 or Node 2 at any given point. The Middleware Homes, which contain the software, (WebLogic Home and the Oracle Home) are not necessarily on a shared disk. They can be on the local disk as well. The Administration Server uses the Middleware Home on Node1 for the software when it runs on Node1 and it uses the Middleware Home on Node2 when it runs on Node2. You must maintain the two Middleware Homes to be identical in terms of deployed products, Oracle Homes, and patches. In both cases, it uses the configuration available in the shared Domain Directory/Domain Home. Since this is shared, it ensures that the same configuration is used before and after failover.

This shared domain directory may also have other Managed Servers running. It may also be used exclusively for the Administration Server. If the domain directory is shared with other managed servers, appropriate consideration must be made for their failover when the Administration Server fails over. Some of these considerations are:

1. If the shared storage can be mounted as read/write on multiple nodes simultaneously, the Administration Server domain directory can be shared with other managed servers. In addition, it can fail over independently of the Managed Server. The Administration Server can failover and Managed Servers can continue to run independently on their designated nodes. This is possible because the Administration Server in this case requires only failover of the VIP, and does not require failover of the shared disk. The domain directory/domain home continues to remain available by the Managed Servers. Example of such storage include a NAS or a SAN/Direct attached storage with Cluster file system.

2. If only one node can mount the shared storage a time, sharing the Administration Server domain directory with a Managed Server implies that when the Administration Server fails over, the Managed Server that runs off the same domain directory must be shut down.

In this topology, you can use a hardware cluster to help automate failover (when used with properly configured clusterware). However, it is not required. A single hardware cluster can be deployed to host these multiple Administration Servers. Each Administration Server can use its own virtual IP and set of shared disks to provide domain services high availability.

This topology is supported for Oracle SOA Suite and Oracle WebCenter Portal Suite only.

> **Note:** For the Oracle Identity Management, an alternate topology is also supported for Cold Failover Cluster. See the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management* for more details.

15.2.3.4 Topology 2 Installation Procedure

To install and configure Cold Failover Cluster for the Administration Server in this topology:

Install the Middleware Home

Install the Middleware Home including the Oracle Home and WebLogic Home separately on the two nodes of the domain. The Administration Server domain directory is created on a shared disk. This disk should be mountable by all the nodes that act as the failover destination for the Administration Server. Depending on the storage sub-system used, the shared disk may be mountable only on one node at a time. This is a regular single-instance installation. Refer to the product suite for details on installing the Administration Server, and Enterprise Manager alone.

To install the Middleware Home for Oracle SOA Suite, Oracle WebCenter Portal, or Oracle WebCenter Content:

1. Install the Oracle WebLogic Server software on Node 1.
2. Install the Oracle Home for SOA or WebCenter on Node 1.
3. Repeat steps 1 and 2 on Node 2.
4. Start the Configuration Wizard on Node 1 and create a domain with just the Administration Server.

   In the Select Domain Source screen, select the following:

   - **Generate a domain configured automatically to support the following products**.
   - Select **Enterprise Manager** and **Oracle JRF**.
5. In the Specify Domain Name and Location screen, enter the domain name, and be sure the domain directory matches the directory and shared storage mount point.

Configuring the Middleware Home for Cold Failover Cluster

To configure the Middleware Home for Cold Failover Cluster:

1. Provision the Virtual IP. For example:

   ```
   /sbin/ifconfig eth0:1 IP_Address netmask netmask
   ```

```
/sbin/arping -q -U -c 3 -I eth0 IP_Address
```

Where *IP_Address* is the virtual IP_Address and the *netmask* is the associated netmask. In the following example, the IP_Address is being enabled on the interface eth0.

```
/sbin/ifconfig eth0:1 130.35.46.17 netmask 255.255.224.0
/sbin/arping -q -U -c 3 -I eth0 130.35.46.17
```

2. Transform the Administration Server instance to Cold Failover Cluster using the procedure in Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster."

3. Validate the Administration Server by accessing the consoles on the virtual IP.

   http://cfcvip.example.com:7001/console

   http://cfcvip.example.com:7001/em

4. Failover the Administration Server manually to the second node:

   a. Stop the Administration Server process and any other process running out of a given Middleware Home.

   b. Unmount the shared storage from Node1 where the Middleware Home or domain directory exists.

   c. Mount the shared storage on Node2, following storage specific commands.

   d. Disable the virtual IP on Node1:

   ```
   /sbin/ifconfig interface:index down
   ```

   In the following example, the IP_Address is being disabled on the interface `eth0`.

   ```
   /sbin/ifconfig eth0:1 down
   ```

   e. Enable the virtual IP on Node 2.

   f. Start the Administration Server process using the following command:

   ```
   DOMAIN_HOME/bin/startWebLogic.sh
   ```

   Where *DOMAIN_HOME* is the location of your domain directory.

   g. Validate access to both Administration Server and Enterprise Manager console.

15.2.3.5 Transforming the Administration Server for Cold Failover Cluster

To transform the Administration Server installed on a shared disk from Node 1, follow the steps in this section. These steps transform the container, therefore, both the Administration Console and Oracle Enterprise Manager Fusion Middleware Control, are transformed for Cold Failover Cluster. This results in other components such as OWSM-PM, deployed to this container, to become Cold Failover Cluster ready as well. The address for all of these services transforms `cfcvip.example.com`. After installation, to transform a non Cold Failover Cluster instance, to a Cold Failover Cluster:

1. Log into the Administration Console.

2. Create a machine for the virtual host

   a. Select **Environment**, and then **Machines**.

b. In the Change Center, click **Lock & Edit** then **New**.

c. In the Name field, enter **cfcvip.example.com**

d. For the **Machine OS** field, select the appropriate operating system.

e. Click **Next** then **Finish**.

> **Note:** Keep the **Listen Address field** set to *localhost*; the CFC solution relies on this setting. Do not change it to the virtual IP_Address or any other value.

f. On the Summary of Machines tab, click the machine name just created.

g. Click the **Servers** tab then click **Add**.

h. In the select server drop down list ensure AdminServer is selected.

i. Click **Finish**.

j. Click **Activate Changes**.

3. Configure the Administration Server to Listen on cfcvip.example.com.

   a. Select **Environment**, and then **Servers** from the Domain Structure menu.

   b. In the Change Center, click **Lock & Edit**.

   c. Click on the Administration Server (**AdminServer**)

   d. Change the Listen Address to **cfcvip.example.com**

   e. Click **Save**.

   f. Click **Activate Changes**.

   g. Restart the Administration Server.

> **Note:** Typically Administration Server transformation to Cold Failover Cluster takes place at domain creation time No other changes to other parts of the domain are expected. If this change happens after domain creation and other components are installed in the domain, follow the steps in the following Administration Server transformation section.

Changing Client Side Configuration for Administration Server

Any existing entities in the domain must communicate with the Administration Server using the new address. For example, when you start the Managed Servers manually, the Administration Server address should be specified as `cfcvip.example.com`.

In the `instance.properties` file, located in the *ORACLE_INSTANCE*/config/OPMN/opmn directory, make the following change:

```
adminHost=cfcvip.example.com
```

If the Oracle Instance is to be registered or reregistered with a Cold Failover Cluster Administration Server using the OPMN registration commands, the `AdminHost` location in the opmnctl command should reference the new location of the Administration Server (`cfcvip.example.com`).

Changing Client Side Configuration for Oracle Enterprise Manager

Since the Enterprise Manager is part of the same container where the Administration Server runs, transforming the Administration Server to Cold Failover Cluster also transforms the Enterprise Manager. If there are existing Enterprise Manager Agents configured to be part of the domain, these agent configurations must use the new location for the Enterprise Manager. To configure the new location for Enterprise Manager, use the following steps for each agent:

1. Set the directory to *ORACLE_INSTANCE*/EMAGENT/emagent_dir/sysman/config.
2. In the `emd.properties` file, change `node1.example.com` to `cfcvip.example.com` in the following attributes:
   - REPOSITORY_URL
   - emdWalletSrcUrl
3. Stop and restart the agent using the following commands:

```
cd ORACLE_INSTANCE/EMAGENT/emagent_dir/bin
./emctl stop agent
./emctl start agent
./emctl status agent
```

   This shows the Repository URL and it should now point to the new host.

15.2.3.6 Transforming Oracle WebLogic Managed Servers

All Oracle Fusion Middleware components are deployed to a Managed Server. An important step to convert an application or component that is deployed to Oracle WebLogic Server to Cold Failover Cluster is to change its listen address to the virtual IP being used. This change is done for the specific Managed Server to which the component has been deployed. You can make this change using the Administration Console or using WLST commands.

The following example describes the generic steps for Cold Failover Cluster transformation of a Managed Server named WLS_EXMPL. These steps apply to any Managed Server in the Fusion Middleware components.

This section includes the following topics:

- Section 15.2.3.6.1, "Transforming an Oracle WebLogic Managed Server using the Fusion Middleware Administration Console"
- Section 15.2.3.6.2, "Transforming an Oracle WebLogic Managed Server using the WLST Command Line"

15.2.3.6.1 Transforming an Oracle WebLogic Managed Server using the Fusion Middleware Administration Console In the following procedure, `cfcvip.example.com` is the virtual IP used for the Cold Failover Cluster and `WLS_EXMPL` is the managed server to be transformed.

1. Log into the Administration Console.
2. Create a machine for the virtual host:
   a. Select **Environment > Machines**.
   b. In the Change Center, click **Lock & Edit** then click **New**.

Note: Create a new machine only if you used a different VIP to transform managed servers. If so, go to step g.

c. For the **Name** field, enter **cfcvip.example.com**

d. For the **Machine OS** field, select the appropriate operating system.

e. Click **Next** and then click **Finish**.

   If you are transforming the Node Manager, you must complete the following steps:

f. Click the newly created machine.

g. Click **Node Manager** tab.

h. Update Listen Address: **cfcvip.example.com**.

i. Click **Save**.

j. Click **Activate Changes**.

k. Complete the steps in Section 15.2.3.7, "Transforming Node Manager."

3. Stop the WLS_EXMPL Managed server:

   a. Choose **Environment > Servers**.

   b. Click **Control**.

   c. Select **WLS_EXMPL**.

   d. Select **Force Shutdown Now** in the **Shutdown** drop-down menu.

4. Associate the WLS_EXMPL Managed Server with the VirtualHost Machine:

   a. Choose **Environment > Servers**.

   b. In the Change Center, click **Lock & Edit**.

   c. Click **Configuration**.

   d. Select **WLS_EXMPL**.

   e. For Machine, assign the newly created Machine by assigning it from the pull down menu.

   f. For **Listen Address**, enter **cfcvip.example.com**.

   g. Click **Save**.

   h. Click **Activate Changes**.

5. Start the WLS_EXMPL Managed Server.

Note: You can use several ways to start and stop a server instance. Use a method you have used previously to start and stop the server instances.

15.2.3.6.2 Transforming an Oracle WebLogic Managed Server using the WLST Command Line

You can transform an Oracle WebLogic managed server using WLST commands as well.

Oracle recommends shutting down the managed server you are transforming before performing these steps.

To transform a Managed Server using the WLST command line in online mode (with the WebLogic Server Administration Server up):

1. In the command line, enter:

```
WL_HOME/server/bin/setWLSEnv.sh
WL_HOME/common/bin/wlst.sh
```

2. In WLST, enter the following commands:

```
wls:/offline>connect(<username>,<password>,<AdminServer location>)
```

For example:

```
wls:/offline>connect('WebLogic', 'welcome1', 't3://cfcvip.example.com:7001')

wls:/DomainName/serverConfig> edit()
wls:/DomainName/edit> startEdit()
wls:/DomainName/edit !> create('cfcvip.example.com','Machine')
wls:/DomainName/edit !>
cd('Machines/cfcvip.example.com/NodeManager/cfcvip.example.com')
wls:/DomainName/edit !> set('ListenAddress', 'cfcvip.example.com')
wls:/DomainName/edit !>cd ('Servers')
wls:/DomainName/edit/Servers !>cd ('WLS_EXMPL')
wls:/DomainName/edit/Servers/WLS_EXMPL !>set('Machine',' cfcvip.example.com ')
wls:/DomainName/edit/Servers/WLS_EXMPL !>set('ListenAddress','
cfcvip.example.com ')
wls:/DomainName/edit/Servers/WLS_EXMPL !> save()
wls:/DomainName/edit/Servers/WLS_EXMPL !> activate()
wls:/DomainName/edit/Servers/WLS_EXMPL> exit()
```

3. Stop (if not already down) and start the Managed server.

After the Managed server transformation completes, verify that all references to it use the new Listen Address `cfcvip.example.com`. If Oracle HTTP Server serves as a front end to this Managed server, change any mod_wls_ohs configuration with mount points referring to applications in this Managed server to route to the new listening end point.

Note:

When transforming Oracle FMW SOA server with existing or deployed composites, there are a few locations where possible appearances of the old server's listen address may occur. When the server is front ended by OHS or an LBR, the references remain the same as they do not reflect the server's listen address.

- Deployed Composites may include specific endpoint references. For example they can use `callbackServerURL` specified as a binding property for the specific reference.These references must be updated to the new server's VIP.
- Composites relay on the SOA properties specified for the soa-infra application in Enterprise Manager: `ServerURL` and `CAllBackURL` must be updated if they were modified from the default null values.
- The system uses the front end address set at server level. So it must be updated to reflect the new VIP.

15.2.3.7 Transforming Node Manager

You can use Node Manager in a Cold Failover Cluster environment. A Node Manager that does not failover with the rest of the Cold Failover Cluster stack. In this case, Node Manager is not configured for Cold Failover Cluster and listens on all IPs on the machine, and not specifically on the virtual IP for Cold Failover Cluster. The failover nodes also have a similarly configured Node Manager already available and configured. The Node associated with the WebLogic instance communicates with the Node Manager on the localhost. For more details, see the *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*

For Cold Failover Cluster in general, port usage should be planned so that there are no port conflicts when failover occurs.

To convert the Node Manager to Cold Failover Cluster:

1. If Node Manager is running, stop it.

   The `nodemanager.properties` file is created only after the first start of Node Manager.

   Restart the Node Manager if necessary.

2. In the `nodemanager.properties` file located in the `WL_HOME`/common/nodemanager/ directory, set the `ListenAddress` to the virtual IP.

   For example:

   ```
   ListenAddress=cfcvip.example.com
   ```

3. Restart the Node Manager using the `startNodeManager.sh` file, located in the *WL_HOME*/server/bin directory.

> **Note:** For WebLogic Managed Servers and Administration Servers, hostname verification may be enabled or disabled in a given installation. For CFC installation where hostname verification is enabled and Node Manager is managing these instances, the hostname verification step should use certificates for the virtual IP cfcvip.example.com as part of these steps.

15.2.3.8 Transforming Oracle Process Management and Notification Server

Oracle Process Management and Notification Server (OPMN) is used for Process Management of system components and is part of the managed server instance.

Oracle recommends keeping the default OPMN configuration in a Cold Failover Cluster environment. No further steps are necessary for Cold Failover Cluster transformation of the OPMN process itself.

If you are transforming an Oracle Instance for Cold Failover Cluster and it is registered with an Administration Server, make the following changes in these files:

1. In the `topology.xml` file in the *DOMAIN_HOME*/opmn directory of the Administration Server domain, change hostname entries for this specific Oracle instance (being transformed to Cold Failover Cluster) to `cfcvip.example.com`.

   For example, for an Oracle HTTP Server instance transformed to Cold Failover Cluster, set the following in the `topology.xml` file

   ```
   <property name="HTTPMachine" value="cfcvip.example.com"/>
   ```

   For the instance itself:

   ```
   <ias-instance id="asinst " instance-home="/11gas3/MW/asinst"
   host="cfcvip.example.com" port="6701">
   ```

2. In the `instance.properties` file in the *ORACLE_INSTANCE*`/config/OPMN/opmn` directory, change `adminHost=<physical hostname>` to `adminHost=<cfcvip.example.com>`.
3. Restart all OPMN components.

15.2.3.9 Transforming Oracle Enterprise Manager for an Oracle Instance

When an Oracle instance transform to Cold Failover Cluster, you must also transform the Enterprise Manager agent that is part of this Oracle instance to Cold Failover Cluster. This topic describes how to transform the agent and the server.

To transform the Enterprise Manager agent:

1. Stop the Enterprise Manager agent using the following command:

```
cd ORACLE_INSTANCE/EMAGENT/emagent_dir/bin
./emctl stop agent
```

2. Set the directory to *ORACLE_INSTANCE*/EMAGENT/*emagent_dir*/sysman/config.
3. Replace the physical host name with the virtual host name using the managed bean and setting the operation on it, for example: `emoms.props:Location=`*AdminServer*`,name=emoms.properties,type=Properties,Application=em`

   To do this for each attribute:

   a. Log in to Enterprise Manager at `http://`*cfcvip.example.com*`:7001/em`

   > **Note:** This step assumes that you transformed the Administration Server listen address to listen on the virtual host name *cfcvip.example.com*. See Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster" for more information.

   b. Expand the WebLogic domain.

   c. Right click on the domain name and select **System MBean browser**.

   d. Select the search icon (binoculars). Enter `enoms.props:Location` to bring up the managed bean.

   e. Select the **Attributes** tab.

   f. Select **Properties**. The host name is in the Element listings. For example:

```
+Element_ElementNumber
 key example.sysman.emSDK.svlt.ConsoleServerName
value hostname.example.com:7001_Management_Service

 +Element_ElementNumber
 key example.sysman.emSDK.svlt.ConsoleServerHost
 value hostname.example.com
```

   g. Select the **Operations** tab then the **setProperty** link.

   h. Enter the key name and new value. Select **Invoke**.

4. In the `emd.properties` file, change `node1.example.com` to `cfcvip.example.com` for the `EMD_URL` attribute.

5. Change the `targets.xml` file on the agent side:

   ```
   cd ORACLE_INSTANCE/EMAGENT/emagent_dir/sysman/emd
   cp targets.xml targets.xml.org
   ```

   Modify `targets.xml` so that it has only targets related to the host and `oracle_emd`. Remove all other entries. For example:

   ```
   <Targets AGENT_TOKEN="ad4e5899e7341bfe8c36ac4459a4d569ddbf03bc">
          <Target TYPE="oracle_emd" NAME="cfcvip.example.com:port"/>
          <Target TYPE="host" NAME="cfcvip.example.com"  DISPLAY_
   NAME="cfcvip.example.com/>"
   </Targets>
   ```

6. Restart the agent

   ```
   cd ORACLE_INSTANCE/EMAGENT/emagent_dir/bin
   ./emctl start agent
   ```

To transform the Enterprise Manager server, make the following changes in the Administration Server domain directory:

Stop the Administration Server before making any changes.

1. Set your directory to *MW_HOME*/user_projects/domains/*domain_name*/sysman/state.

2. In the `targets.xml` file, located in *MW_HOME*/user_projects/domains/*domain_name*/sysman/state directory, modify the hostname from `node1.example.com` to `cfcvip.example.com`.

3. Restart the Administration Server.

15.2.3.10 Transforming Web Tier Components and Clients

The Web tier is made up of two primary components, Oracle HTTP Server and Oracle Web Cache. The next two sections describe how to transform Oracle HTTP Server and Oracle Web Cache for Cold Failover Cluster.

15.2.3.10.1 Transforming Oracle HTTP Server To transform Oracle HTTP Server for Cold Failover Cluster:

1. In *ORACLE_INSTANCE*/config/OHS/*component_name*/httpd.conf, change the following attributes:

   ```
   Listen cfcvip.example.com:port #OHS_LISTEN_PORT
   Listen cfcvip.example.com:port #OHS_PROXY_PORT
   ServerName cfcvip.example.com
   ```

2. In *ORACLE_INSTANCE*/config/OHS/*component_name*/admin.conf, change the following attributes:

   ```
   Listen cfcvip.example.com:port #OHS_LISTEN_PORT
   Listen cfcvip.example.com:port #OHS_ADMINISTRATOR_PORT
   ServerName cfcvip.example.com
   ```

3. Restart Oracle HTTP Server:

   ```
   .cd ORACLE_INSTANCE/bin
   ./opmnctl restartproc process-type=OHS
   ```

Also, perform a single sign-on registration, as described in Section 15.2.4.7, "Single Sign-On Reregistration (If required)."

Clients of Oracle HTTP Server

If an Oracle Web Cache instance is routing to Oracle HTTP Server that has been transformed to Cold Failover Cluster, in *ORACLE_INSTANCE*/config/WebCache/*component_name*/webcache.xml, change the following attributes:

Change node1.example.com to cfcvip.example.com, where node1.example.com is the previous address of the Oracle HTTP server before transformation.

```
<HOST ID="h1" NAME="cfcvip.example.com" PORT="8888" LOADLIMIT="100"
 OSSTATE="ON"/>
<HOST ID="h2" NAME="cfcvip.example.com" PORT="8890" LOADLIMIT="100" OSSTATE="ON"
 SSLENABLED="SSL"/>
```

15.2.3.10.2 Transforming Oracle Web Cache To transform an Oracle Web Cache for Cold Failover Cluster:

1. Set up an alias to the physical hostname on both nodes of the cluster in `/etc/hosts`.

   This is an alias to the IP_Address of the node. Set this in `/etc/hosts`. The alias name is `wcprfx.example.com`. For example, On node Node1, the `/etc/hosts` file entry would be **n.n.n.n node1 node1.example.com wcprfx wcprfx.example.com**

   On the failover node Node2, the `/etc/hosts` file, the entry would be **n.n.n.m node2 node2.example.com wcprfx wcprfx.example.com**.

2. In *ORACLE_INSTANCE*/config/WebCache/*component_name*/webcache.xml:

   - Change `node1.example.com` to `cfcvip.example.com` node1.example.com is where Oracle Web Cache was installed, and the host address it is listening on before transformation.

     ```
     SITE NAME="cfcvip.example.com"
     ```

   - Change the Virtual Host Name entries to be cfcvip.example.com for the SSL and non-SSL ports. For example:

     ```
     <HOST SSLENABLED="NONE" ISPROXY="NO" OSSTATE="ON" NUMRETRY="5"
     PINGINTERVAL="10" PINGURL="/" LOADLIMIT="100" PORT="8888"
     NAME="cfcvip.example.com" ID="h0"/>
     <HOST SSLENABLED="SSL" ISPROXY="NO" OSSTATE="ON" NUMRETRY="5"
     PINGINTERVAL="10" PINGURL="/" LOADLIMIT="100" PORT="8890"
     NAME="cfcvip.example.com" ID="h3"/>
     <VIRTUALHOSTMAP PORT="8094" NAME="cfcvip.example.com">
        <HOSTREF HOSTID="h3"/>
     </VIRTUALHOSTMAP>
     <VIRTUALHOSTMAP PORT="8090" NAME="cfcvip.example.com">
        <HOSTREF HOSTID="h0"/>
     </VIRTUALHOSTMAP>
     ```

   - Change cache name entries to be based of wcprfx.example.com where wcprfx.example.com is an alias created in /etc/hosts on all nodes of the cluster. For example:

     ```
     <CACHE WCDEBUGON="NO" CAPACITY="30" VOTES="1" INSTANCENAME="asinst_1"
     COMPONENTNAME="wc1" ORACLEINSTANCE="ORACLE_INSTANCE"
     HOSTNAME="wcprfx.example.com" ORACLEHOME="ORACLE_HOME"
     ```

```
NAME="wcprfx.example.com-WebCache">
```

- In the MULTIPORT section, change IPADDR from **ANY** to **cfcvip.example.com** for the following:

  ```
  PORTTYPE="NORM"
  SSLENABLED="SSL" PORTTYPE="NORM"
  PORTTYPE="ADMINISTRATION"
  PORTTYPE="INVALIDATION"
  PORTTYPE="STATISTICS"
  ```

  For example:

  ```
  <MULTIPORT>
     <LISTEN PORTTYPE="NORM" PORT="8090"
  IPADDR="cfcvip.example.com"/>
     <LISTEN SSLENABLED="SSL" PORTTYPE="NORM" PORT="8094"
  IPADDR="cfcvip.example.com">
         <WALLET>ORACLE_INSTANCE/config/WebCache/wc1/keystores/
  default</WALLET>
     </LISTEN>
     <LISTEN PORTTYPE="ADMINISTRATION" PORT="8091"
  IPADDR="cfcvip.example.com"/>
     <LISTEN PORTTYPE="INVALIDATION" PORT="8093" IPADDR="
  cfcvip.example.com"/>
     <LISTEN PORTTYPE="STATISTICS" PORT="8092"
  IPADDR="cfcvip.example.com"/>
  </MULTIPORT>
  ```

3. Restart Oracle Web Cache:

   ```
   cd ORACLE_INSTANCE/bin
   ./opmnctl restartproc process-type=WebCache
   ```

15.2.4 Transforming Oracle Fusion Middleware Components

This section describes the following Fusion Middleware components:

- Section 15.2.4.1, "Transforming Oracle Virtual Directory and Its Clients"
- Section 15.2.4.2, "Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager and Their Clients"
- Section 15.2.4.3, "Transforming Oracle Identity Federation and Its Client"
- Section 15.2.4.4, "Transforming Oracle Access Manager and Its Clients"
- Section 15.2.4.5, "Transforming Oracle Adaptive Access Manager and Its Clients"
- Section 15.2.4.6, "Transforming Oracle Identity Manager and Its Clients"
- Section 15.2.4.7, "Single Sign-On Reregistration (If required)"

For detailed explanation on the product components, see the appropriate component chapter in this guide.

15.2.4.1 Transforming Oracle Virtual Directory and Its Clients

This section describes how to transform Oracle Virtual Directory and its clients. It includes the following topics:

- Section 15.2.4.1.1, "Transforming Oracle Virtual Directory"
- Section 15.2.4.1.2, "Generating a New Key for the Keystore"

- Section 15.2.4.1.3, "Transforming Oracle Virtual Directory Clients"

15.2.4.1.1 Transforming Oracle Virtual Directory To transform an Oracle Virtual Directory server:

1. In a text editor, open the `listeners.os_xml` file in the *ORACLE_INSTANCE*/config/OVD/*componentname* directory.
2. Enter the following value to set the LDAP address to the virtual IP:

   ```
   <host>cfcvip.example.com</host>
   ```

3. Restart the Oracle Virtual Directory server using opmnctl.

   For example:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=ovd1
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ovd1
   ```

15.2.4.1.2 Generating a New Key for the Keystore Admin should generate a new key pair with the virtual hostname (self-signed or signed by CA) and associate this certificate in OVD listener configuration. This certificate should also be trusted by EMAgent for which it should be imported as a trusted cert in EMAgent's `cwallet.sso`.

For more information, see Section 8.3.5.2, "Generating a New Key for the Keystore Using WLST".

If you select a different keystore or change the certificate in the keystore for the Admin Gateway Listener or the LDAP SSL Endpoint Listener, you must import the certificate into the Oracle Enterprise Manager Fusion Middleware Control Agent's wallet. If you do not import the certificate, Oracle Enterprise Manager Fusion Middleware Control cannot connect to Oracle Virtual Directory to retrieve performance metrics.

To import the certificate into the Oracle Enterprise Manager Fusion Middleware Control Agent's wallet:

1. Export the Oracle Virtual Directory server certificate by executing the following command:

   ```
   ORACLE_HOME/jdk/jre/bin/keytool -exportcert \
   -keystore OVD_KEYSTORE_FILE -storepass PASSWORD \
   -alias OVD_SERVER_CERT_ALIAS -rfc \
   -file OVD_SERVER_CERT_FILE
   ```

2. Add the Oracle Virtual Directory server certificate to the Oracle Enterprise Manager Fusion Middleware Control Agent's Wallet by executing the following command:

   ```
   ORACLE_COMMON_HOME/bin/orapki wallet add -wallet \
   $ORACLE_INSTANCE/EMAGENT/EMAGENT/sysman/config/monwallet \
   -trusted_cert -cert OVD_SERVER_CERT_FILE -pwd WALLET_PASSWORD
   ```

15.2.4.1.3 Transforming Oracle Virtual Directory Clients All clients of Oracle Virtual Directory must use the virtual IP `cfcvip.example.com` to access Oracle Virtual Directory. For example, when using Oracle Directory Services Manager to administer a Cold Failover Cluster Oracle Virtual Directory instance, create a connection using `cfcvip.example.com` as the location of the Oracle Virtual Directory instance.

15.2.4.2 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager and Their Clients

This section describes how to transform Oracle Directory Integration Platform, Oracle Directory Services Manager, and their clients.

15.2.4.2.1 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager Oracle Directory Integration Platform and Oracle Directory Services Manager are deployed to a Managed Server. The procedure for CFC transformation is to configure the Managed Server to which they are deployed to listen on the `cfcvip.example.com` virtual IP. Follow the steps in Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers" to configure the WLS_ODS managed server to listen on the `cfcvip.example.com` virtual IP.

15.2.4.2.2 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager Clients Follow these steps to transform Oracle Directory Integration Platform and Oracle Directory Services Manager clients:

1. Clients of Oracle Directory Integration Platform and Oracle Directory Services Manager must use the virtual IP `cfcvip.example.com` to access these applications.
2. When Oracle HTTP Server is the front end for Oracle Directory Services Manager, the WebLogic configuration for Oracle Directory Services Manager must specify the virtual IP `cfcvip.example.com` as the address for the WLS_ODS Managed Server. To do this, change the WebLogic host configuration in the webserver proxy plugin configuration files for the mount points used by Oracle HTTP Server and Oracle Directory Services Manager. For example, use a text editor to make the following edits in the `mod_wl_ohs.conf` file:

```
#Oracle Directory Services Manager
<Location /odsm>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.example.com
    WebLogic port
</Location>
```

15.2.4.3 Transforming Oracle Identity Federation and Its Client

This section describes how to transform Oracle Identity Federation and its clients.

15.2.4.3.1 Transforming Oracle Identity Federation Oracle Identity Federation is a component that is deployed to a Managed Server. The procedure for Cold Failover Cluster transformation is to configure the Managed Server to which it is deployed to listen on the `cfcvip.example.com` virtual IP. Follow the steps in Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers" to configure the WLS_OIF Managed Server to listen on the `cfcvip.example.com` virtual IP. Since Oracle Identity Federation Cold Failover Cluster deployments are likely to be split into Service Provider and Identity Provider, more than one instance of WLS_OIF is likely to exist in a given deployment. Use the same Cold Failover Cluster procedure for both WLS_OIF instances.

After configuring the Managed Server to listen on the `cfcvip.example.com` virtual IP, log into the Oracle Enterprise Manager Fusion Middleware Control and perform these steps:

1. Go to **Farm > Identity and Access > OIF**.

2. In the right frame, go to **Oracle Identity Federation > Administration** and then make these changes:
   a. **Server Properties**: change the host to `cfcvip.example.com`
   b. **Identity Provider > Common**: change the **providerId** to `cfcvip.example.com`
   c. **Service Provider > Common**: change the **providerId** to `cfcvip.example.com`
   d. **Data Stores**: If LDAP is the data store, then replace the value of **Connection URL for User Data Store and Federation Data Store** with `cfcvip.example.com`
   e. **Authentication Engines > LDAP Directory**: Set the **ConnectionURL** to `cfcvip.example.com`
3. Restart the managed server so that the metadata generates.

15.2.4.3.2 Transforming Oracle Identity Federation Clients

Follow these steps to transform Oracle Identity Federation clients:

1. Clients of Oracle Identity Federation must use the virtual IP `cfcvip.example.com` to access these applications.
2. When Oracle HTTP Server is the front end for Oracle Identity Federation, the WebLogic configuration for Oracle Identity Federation must specify the virtual IP `cfcvip.example.com` as the address for the WLS_OIF Managed Server. To do this, change the WebLogic host configuration in the webserver proxy plugin configuration files for the mount points used by Oracle HTTP Server and Oracle Identity Federation. For example, use a text editor to make the following edits in the `oif.conf` file:

```
#Oracle Identity Federation
<Location /oif>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.example.com
    WebLogic port
</Location>
```

15.2.4.4 Transforming Oracle Access Manager and Its Clients

This section describes how to transform Oracle Access Manager to work in a Cold Failover Cluster environment.

As with other managed servers created using Configuration Wizard, a Cold Failover Cluster set up that requires the managed server to listen on a virtual IP can be done during the initial creation and by specifying the listen address of the managed server as the virtual hostname (`cfcvip.example.com`). In this case, the explicit transformation step is not needed.

15.2.4.4.1 Transforming Oracle Access Manager

Oracle Access Manager is deployed to a managed server (for example, `WLS_OAM1`) and the procedure for CFC transformation is to configure this managed server to listen on the virtual IP cfcvip.example.com. Follow the steps in Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers" to configure the WLS_OAM1 managed server to listen on the `cfcvip.example.com` virtual IP. All other requirements related to the placement of the Middleware Home and other related domain artifacts on a shared storage that can be failed over apply, as described in Section 15.2.1, "Cold Failover Cluster Requirements."

The Oracle Access Manager console is deployed as part of the Administration Server for the domain. For Cold Failover Cluster configuration of this console, the whole Administration Server needs to be configured in Active Passive. The Administration Server can share the same virtual IP `cfcvip.example.com`, or you can configure it to fail over independently using a separate virtual IP and shared disk. If they are using the same virtual IP, the Administration Server and the Oracle Access Manager managed server will fail over as a unit.

15.2.4.4.2 Transforming Oracle Access Manager Clients Follow these steps to transform Oracle Access Manager clients:

1. Clients of Oracle Access Manager must use the virtual IP cfcvip.example.com to access these applications. Any wiring done with other components such as Oracle Identity Manager and Oracle Adaptive Access Manager should also use the virtual IP cfcvip.example.com to access the applications.
2. When Oracle HTTP Server is the front end for Oracle Access Manager, the mod webLogic configuration for Oracle Access Manager must specify the virtual IP cfcvip.example.com as the address for the Oracle Access Manager managed server. To do this, change the WebLogic host configuration in the webserver proxy plugin configuration files for the mount points used. For example, use a text editor to make the following edits in the `mod_wl_ohs.conf` file:

```
#Oracle Access Manager
<Location /oam>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.example.com:port
</Location>
```

3. When the Oracle Access Manager console as part of the Oracle Administration Server is also configured to be Active Passive, and the Administration Server is configured to be front-ended by Oracle HTTP Server, you must change the WebLogic host configuration in the webserver proxy plugin configuration files. For example, use a text editor to make the following edits in the `mod_wl_ohs.conf` file:

```
#Oracle Access Manager Admin Console deployed to the Admin Server
<Location /oamconsole>
    SetHandler weblogic-handler
    WebLogicHost ADMINVHN
    WebLogicPort 7001
</Location>
```

15.2.4.5 Transforming Oracle Adaptive Access Manager and Its Clients

This section describes how to transform Oracle Adaptive Access Manager to work in a Cold Failover Cluster environment.

As with other managed servers created using Configuration Wizard, a Cold Failover Cluster set up that requires the managed server to listen on a virtual IP can be done during the initial creation as well by specifying the listen address of the managed server as the virtual hostname (cfcvip.example.com). In this case, the explicit transformation step is not needed.

15.2.4.5.1 Transforming Oracle Adaptive Access Manager Oracle Adaptive Access Manager is deployed to managed servers, both deployed to fail over as a single unit, and therefore sharing the same virtual IP and the same shared storage. The procedure to convert the OAAM Admin managed server and the OAAM Server managed server for

CFC transformation is to configure these managed servers to listen on the virtual IP cfcvip.example.com. Follow the steps in Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers" to configure these managed servers to listen on the cfcvip.example.com virtual IP. All other requirements related to the placement of the Middleware Home and other related domain artifacts on a shared storage that can be failed over apply, as described in Section 15.2.1, "Cold Failover Cluster Requirements."

15.2.4.5.2 Transforming Oracle Adaptive Access Manager Clients Follow these steps to transform Oracle Adaptive Access Manager clients:

1. Clients of Oracle Adaptive Access Manager must use the virtual IP cfcvip.example.com to access these applications. Any wiring done with other components such as Oracle Access Manager and Oracle Identity Manager should also use the virtual IP cfcvip.example.com to access the applications.
2. When Oracle HTTP Server is the front end for Oracle Adaptive Access Manager, the mod webLogic configuration for Oracle Adaptive Access Manager must specify the virtual IP cfcvip.example.com as the address for the Oracle Adaptive Access Manager managed servers. To do this, change the WebLogic host configuration in the webserver proxy plugin configuration files for the mount points used. For example, use a text editor to make the following edits in the `mod_wl_ohs.conf` file:

```
#Oracle Adaptive Access Manager
<Location /oaam_server>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.example.com WebLogicPort port
</Location>

#Oracle Adaptive Access Manager Admin Console
<Location /oaam_admin>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.example.com:port
</Location>
```

15.2.4.6 Transforming Oracle Identity Manager and Its Clients

This section describes how to transform Oracle Identity Manager to work in a Cold Failover Cluster environment.

As with other managed servers created using Configuration Wizard, a Cold Failover Cluster set up that requires the managed server to listen on a virtual IP can be done during the initial creation as well by specifying the listen address of the managed server as the virtual hostname (cfcvip.example.com). In this case, the explicit transformation step is not needed.

15.2.4.6.1 Transforming Oracle Identity Manager Oracle Identity Manager is deployed to managed servers. The typical CFC deployment will have the Oracle Identity Manager managed server and the another managed server with Oracle SOA and Oracle Web Services Manager deployed to fail over as a single unit and therefore sharing the same virtual IP and the same shared storage. The procedure to convert both these managed server for CFC transformation is to configure this managed servers to listen on the virtual IP cfcvip.example.com. Follow the steps in Section 15.2.3.6, "Transforming Oracle WebLogic Managed Servers" to configure these managed servers to listen on the cfcvip.example.com virtual IP. All other requirements related to the placement of the Middleware Home and other related domain artifacts on a shared storage that can be failed over apply, as described in Section 15.2.1, "Cold Failover Cluster Requirements."

15.2.4.6.2 Transforming Oracle Identity Manager Clients

Follow these steps to transform Oracle Identity Manager clients:

1. Clients of Oracle Identity Manager must use the virtual IP cfcvip.example.com to access these applications. Any wiring done with other components such as Oracle Access Manager and Oracle Adaptive Access Manager should also use the virtual IP cfcvip.example.com to access the applications. Since SOA is also configured to fail over with Oracle Identity Manager, the wiring from Oracle Identity Manager to SOA should also use the common virtual IP.
2. When Oracle HTTP Server is the front end for Oracle Identity Manager, the mod webLogic configuration for Oracle Identity Manager must specify the virtual IP cfcvip.example.com as the address for the managed servers. To do this, change the WebLogic host configuration in the webserver proxy plugin configuration files for the mount points used. For example, use a text editor to make the following edits in the `mod_wl_ohs.conf` file or `oim.conf` file:

   ```
   #Oracle Identity Manager
   <Location /oim>
       SetHandler weblogic-handler
       WebLogicHost cfcvip.example.com:port
       WebLogicPort port
   </Location>
   ```

   Refer to *Oracle Fusion Middleware System Administrator's Guide for Oracle Identity Manager* for more information about making configuration changes in Oracle Identity Manager.

15.2.4.7 Single Sign-On Reregistration (If required)

Single Sign-On (SSO) reregistration typically applies only to Oracle Portal, Forms, Reports, and Discoverer. After the front end listening endpoint on Oracle HTTP Server for this tier changes to the Virtual IP, it becomes necessary to do SSO reregistration so that the URL to be protected is configured with the virtual IP.

> **Note:** For 11g, you must create a WebGate using the new VIP instead of reregistering.

To reregister SSO, perform these steps on the 10.1.x installation of Identity Management where the SSO server resides

1. Set the *ORACLE_HOME* variable to the SSO ORACLE_HOME location.
2. Run *ORACLE_HOME*`/sso/bin/ssoreg.sh` with the following parameters:

   ```
   -site_name cfcvip.example.com:port
   -mod_osso_url http://cfcvip.example.com
   -config_mod_osso TRUE
   -oracle_home_path ORACLE_HOME
   -config_file /tmp/osso.conf
   -admin_info cn=orcladmin
   -virtualhost
   -remote_midtier
   ```

3. Copy `/tmp/osso.conf` file to the mid-tier home location:

   ORACLE_INSTANCE`/config/OHS/ohs1`
4. Restart Oracle HTTP Server by running the following command from the *ORACLE_INSTANCE*/bin directory:

```
./opmnctl restartproc process-type=OHS
```

5. Log into the SSO server through the following URL:

   ```
   http://sso.example.com/pls/orasso
   ```

6. In the **Administration** page and then **Administer Partner** applications, delete the entry for **node1.example.com**.

15.2.5 Additional Actions for Fusion Middleware Failover

In a Cold Failover Cluster environment, a failover node (node2.example.com) must be equivalent to the install machine (node1.example.com) in all respects. To make the failover node equivalent to the installation node, perform the following procedure on the failover instance:

For UNIX platforms follow these steps:

1. Failover the Middleware Home from Node 1 (the installation node) to the failover node (Node 2), following the mount/unmount procedure described previously.
2. As root, do the following:
   - Create an `oraInst.loc` file located in the `/etc` directory identical to the file on Node1.
   - Run the `root.sh` file located in the *ORACLE_HOME* directory on Node2, if it is required, and is available for the product suite.
3. Create the `oraInventory` on the second node, using the `attachHome` command located in the *ORACLE_HOME*`/oui/bin/attachHome.sh` directory.

15.2.6 Transforming an Oracle Database

In a typical Cold Failover Cluster deployment of Oracle Fusion Middleware, the database is also deployed as a cold failover cluster. This section describes how to transform a single instance Oracle database to a Cold Failover Cluster database. Perform this transformation before seeding the database using RCU and subsequent Fusion Middleware installations that use this seeded database.

To enable the database for Cold Failover Cluster:

1. Change the listener configuration that the database instance uses in the `listener.ora` file.

   Ensure that the HOST name in the listener configuration has the value of the virtual hostname. In addition, ensure that no other process (Oracle or third party) uses the listener port.

   ```
   <listener_name>  =
    (DESCRIPTION_LIST =
      (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST = <virtual_hostname>)(PORT = port))
      )
   )
   ```

   For example:

   ```
   LISTENER_CFCDB =
    (DESCRIPTION_LIST =
      (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST cfcdbhost.example.com)(PORT = 1521))
      )
    )
   ```

2. Change the `tnsnames.ora` file.

   Change an existing TNS service alias entry or create a new one:

   ```
   <tns_alias_name> =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = <virtual_hostname>)(PORT = port))
      )
      (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = <db_service_name>)
         (INSTANCE_NAME = <db_instance_name>)
      )
    )

   )
   ```

   For example:

   ```
   CFCDB =
    (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = cfcdbhost.example.com)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = cfcdb)
         (INSTANCE_NAME = cfcdb)
      )
    )
   ```

3. Change the local sp file to update the local_listener parameter of the instance.

   Log in as sysdba using SQL*Plus:

   ```
   SQL> alter system set local_listener='<tns_alias_name>' scope=both;
   )
   ```

   For example:

   ```
   SQL> alter system set local_listener='CFCDB' scope=both;
   ```

4. Shutdown and restart the listener.
5. Shutdown and restart the database instance.
6. Create the database service for the application server.

   Oracle recommends a dedicated service separate from the default database service. To create this service, execute the following SQL*Plus command:

   ```
   SQL> execute DBMS_SERVICE.CREATE_SERVICE
   ('<cfc_db_service_name>','<cfc_db_network_name>')
   ```

   For example:

   ```
   SQL> execute DBMS_SERVICE.CREATE_SERVICE
   ('cfcdb_asservice','cfcdb_asservice')
   ```

   To start the service, execute the following SQL*PLUS command:

   ```
   SQL> execute DBMS_SERVICE.START_SERVICE ('cfcdb_asservice')
   ```

You can set additional parameters for this service depending on the needs of the installation. See the *Oracle Database PL/SQL Packages and Types Reference* for details about the DBMS_SERVICE command.

15.2.6.1 Database Instance Considerations

Consider the following procedures for the Unix platform database instance:

1. Manually failover the Database Oracle Home from Node 1 (the installation node) to the failover node (Node 2) following the mount/unmount procedure described earlier.
2. As root, do the following:
   - Create an *`oraInst.loc`* file located in the `/etc` directory identical to the file on Node1.
   - Create an `oratab` file located in the `/etc` directory, identical to the file on Node1.
   - Run the `oracleRoot.sh` file located in the `ORACLE_HOME` directory on Node2, if it is required, and is available for the product suite.
3. Create the `oraInventory` file on the second node, using the `attachHome` command located in `ORACLE_HOME/oui/bin/attachHome.sh` directory.

15.3 Oracle Fusion Middleware Cold Failover Cluster Example Topologies

This section shows sample Cold Failover Cluster topologies. Since there are many possible combinations of topologies, these topologies are illustrative only. To achieve these topologies, more than one of the transformation steps apply. Refer to the steps mentioned earlier to configure the transformation.

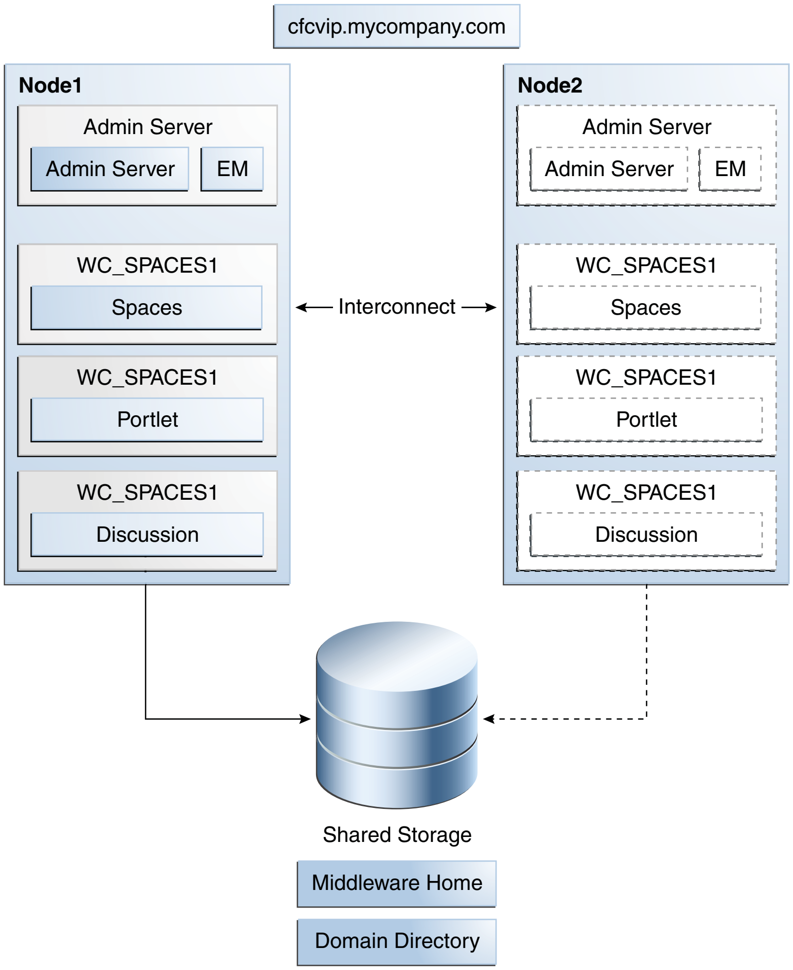
This section includes the following topics:

- Section 15.3.1, "Example Topology 1"
- Section 15.3.2, "Example Topology 2"
- Section 15.3.3, "Example Topology 3"

15.3.1 Example Topology 1

Figure 15–3 shows an Oracle WebCenter Portal Cold Failover Cluster deployment. Both the Administration Server and the WebCenter Managed Servers are in the domain and failover as unit. Therefore, they share the same virtual IP and are installed together on the same shared disk. There may be an Oracle HTTP Server front ending this topology. It is on a separate node in the example topology. It can also be on the same node, and can be part of the Cold Failover Cluster deployment. In this example, the database is also on a separate node. However, it is equally likely that the database is on the same cluster and is also Cold Failover Cluster-based (using its own virtual IP and shared disk).

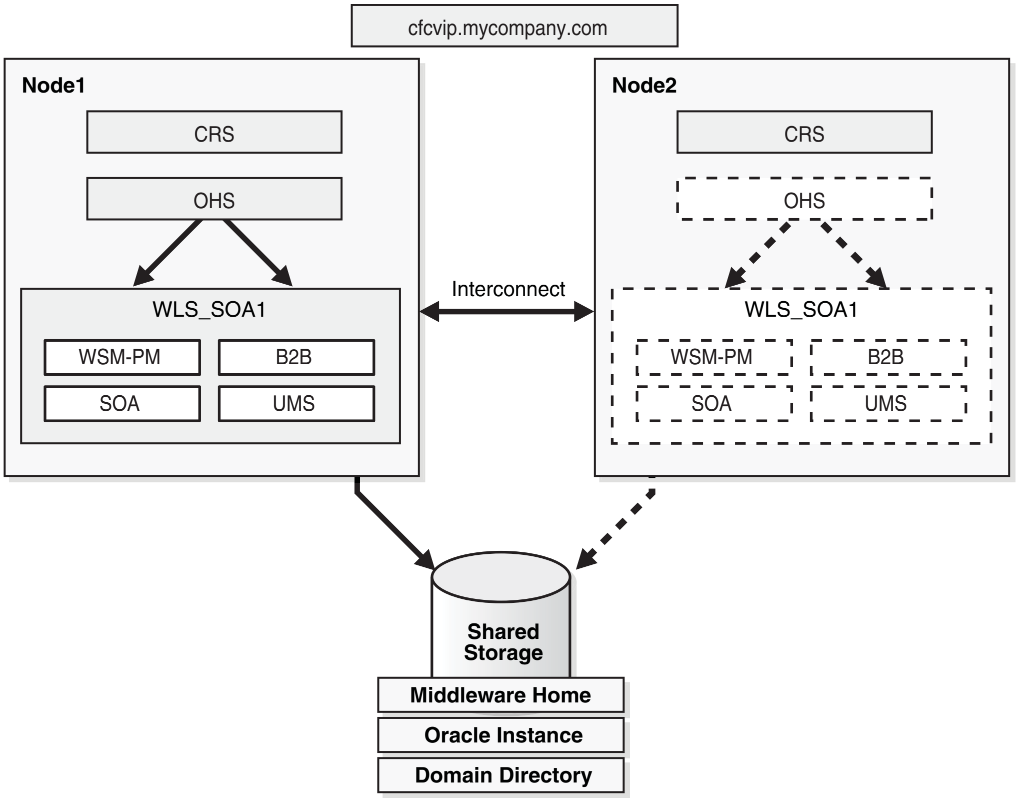
Figure 15–3 Cold Failover Cluster Example Topology 1



15.3.2 Example Topology 2

Figure 15–4 shows an example SOA Cold Failover Cluster deployment. In this example, only the SOA instance is deployed as Cold Failover Cluster, and the Administration Server is on a separate node. The database is also on a separate node in this example topology. Oracle HTTP Server in this case is part of the Cold Failover Cluster deployment, and part of the same failover unit as the SOA Managed Servers. Important variants of this topology include a Cold Failover Cluster Administration Server on the same hardware cluster. It may share the same virtual IP and shared disk as the SOA Managed Servers (SOA and Administration Server are part of the same failover unit) or use a separate virtual IP, and shared disk (Administration Server fails over independently). Similarly, depending on the machine capacity, the database instance can also reside on the same hardware cluster.

Figure 15–4 Cold Failover Cluster Example Topology 2

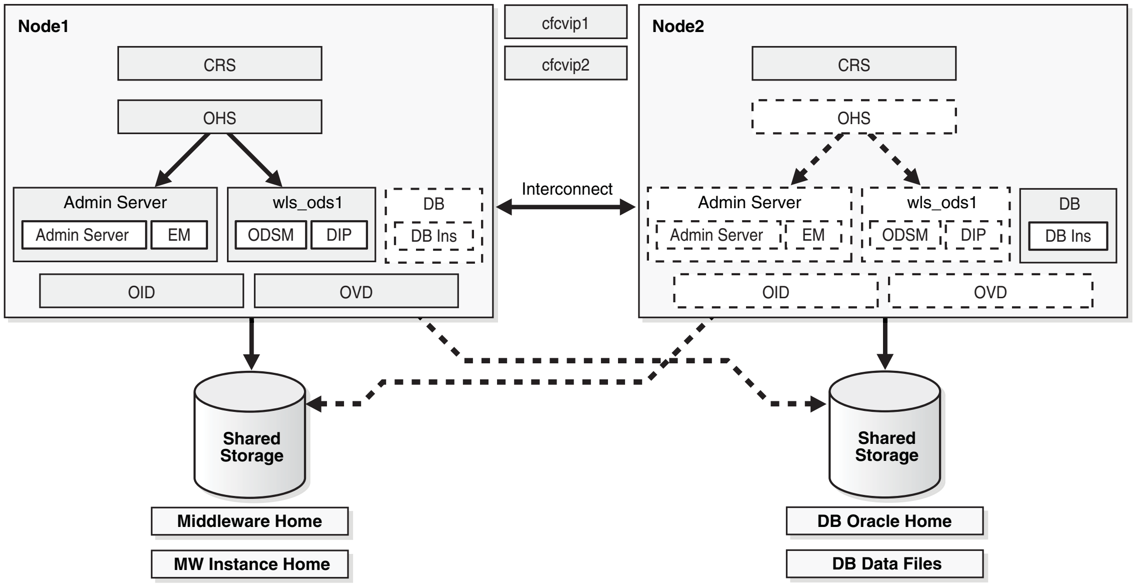


15.3.3 Example Topology 3

Figure 15–5 shows an Oracle Identity Management deployment. In this example topology, all components are on a two-node hardware cluster. Identity Management fails over as a unit, and both the Java EE (Administration Server and WLS_ods Managed Server) and system components are part of the same failover unit. They share the same virtual IP and shared disk (cfcvip1.example.com). The database is also on the same hardware cluster. It uses a different virtual IP, `cfcvip2.example.com`, and a different set of shared disks. During normal operations, the database runs on Node2 and the IDM stack runs on Node1. The other node acts as a backup for each.

This topology is recommended for most Cold Failover Cluster deployments. The example is for Identity Management, but this is true for the Oracle SOA, Oracle WebCenter Portal, and Oracle Portal, Forms, Reports, and Discoverer suites. In the recommended architecture, Oracle Fusion Middleware runs as one node of the hardware cluster. The Oracle database runs on the other node. Each node is a backup for the other. The Oracle Fusion Middleware instance and the database instance failover independently of each other, using different shared disks and different VIPs. This architecture also ensures that the hardware cluster resources are optimally used.

Figure 15–5 Cold Failover Cluster Example Topology 3



15.4 Transforming the Administration Server in an Existing Domain for Cold Failover Cluster

This section describes the steps for transforming an Administration Server in an existing domain to Cold Failover Cluster.

> **Note:** After Administration Server transformation, client side changes for Administration Server and Enterprise Manager, as mentioned in Section 15.2.3.5, "Transforming the Administration Server for Cold Failover Cluster," may be required.

Assumptions

The procedures in this section assume that:

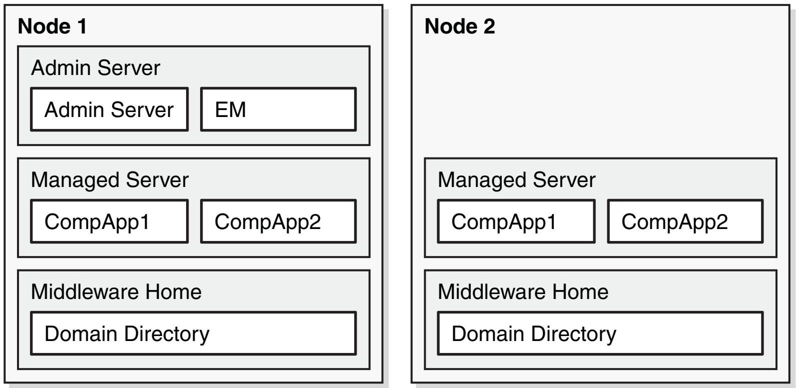
- All Fusion Middleware components are in a nostage deployment.
- The starting topology is an active-active cluster of the product suite (Node1 and Node2).
- The administration server is on Node1 to start.
- It shares the same domain home as the Managed Server on Node1.
- The *MW_HOME* path is the same on both Node1 and Node2.

These procedures also apply to customer applications deployed to Oracle WebLogic cluster installations, if the assumptions are met.

Start Topology

Figure 15–6 shows an example start topology before transforming the Administration Server.

Figure 15–6 Cold Failover Cluster Example Start Topology

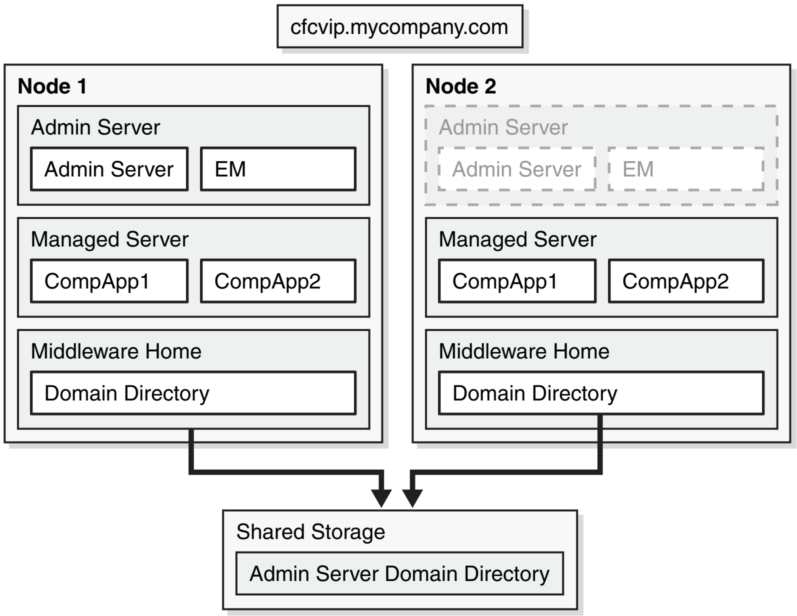


15.4.1 Destination Topologies

Figure 15–7 shows the possible destination topology after transforming the Administration Server for Cold Failover Cluster with the following characteristics:

- The Administration Server Domain Home is moved out onto a shared disk that Node1 and Node2 can mount, but is mounted by either one of the two at any given point in time.
- It continues to use the original Middleware Home available on Node1 and Node2.
- The Listen Address of the Administration Server is moved to a virtual IP.

Figure 15–7 Possible Destination Topologies



15.4.2 Cold Failover Cluster Transformation Procedure

To transform the Administration Server in an existing domain:

1. Shut down the cluster of the component Managed Servers and Administration Server.

2. Shut down the Node Manager process on each of the nodes, if it is running.
3. Back up the entire domain.
4. Provision the Virtual IP on Node1. For example:

   ```
   /sbin/ifconfig eth0:1 IP_Address netmask netmask
   /sbin/arping -q -U -c 3 -I eth0 IP_Address
   ```

   Where *IP_Address* is the virtual IP_Address and the *netmask* is the associated netmask.

   In the example below, the IP_Address is enabled on the interface `Local Area Connection`.

   ```
   /sbin/ifconfig eth0:1 130.35.46.17 netmask 255.255.224.0
   /sbin/arping -q -U -c 3 -I eth0 IP_Address
   ```

   Where *IP_Address* is the virtual IP_Address and the *netmask* is the associated netmask.

   In the example below, the IP_Address is enabled on the interface `Local Area Connection`.

   ```
   netsh interface ip add address "Local Area connection" IP_Address netmask
   ```

5. Start the Administration Server from the local Domain Home.

   ```
   cd DOMAIN_HOME/bin
   ./startWeblogic.sh
   ```

6. Transform the Administration Server instance to Cold Failover Cluster:

   Log into the Administration Console.

   Create a machine for the Virtual Host.

   a. Select **Environment**, and then **Machines**.
   b. In the Change Center, click **Lock & Edit**. Click **New**.
   c. In the **Name** field, enter **cfcvip.example.com**.

   > **Note:** Keep the **Listen Address field** set to *localhost*; the CFC solution relies on this setting. Do not change it to the virtual IP_Address or any other value.

   The Administration Server requires a new Virtual Host Machine so that it can interact with multiple node managers, not just the local one. Each node manager on each host listens on all interfaces, both the real IP and the local host (127.0.0.1). The Administration Server uses the new Virtual Host Machine definition exclusively.

   The Virtual Host Machine must point to *localhost* because *localhost* is the relative internal address for whatever machine is active; it sticks with the Administration Server. The Administration Server changes from one host to another but keeps the same Virtual Name and VIP. The node manager that is associated with the Administration Server also changes, because the Administration Server uses the *localhost* attribute in conjunction with the first host and then again, after failover, in conjunction with the second host. See Figure x

   d. Select the appropriate operating system and click **OK**.
   e. Select the machine you just created.
   f. Click the **Servers** tab then click **Add**.
   g. Select an existing server, and associate it with this machine.
   h. In the **Select Server** drop-down list, ensure **AdminServer** is selected.
   i. Click **Finish** then click **Activate Changes**.

   Configure the administration server to listen on cfcvip.example.com.

   a. Select **Environment**, and then **Servers** from the Domain Structure menu.
   b. In the Change Center, click **Lock & Edit**.
   c. Click on the Administration Server (**AdminServer**).
   d. Change the **Listen Address** to **cfcvip.example.com** Click **Save**.

   Stop the Administration Server from the Administration Console.

   a. Select **Environment**, and then **Servers** from the Domain Structure menu.
   b. Click **Control**.
   c. Select **Adminserver** by clicking on the checkbox next to it.
   d. Shut down Adminserver by selecting **Force Shutdown Now** under **Shutdown** pull-down menu.
   e. Click **Yes**.

   Ensure that the VIP is enabled on the system and start the Administration Server from the command line:

```
cd DOMAIN_HOME/bin
./startWeblogic.sh
```

7. Validate the Administration Server by accessing the consoles on the virtual IP:
   - http://cfcvip.example.com:7001/console
   - http://cfcvip.example.com:7001/em
8. Shut down the Administration Server on Node1 using the Administration Console.
9. Ensure that the shared disk is provisioned and mounted on Node1.
10. Pack the entire domain on Node1:

```
cd ORACLE_COMMON_HOME/common/bin
./pack.sh -managed=false -domain=/localdisk/user_projects/domains/domain_name
-template=cfcdomaintemplate_all.jar -template_name=cfc_domain_template_all
```

11. Unpack it on the shared disk:

```
./unpack.sh -domain=/shareddisk/user_projects/domains/domain_name
-template=cfcdomaintemplate_all.jar
-app_dir=/shareddisk/user_projects/apps -server_start_mode=prod
```

12. Product suite-specific changes:

    In the Oracle Identity Management Product Suite:

    - Back up the `config.xml` file, located in the `/shareddisk/user_projects/domains/`*domain_name*`/config/` directory in this example.

- Change WCC Socket Host Filter to VIP in the following files:
  - {domain}/ucm/ibr/config/config.cfg
  - {domain}/ucm/urm/config/config.cfg
  - {domain}/ucm/cs/config/config.cfg
- Edit the `config.xml` file, located in the `/shareddisk/user_projects/domains/`*domain_name*`/config` directory in this example, and make the following changes to source-path:

  For `dipapp`, change source path to `ORACLE_HOME/ldap/odi/dipapp/dipapps.ear`.

  For example:

  ```
  <app-deployment>
    <name>DIP#11.1.1.2.0</name>
    <target>cluster_ods</target>
    <module-type>ear</module-type>
    <source-path>ORACLE_HOME/ldap/odi/dipapp/dipapps.ear</source_path>
    <security-dd-model>DDOnly</security-dd-model>
    <staging-mode>nostage</staging-mode>
  </app-deployment>
  ```

  For the ODSM app, change the source path to `ORACLE_HOME/ldap/odsm/odsm.ear`

  For example:

  ```
  <app-deployment>
      <name>odsm#11.1.1.2.0</name>
      <target>cluster_ods</target>
      <module-type>ear</module-type>
      <source-path>ORACLE_HOME/ldap/odsm/odsm.ear</source-path>
      <security-dd-model>DDOnly</security-dd-model>
      <staging-mode>nostage</staging-mode>
    </app-deployment>
  ```

- For the OIF:

  Change the source path of `OIF-APP` to *ORACLE_HOME*`/fed/install/oif.ear`:

  ```
  <app-deployment>
      <name>OIF#11.1.1.2.0</name>
      <target>cluster_oif</target>
      <module-type>ear</module-type>
      <source-path>ORACLE_HOME/fed/install/oif.ear</source-path>
      <security-dd-model>Advanced</security-dd-model>
      <staging-mode>nostage</staging-mode>
    </app-deployment>
  ```

  Change the source path of oif-libs to `ORACLE_HOME/lib/java/shared/oracle.idm.oif/11.1.1.0.0/oif-libs.ear`:

  ```
  <library>
    <name>oif-libs#11.1.1.2.0@11.1.1.2.0</name>
      <target>cluster_oif</target>
      <module-type>ear</module-type>
      <source-path>ORACLE_
  HOME/lib/java/shared/oracle.idm.oif/11.1.1.0.0/oif-libs.ear</source-path>
  ```

```
    <security-dd-model>DDOnly</security-dd-model>
  </library>
```

13. Start the Administration Server:

    ```
    cd /shareddisk/user_projects/domains/domain_name/bin
    ./startWeblogic.sh
    ```

14. Validate the Administration Server by accessing the consoles on the virtual IP.

    - http://cfcvip.example.com:7001/console
    - http://cfcvip.example.com:7001/em

15. **Shut down** the Administration Server.

16. Perform the following on Node1:

    ```
    mv /localdisk/user_projects/domains/domain_name
    /localdisk/user_projects/domains/domain_name_old
    mv /localdisk/user_projects/applications/domain_name
    /localdisk/user_projects/applications/domain_name_old
    cd ORACLE_COMMON_HOME/common/bin
    ./pack.sh -managed=true -domain=/shareddisk/user_projects/domains/domain_name
    -template=cfcdomaintemplate_mngd.jar -template_name=cfc_domain_template_mngd
    ./unpack.sh -domain=/localdisk/user_projects/domains/domain_name
    -template=cfcdomaintemplate_mngd.jar
    ```

    > **Note:** These commands assume an applications directory exists under `user_projects`.

17. Copy the template to Node2:

    ```
    scp ORACLE_COMMON_HOME/common/bin/cfcdomaintemplate_mngd.jar
    Node2:ORACLE_COMMON_HOME/common/bin
    ```

18. Log in to Node2, and unpack on Node2

    ```
    mv /localdisk/user_projects/domains/domain_name
    /localdisk/user_projects/domains/domain_name_old
    mv /localdisk/user_projects/applications/domain_name
    /localdisk/user_projects/applications/domain_name_old
    cd ORACLE_COMMON_HOME/common/bin
    ./unpack.sh -domain=/localdisk/user_projects/domains/domain_name
    -template=cfcdomaintemplate_mngd.jar
    ```

    > **Note:** These commands assume an applications directory exists under `user_projects`.

19. For Oracle Identity Manager Suite, the following addition steps are required:

    For DIP:

    a. Locate the applications directory in the Oracle WebLogic Server domain directory on Node1:

       ```
       MW_HOME/user_projects/domains/IDMDomain/config/fmwconfig/servers/wls_
       ods1/applications
       ```

b. Copy the applications directory and its contents on Node1 to the same location in the domain directory on Node2.

```
scp -rp MW_HOME/user_projects/domains/IDMDomain/config/fmwconfig/servers
                /wls_ods1/applications
     user@IDMHOST2:MW_HOME/user_projects/domains/IDMDomain/config/fmwconfig
                /servers/wls_ods2/applications
```

For OIF:

a. Locate the applications directory in the Oracle WebLogic Server domain directory on Node1:

```
MW_HOME/user_projects/domains/OIFDomain/config/fmwconfig/servers/wls_
oif1/applications
```

b. Copy the applications directory and its contents on Node1 to the same location in the domain directory on Node2.

```
scp -rp MW_HOME/user_projects/domains/OIFDomain/config/fmwconfig/servers
                /wls_oif1/applications
     user@IDMHOST2:MW_HOME/user_projects/domains/OIFDomain/config/fmwconfig
                /servers/wls_oif2/applications
```

20. Start the Administration Server:

```
cd /shareddisk/user_projects/domains/domain_name/bin
./startWeblogic.sh
```

21. Start the node manager (if used) on Node1 and Node2:

```
cd WL_HOME/server/bin
./startNodeManager.sh
```

22. Start the component Managed Servers on Node1 and Node2 from the Administration Server Console (if Node Manager used) or from the command line.

23. Validate the deployment using component-specific functional tests.

24. Test Administration Server failover.

    Failover the Administration Server manually to the second node:

    a. Stop the Administration Server process (and any other process running out of a given Middleware Home).

    b. Unmount the shared storage from Node1 where the Middleware Home or domain directory exists.

    c. Mount the shared storage on Node2, following storage specific commands.

    d. Disable the virtual IP on Node1:

    ```
    ifconfig interface:index down
    ```

    In the following example, the IP_Address is disabled on the interface eth0:

    ```
    ifconfig eth0:1 down

    netsh interface ip delete address interface addr=IP_Address
    ```

    Where `IP_Address` is the virtual IP_Address.

In the following example, the IP_Address is enabled on the interface `Local Area Connection`.

```
netsh interface ip delete address 'Local Area connection' addr=130.35.46.17
```

e. Enable the virtual IP on Node2.

f. Start the Administration Server process using the following command:

```
DOMAIN_HOME/bin/startWebLogic.sh
```

Where *DOMAIN_HOME* is the location of your domain directory.

Validate access to both the Administration Server and Oracle Enterprise Manager Administration Console.

Validate with component-specific tests.

25. After validation, fail back the Administration Server to the node where it will normally run (this could be Node1 or Node2) and run normal operations.

Part III

Appendices

This part contains the following appendices:

- Appendix A, "Setting Up Auditing with an Oracle RAC Database Store"
- Appendix B, "Recommended Multi Data Sources"
- Appendix C, "Oracle Identity Management Workbook"
- Appendix D, "ascrsctl Online Help"
- Appendix E, "Configuring Distributed Notifications for MDS"

A

Setting Up Auditing with an Oracle RAC Database Store

With Oracle Fusion Middleware 11*g*, you have the option of setting up the Oracle Fusion Middleware Audit Framework service.

The Oracle Fusion Middleware Audit Framework provides a centralized audit framework for middleware products. The framework provides audit service for the following:

- Middleware Platform - This includes Java components such as Oracle Platform Security Services (OPSS) and Oracle Web Services, components that are leveraged by applications deployed in the middleware. Indirectly, all the deployed applications leveraging these Java components benefit from the audit framework auditing events that occur at the platform level.
- JavaEE applications - The objective is to provide a framework for JavaEE applications, starting with Oracle's own Java components. JavaEE applications will be able to create application-specific audit events. In the current release, the Java EE components using the Oracle Fusion Middleware Audit Framework are internal Oracle components.
- System components - For system components in the middleware that are managed by Oracle Process Manager and Notification Server (OPMN), the audit framework also provides an end-to-end service similar to that for Java components.

See the "Introduction to Oracle Fusion Middleware Audit Framework" chapter in the *Oracle Fusion Middleware Application Security Guide* for more introductory information about Oracle Fusion Middleware Audit Framework.

Out of the box, the Audit Framework uses the file system to store audit records. In a production environment, however, Oracle recommends that you use a database audit store to provide scalability and high availability for the audit framework. In high availability configurations such as the configurations described in this chapter, Oracle recommends that you use an Oracle Real Application Clusters (Oracle RAC) database as the database audit store.

The "Configuring and Managing Auditing" chapter in the *Oracle Fusion Middleware Application Security Guide* includes the steps for configuring auditing. The "Managing the Audit Store" section in that chapter includes steps for setting up a database as the audit data store.

When you set up the Oracle Fusion Middleware Audit Framework with an Oracle RAC database audit store, you must manually configure the following:

- Data sources and multi data sources for the audit data source using WebLogic Server

- The JDBC string for the OPMN loader in the opmn.xml file

The following sections provide additional information specific to configuring auditing when an Oracle RAC database is used as the audit data store.

A.1 Using WebLogic Server to Configure Audit Data Sources and Multi Data Sources

To set up the audit data source and multi data sources for an Oracle RAC database, follow the instructions in the "Managing the Audit Store" section of the *Oracle Fusion Middleware Application Security Guide*. Use the information in the "Set Up Audit Data Sources" section to set up the audit data sources and the information in the "Multiple Data Sources" section to configure an Oracle RAC database as the audit data store.

Use the information in the "Set Up Audit Data Sources" section to set up the audit data sources. To use an Oracle RAC database as the audit data store, you must create two individual data sources pointing to each individual Oracle RAC instance where the audit schemas are installed. The following settings are required:

- The connection URL should be in the following format:

  ```
  jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=PROTOCOL=TCP)(HOST=host-vip)
  (PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1))
  ```

  Note that the service name and instance name are required, in addition to the host and port.

- The driver used is `oracle.jdbc.OracleDriver`.
- The following property should be set:

  ```
  <property>
  <name>oracle.net.CONNECT_TIMEOUT</name>
  <value>10000</value>
  </property>
  ```

- The following settings are required for the individual data sources:
  - initial-capacity: **0**
  - connection-creation-retry-frequency-seconds: **10**
  - test-frequency-seconds: **300**
  - test-connections-on-reserve: **true**
  - test-table-name: **SQL SELECT 1 FROM DUAL**
  - seconds-to-trust-an-idle-pool-connection: **0**
  - global-transactions-protocol: **None**

Use the information in the "Multiple Data Sources" section to configure an Oracle RAC database as the audit data store. Create a multi data source with JNDI name jdbc/AuditDB. This multi data source should point to the individual data sources you created.

The following settings are required for the multi data source:

- test-frequency-seconds: **5**
- algorithm-type: **Load-Balancing**

- data-source-list: point to a list of comma separated child data sources **("JDBC Data Source-0,JDBC Data Source-1")**. This list is the same set of data sources that you created for each individual node of the Oracle RAC database.

A.2 Configuring the JDBC String for the Audit Loader

If you have an audit store configured, Oracle Process Manager and Notification Server (OPMN) manages several system components running in WebLogic Server. For these components, OPMN pushes the audit events to the database audit store.

The "Configure a Database Audit Store for System Components" section in the *Oracle Fusion Middleware Application Security Guide* describes how to set up the OPMN startup audit loader.

During the setup of the OPMN startup audit loader, you must modify the `rmd-definitions` element in the `opmn.xml` file. By default, the `rmd-definitions` element includes a JDBC string for a single instance database in this format:

```
jdbc:oracle:thin:@host:port:sid
```

When you are using an Oracle RAC database as the audit data store, you must use a JDBC string for an Oracle RAC database in the following format in the `rmd-definitions` element:

```
jdbc:oracle:thin@(DESCRIPTION=(ADDRESS_LIST=(LOAD_BALANCE=on)(ADDRESS=(PROTOCOL=
tcp)(HOST=node1-vip)(PORT=1521))(ADDRESS=(PROTOCOL=tcp)(HOST=node2-vip)(PORT=1521)
))(CONNECT_DATA=SERVICE_NAME=service-name.example.com)))
```

If you also need to configure the Oracle RAC database audit store for Java components, refer to the instructions in the "Configure a Database Audit Store for Java Components" section in the *Oracle Fusion Middleware Application Security Guide*.

B

Recommended Multi Data Sources

The following are examples of multi data source configuration files. Multi pool DS JDBC Multi Data Source-0 is made up of two data sources, JDBC Data Source-0, and JDBC Data Source-1. The property settings in bold text are recommended settings:

B.1 JDBC Multi Data Source-0

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Multi Data Source-0</name>
  <jdbc-connection-pool-params>
    <test-frequency-seconds>5</test-frequency-seconds>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/OracleDS</jndi-name>
    <algorithm-type>Load-Balancing</algorithm-type>
    <data-source-list>JDBC Data Source-0,JDBC Data Source-1</data-source-list>
    <failover-request-if-busy>false</failover-request-if-busy>
  </jdbc-data-source-params>
</jdbc-data-source>
```

B.2 JDBC Data Source-0 (non-XA)

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Data Source-0</name>
  <jdbc-driver-params>
    <url>jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))</url>
    <driver-name>oracle.jdbc.OracleDriver</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>username</value>
      </property>
      <property>
        <name>oracle.net.CONNECT_TIMEOUT</name>
        <value>10000</value>
      </property>
    </properties>
    <password-encrypted>{3DES}LMWCj6TOOuI=</password-encrypted>
  </jdbc-driver-params>
  <jdbc-connection-pool-params>
    <initial-capacity>0</initial-capacity>
    <max-capacity>20</max-capacity>
    <capacity-increment>1</capacity-increment>
    <shrink-frequency-seconds>900</shrink-frequency-seconds>
    <highest-num-waiters>2147483647</highest-num-waiters>

<connection-creation-retry-frequency-seconds>10</connection-creation-retry-frequen
cy-seconds>
    <connection-reserve-timeout-seconds>10</connection-reserve-timeout-seconds>
    <test-frequency-seconds>300</test-frequency-seconds>
    <test-connections-on-reserve>true</test-connections-on-reserve>
    <ignore-in-use-connections-enabled>true</ignore-in-use-connections-enabled>
    <inactive-connection-timeout-seconds>0</inactive-connection-timeout-seconds>
    <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
    <login-delay-seconds>0</login-delay-seconds>
    <statement-cache-size>10</statement-cache-size>
    <statement-cache-type>LRU</statement-cache-type>
    <remove-infected-connections>true</remove-infected-connections>

<seconds-to-trust-an-idle-pool-connection>0</seconds-to-trust-an-idle-pool-connect
ion>
    <statement-timeout>-1</statement-timeout>
    <pinned-to-thread>false</pinned-to-thread>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/OracleDS0</jndi-name>
    <global-transactions-protocol>None</global-transactions-protocol>
  </jdbc-data-source-params>
</jdbc-data-source>
```

B.3 JDBC Data Source-0 (XA)

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Data Source-0</name>
  <jdbc-driver-params>
    <url>
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))</url>
    <driver-name>oracle.jdbc.xa.client.OracleXADataSource</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>username</value>
      </property>
      <property>
        <name>oracle.net.CONNECT_TIMEOUT</name>
        <value>10000</value>
      </property>
    </properties>
    <password-encrypted>{3DES}LMWCj6TOOuI=</password-encrypted>
    <use-xa-data-source-interface>true</use-xa-data-source-interface>
  </jdbc-driver-params>
  <jdbc-connection-pool-params>
    <initial-capacity>0</initial-capacity>
<connection-creation-retry-frequency-seconds>10</connection-creation-retry-frequen
cy-seconds>
    <max-capacity>15</max-capacity>
    <capacity-increment>1</capacity-increment>
    <shrink-frequency-seconds>900</shrink-frequency-seconds>
    <highest-num-waiters>2147483647</highest-num-waiters>
    <connection-reserve-timeout-seconds>10</connection-reserve-timeout-seconds>
    <test-frequency-seconds>300</test-frequency-seconds>
    <test-connections-on-reserve>true</test-connections-on-reserve>
    <ignore-in-use-connections-enabled>true</ignore-in-use-connections-enabled>
    <inactive-connection-timeout-seconds>0</inactive-connection-timeout-seconds>
    <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
    <login-delay-seconds>0</login-delay-seconds>
    <statement-cache-size>10</statement-cache-size>
    <statement-cache-type>LRU</statement-cache-type>
    <remove-infected-connections>true</remove-infected-connections>
<seconds-to-trust-an-idle-pool-connection>0</seconds-to-trust-an-idle-pool-connect
ion>
    <statement-timeout>-1</statement-timeout>
    <jdbc-xa-debug-level>10</jdbc-xa-debug-level>
    <pinned-to-thread>false</pinned-to-thread>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/OracleDS1</jndi-name>
    <global-transactions-protocol>TwoPhaseCommit</global-transactions-protocol>
  </jdbc-data-source-params>
  <jdbc-xa-params>
    <keep-xa-conn-till-tx-complete>true</keep-xa-conn-till-tx-complete>
    <need-tx-ctx-on-close>false</need-tx-ctx-on-close>
    <xa-end-only-once>false</xa-end-only-once>
    <keep-logical-conn-open-on-release>false</keep-logical-conn-open-on-release>
    <resource-health-monitoring>true</resource-health-monitoring>
    <recover-only-once>false</recover-only-once>
    <xa-set-transaction-timeout>false</xa-set-transaction-timeout>
    <xa-transaction-timeout>0</xa-transaction-timeout>
    <rollback-local-tx-upon-conn-close>false</rollback-local-tx-upon-conn-close>
    <xa-retry-duration-seconds>300</xa-retry-duration-seconds>
    <xa-retry-interval-seconds>60</xa-retry-interval-seconds>
  </jdbc-xa-params>
</jdbc-data-source>
```

The only difference between JDBC Data Source-1 for XA and non-XA is they point to a different instance of Oracle RAC (host:port).

C

Oracle Identity Management Workbook

This appendix includes a workbook that you can use when performing a high availability configuration Oracle Identity Management components.

In this workbook, you can fill out the equivalent names that you plan to use in your environment when configuring high availability for the Oracle Identity Management components.

C.1 Workbook Tables for Oracle Identity Management

Use the following tables to record the names you plan to use in your Oracle Identity Management high availability configuration.

Enter the application URLS for your configuration in Table C–1.

Table C–1 Application URLs

Application	URL
Oracle WebLogic Administration Console	
Oracle Enterprise Manager Fusion Middleware Control	
Oracle Access Manager Console	

Enter the virtual IP address for your configuration in Table C–2.

Table C–2 Virtual IP Addresses

Purpose	IP Address	DNS Name
Oracle WebLogic Administration Server		

Enter the following generic file locations for your configuration in Table C–3.

Table C–3 Generic File Locations

Type	Location	Shared
ORACLE_BASE		No
MW_HOME		No

Enter the file locations for IDMHOST*n* for your configuration in Table C–4.

Table C–4 File Locations for IDMHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
Administration Server ORACLE_INSTANCE		No
WLS_ODS1 ORACLE_INSTANCE		No
WLS_ODS2 ORACLE_INSTANCE		No
WL_HOME		No
DOMAIN_HOME		No

Enter the file locations for OIDHOST*n* for your configuration in Table C–5.

Table C–5 File Locations for OIDHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OVDHOST*n* for your configuration in Table C–6.

Table C–6 File Locations for OVDHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OAMHOST*n* for your configuration in Table C–7.

Table C–7 File Locations for OAMHOSTn

Environment Artifact	Directory Location	Shared
IDM_ORACLE_HOME		No
IAM_ORACLE_HOME		No
SOA_ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OIMHOST*n* for your configuration in Table C–8.

Table C–8 File Locations for OIMHOSTn

Environment Artifact	Directory Location	Shared
IAM_ORACLE_HOME		No
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OAAMHOST*n* for your configuration in Table C–9.

Table C–9 File Locations for OAAMHOSTn

Environment Artifact	Directory Location	Shared
IDM_ORACLE_HOME		No
IAM_ORACLE_HOME		No
SOA_ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OIFHOST*n* for your configuration in Table C–10.

Table C–10 File Locations for OIFHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
Administration Server ORACLE_INSTANCE		No
WLS_OIF1 ORACLE_INSTANCE		No
WLS_OIF2 ORACLE_INSTANCE		No
WL_HOME		No
DOMAIN_HOME		No

Enter the file locations for the web tier for your configuration in Table C–11.

Table C–11 File Locations for the Web Tier

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter Oracle Identity Management details for your configuration in Table C–12.

Table C–12 Identity Management Artifacts

Identity Management Artifact	Value
Single Sign-On URL	
Oracle Internet Directory Host Name	
Oracle Internet Directory Non-SSL Port	
Oracle Internet Directory SSL Enabled	
Oracle Internet Directory SSL Port	
Oracle Internet Directory Security Realm	
Oracle Virtual Directory Host Name	

Table C–12 (Cont.) Identity Management Artifacts

Identity Management Artifact	Value
Oracle Virtual Directory Port	
Oracle Virtual Directory SSL Enabled	
Oracle Virtual Directory SSL Port	

Enter Authentication LDAP Artifacts details for the Oracle Identity Federation configuration in Table C–13.

Table C–13 Authentication LDAP Artifacts for Oracle Identity Federation

Authentication LDAP Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Credential ID Attribute	
User Unique ID Attribute	
Person Object Class	
Base DN	

Enter the Oracle Identity Federation Artifacts when using an LDAP User Data Store in your configuration in Table C–14.

Table C–14 LDAP User Data Store Artifacts for Oracle identity Federation

LDAP User Data Store Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Description Attribute	
User ID Attribute	
Person Object Class	
Base DN	

Enter the Oracle Identity Federation Artifacts when using an LDAP Federation Data Store in your configuration in Table C–15.

Table C–15 LDAP Federation Data Store Artifacts for Oracle Identity Federation

Federation Data Store Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Federation Record Context	
LDAP Container Object Class	

Enter the Oracle Identity Federation Artifacts when using an RDBMS User Data Store in your configuration in Table C–16.

Table C–16 RDBMS User Data Store Artifacts for Oracle Identity Federation

RDBMS User Data Store Artifact	Value
Hostname	
Username	
Password	
Login Table	
User ID Attribute	
User Description Attribute	

Enter the Oracle Identity Federation Artifacts when using an RDBMS Federation Data Store in your configuration in Table C–17.

Table C–17 RDBMS Federation Data Store Artifacts for Oracle Identity Federation

RDBMS Federation Data Store Artifacts	Value
Hostname	
Username	
Password	

Enter RDBMS Transient Data Store Artifacts for the Oracle Identity Federation configuration in Table C–18.

Table C–18 RDBMS Transient Data Store Artifacts for Oracle Identity Federation

Transient Data Store Artifact	Value
Hostname	
Username	
Password	

Enter database information for the metadata repository for your configuration in Table C–19.

Table C–19 Database Information for the Metadata Repository

Database Details	Value
Database Hosts (VIPs if using Oracle RAC)	
Listener Port	
Database Service Name	
RCU Prefix	
Main Schema Name/Password	
Auxiliary Schema Name/Password	
SYS Password	

Enter load balancer configuration information for your configuration in Table C–20.

Table C–20 Load Balancer Configuration

Purpose	Virtual Name	Port	Externally Visible	SSL	Destination Hosts	Destination Ports	SSL
Oracle Internet Directory							
Oracle Virtual Directory							
Administration Server							
Oracle Identity Manager							
Single Sign-On							

Enter port information for your configuration in Table C–21.

Table C–21 Port Information

Component	Host(s)	Port
Oracle Internet Directory		
Oracle Internet Directory (SSL)		
Oracle Virtual Directory		
Oracle Virtual Directory (SSL)		
WebLogic Server Console		
Oracle Enterprise Manager Fusion Middleware Control		
Oracle Directory Services Manager		
Oracle Access Manager Server		
Oracle Identity Manager Server		
Oracle Identity Manager		

Table C–21 (Cont.) Port Information

Component	Host(s)	Port
Oracle SOA		
Oracle Adaptive Access Manager Server		
Oracle Adaptive Access Manager Admin Server		
Oracle HTTP Server		
Oracle HTTP Server (SSL)		
Oracle HTTP Server Admin		
OPMN		
Node Manager		

D

ascrsctl Online Help

The detailed usage of ascrsctl can be obtained by following the instructions generated from the command `ascrsctl help`. This appendix provides the full content of the help pages to serve as a complete offline reference.

D.1 start

TOPIC

```
start - start an ASCRS resource
```

COMMAND

```
ascrsctl start -name <string> [-type <string>] [-node <string>]
```

DESCRIPTION

This ASCRS command is used to start a resource already created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name.

Optional arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-node

This argument specifies the cluster node for starting the resource. If it is not specified, the node will be chosen by CRS based on the placement policy for this resource.

EXAMPLE(S)

```
ascrsctl start -n mydisk -t disk
ascrsctl start -n ora.myvip.cfcvip -node hostA.example.com
```

D.2 stop

TOPIC

```
stop - stop an ASCRS resource
```

COMMAND

```
ascrsctl stop -name <string> [-type <string>] [-force] [-noprompt]
```

DESCRIPTION

This ASCRS command is used to stop a resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-force, -f

This argument shuts down the named resource and takes it offline from CRS management. This option guarantees to take offline the CRS monitoring of the resource, but does not guarantee to shut down the resource if it is already in an unmanageable state.

-noprompt, -np

When this argument is specified, the user is not prompted for confirmation, and the resource and its dependents are taken offline.

EXAMPLE(S)

```
ascrsctl stop -n mydisk -t disk
ascrsctl stop -n ora.myvip.cfcvip -f -np
```

D.3 status

TOPIC

```
status - check the status of ASCRS resources
```

COMMAND

```
ascrsctl status [-name <string>] [-type <string>] [-long]
```

DESCRIPTION

This ASCRS command is used to check the status of one or all resources created with ASCRS. The status of a resource includes its current running state, its basic CRS profile information, and its relationship to the other ASCRS resources.

-name, -n

This argument specifies the resource name. If not specified, check all resources.

-type, -t

This argument specifies the type of resources to be checked. It is not needed if the name is in canonical form.

-long, -l

When this argument is specified, the status information is appears in a detailed format.

EXAMPLE(S)

```
ascrsctl status
ascrsctl status -name ora.mydisk.cfcdisk
ascrsctl status -l
```

D.4 switch

TOPIC

```
switch - switchover an ASCRS resource to another cluster node
```

COMMAND

```
ascrsctl switch -name <string> [-type <string>]
        [-node <string>] [-noprompt]
```

DESCRIPTION

The ASCRS command switches over an ASCRS resource that is currently in online state to another node in the cluster. All resources this resource depends upon also switch over.

Mandatory arguments:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-node

This argument specifies the target cluster node of this resource. If it is not specified, the target node will be chosen by CRS based on the placement policy for this resource.

-noprompt, -np

When this argument is specified, the user is not prompted for confirmation.

EXAMPLE(S)

```
ascrsctl switch -n mydisk -t disk hostB.example.com
ascrsctl switch -n ora.myvip.cfcvip -np
```

D.5 delete

TOPIC

delete - delete an ASCRS resource

COMMAND

```
ascrsctl delete -name <string> [-type <string>] [-noprompt]
```

DESCRIPTION

This ASCRS command is used to delete a resource created with ASCRS. Once a resource is successfully deleted, the resource is no longer managed by CRS.

An ASCRS resource cannot be deleted if there are still one or more other resources depending on it, or if it is not in offline state.

Mandatory argument:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-noprompt, -np

When specified, the user is not prompted for confirmation.

EXAMPLE(S)

```
ascrsctl delete -n mydisk -np
ascrsctl delete -n ora.myvip.cfcvip
```

D.6 create/disk

TOPIC

create/disk - create **disk** ASCRS resource

COMMAND

```
ascrsctl create -name <string> -type disk -path <string>
          -mountCommand <string> -umountCommand <string>
          [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a shared disk resource in CRS. To successfully create a disk resource, a signature file needs to be created on the root of the shared disk. See the *Oracle Fusion Middleware Administrator's Guide* for details.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **disk**.

path

This argument specifies the mount point of the shared disk.

-mountCommand, -mc

This argument specifies a platform-specific command to be invoked when mounting the shared disk. Command "nop" takes no action. On Windows, if the shared disk is NTFS, use the command "diskmgr online <disk number>", where <disk number> can be identified in the Microsoft Disk Manager window.

-umountCommand, -umc

This argument specifies a platform-specific command to be invoked when unmounting the shared disk. Command "nop" takes no action. On Windows, if the shared disk is NTFS, use the command "diskmgr offline <disk number>", where <disk number> can be identified in the Microsoft Disk Manager window.

These are non-mandatory options:

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties of this resource, for example, as=1,rt=400.

The property names and their value ranges are shown in Table D–1, where, except for as, st, and ra, all the other numbers are in seconds. These properties can be configured through the ASCRS configuration file. If a property is neither configured nor specified with this option, the default is assumed.

Table D–1 Resource Values for Create Commands

Parameter	Min	Max	Default	Purpose
as	0	1	1	Auto Start
ci	5	6000	600	Check Interval
fd	5	600	50	Failover Delay
fi	5	6000	50	Failure Interval
ft	0	20	5	Failure Threshold
ra	0	20	3	Restart Attempts

Table D–1 (Cont.) Resource Values for Create Commands

Parameter	Min	Max	Default	Purpose
st	20	3600	30	Script Timeouts
rt	20	3600	30	Start Timeout
pt	20	3600	30	Stop Timeout

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ASCRS configuration.

The valid values are **normal** or **fast**. If it not specified, **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
UNIX:
ascrsctl create -n dbhome -t disk -path /cfcdb1
           -mc "/bin/mount /dev/sda1 /cfcdb1" -p fd=30
ascrsctl create -n dbhome -t disk -path /cfcdb1
           -mc "/bin/mount /dev/sda1 /cfcdb1"
           -umc "/bin/umount /dev/sda1"

Windows:
ascrsctl create -n asdisk -t disk -path c:\oracle\asdisk
                    -mc "diskmgr online 2" -mc "diskmgr offline 2"
                    -p fd=30
```

D.7 update/disk

TOPIC

```
update/disk - update disk ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type disk] [-path <string>]
          [-mountCommand <string>] [-umountCommand <string>]
          [options]
```

DESCRIPTION

This ASCRS command is used to update a **disk** resource created with ASCRS.

Mandatory argument:

-name, -n

This argument specifies the resource name to be updated. If it is a fully qualified name, the **-type** option can be omitted.

These are non-mandatory options:

-type, -t

This argument specifies the resource type. Its value must be **disk**.

-path

This argument specifies the mount point of the shared disk.

-mountCommand, -mc

This argument specifies a platform-specific fully qualified command or script name to be executed for mounting the shared disk. On Windows, if the shared disk is NTFS, use the command "diskmgr online <disk number>", where <disk number> can be identified in the Microsoft Disk Manager window.

-umountCommand, -umc

This argument specifies a platform-specific fully qualified command or script name to be executed for unmounting the shared disk. On Windows, if the shared disk is NTFS, use the command "diskmgr offline <disk number>", where <disk number> can be identified in the Microsoft Disk Manager window.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all nodes.

-resourceParams, -params, -p

This argument specifies space-separated or comma separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1, **rt**=400.

The property names and their value ranges are shown in Table D–2, where, except for as, st, and ra, all the other numbers are in seconds.

Table D–2 Resource Values for Update Commands

Parameter	Min	Max	Purpose
as	0	1	Auto Start
ci	5	6000	Check Interval
fd	5	600	Failover Delay
fi	5	6000	Failure Interval
ft	0	20	Failure Threshold
ra	0	20	Restart Attempts
st	20	3600	Script Timeout
rt	20	3600	Start Timeout
pt	20	3600	Stop Timeout

EXAMPLE(S)

```
UNIX:
ascrsctl update -n mydisk -t disk -umfc "/bin/umount -l /sharedisk"

Windows:
    ascrsctl update -n mydisk -t disk -mc "diskmgr online 1"
                                      -umc "diskmgr offline 1"
```

D.8 create/vip

TOPIC

```
create/vip - create vip ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type vip -ipAddr <ip> -netmask <string>
               -interface <string> [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a virtual IP resource in CRS

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **vip**.

-ipAddr, -ip

This argument specifies the IP address of the virtual IP or its hostname.

-netmask, -nm

This argument specifies the network mask for the above virtual IP.

-interface, -if

This argument specifies the network interface(s) on which the IP should be enabled.

On UNIX, the value can be one or more interface names such as eth0 or "eth0|eth1". On Windows, the value can be one or more network connection names, such as "Public network1|Public network2".

These are non-mandatory options:

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–1, where, except for **as**, **st** and **ra**, all the other numbers are in seconds. These properties can be configured through the ASCRS configuration file. If a property is neither configured nor specified with this option, the default is assumed.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ASCRS configuration file.

The valid values are **normal** or **fast**. If it not specified, **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
UNIX:
ascrsctl create -n myvip -t vip -ip 192.168.1.10 -nm 255.255.255.0
          -if eth1 -p ci=5

Windows:
ascrsctl create -n myvip -t vip -ip 192.168.1.10 -nm 255.255.255.0
                  -if "Public network" -p ci=5
```

D.9 update/vip

TOPIC

```
update/vip - update vip ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type vip] [-ipAddr <string>
         [-netmask <string>] [-interface <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update a vip resource created with ASCRS.

Mandatory argument:

-name, -n

This argument specifies the resource name to be updated. If it is a fully qualified name, the **-type** option can be omitted.

These are non-mandatory options:

-type, -t

This argument specifies the resource type. Its value must be **vip**.

-ipAddr, -ip

This argument specifies the IP address of the virtual IP or its hostname.

-netmask, -nm

This argument specifies the network mask for the above virtual IP.

-interface, -if

This argument specifies the network interface(s) on which the IP should be enabled.

On UNIX, the value can be one or more interface names such as eth0 or "eth0|eth1". On Windows, the value can be one or more network connection names, such as "Public network1|Public network2".

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. This value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all the nodes.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–2, where, except for **as**, **st** and **ra**, all the other numbers are in seconds.

EXAMPLE(S)

```
UNIX:
ascrsctl update -n ora.myvip.cfcvip -ip 192.168.1.10
ascrsctl update -n ora.myvip.cfcvip -if eth1 -p ci=3

Windows:
ascrsctl update -n ora.myvip.cfcvip -if Public -p ci=3
```

D.10 create/dblsnr

TOPIC

```
create/dblsnr - create dblsnr ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type dblsnr -listenerName <string>
          -listenerOracleHome <string>
          -vip <string> -disk <string>
          [-tnsAdmin <string>] [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) an Oracle database listener resource in CRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **dblsnr**.

-listenerName, -ln

This argument specifies the database listener name.

-listenerOracleHome, -lsnroh, -loh

This argument specifies the Oracle Home of the database that owns this listener.

-vip

This argument specifies the vip resource this listener runs on.

-disk

This argument specifies the disk resource the Oracle Home resides on.

The other non-mandatory options:

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. This value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all the nodes. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–1, where, except for **as**, **st** and **ra**, all the other numbers are in seconds. These properties can be configured through the ASCRS configuration file. If a property is neither configured nor specified with this option, the default is assumed.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ASCRS configuration file.

The valid values are **normal** or **fast**. If it not specified, **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

-tnsAdmin, -ta

This argument specifies the location of the listener configuration if it is not in the default location within the Oracle Home.

EXAMPLE(S)

```
UNIX:
ascrsctl create -name mydblsnr -type dblsnr -listenerName orcl
           -listenerOracleHome /cfcdb1
           -vip 192.168.1.10 -disk ohdisk

Windows:
ascrsctl create -name mydblsnr -type dblsnr -listenerName orcl
                -listenerOracleHome c:\oraasshare\cfcdb1
                -vip myvip -disk ohdisk
```

D.11 update/dblsnr

TOPIC

```
update/dblsnr - update dblsnr ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type dblsnr]
           [-listenerName <string>]
           [-listenerOracleHome <string>]
           [-vip <string>] [-disk <string>]
           [-tnsAdmin <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update a **dblsnr** resource created with ASCRS.

Mandatory argument:

-name, -n

This argument specifies the resource name to be updated. If it is a fully qualified name, the **-type** option can be omitted.

These are non-mandatory options:

-type, -t

This argument specifies the resource type. Its value must be **dblsnr**.

-listenerName, -ln

This argument specifies the database listener name.

-listenerOracleHome, -lsnroh, -loh

This argument specifies the Oracle Home of the database that owns this listener.

-vip

This argument specifies the vip resource this listener runs on.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all the nodes.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–2, where, except for **as**, **st** and **ra**, all the other numbers are in seconds.

-tnsAdmin, -ta

This argument specifies the new location of the listener configuration file. The special value "nil" sets this location to its default.

-disk

This argument specifies the disk resource the Oracle Home resides on.

EXAMPLE(S)

```
ascrsctl update -n mydblsnr -t dblsnr -vip newvip
ascrsctl update -n mydblsnr -t dblsnr -disk newdisk -p st=30,pt=40,rt=40
```

D.12 create/db

TOPIC

```
create/db - create db ASCRS resource
```

COMMAND

```
For database instance:
ascrsctl create -name <string> -type db -oraHome <string>
          -oraSID <string> -disk <string> [<string> ...]
          -lsnr <string> [-pfile <string>]
```

```
            [-componentID dbinstance] [options]

For database console:
ascrsctl create -name <string> -type db -oraHome <string>
            -oraSID <string> -disk <string> -vip <string>
            [-componentID dbconsole] [options]

For job scheduler (Windows only):
ascrsctl create -name <string> -type db -oraHome <string>
            -oraSID <string> -disk <string>
            [-componentID jobscheduler] [options]

For VSS writer (Windows only):
ascrsctl create -name <string> -type db -oraHome <string>
            -oraSID <string> -disk <string>
            [-componentID vsswriter] [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) an Oracle database resource in CRS. Depending on the specified component ID, the database resource represents either a core database instance, a dbconsole service, or, if the platform is Windows, an Oracle job scheduler service or an Oracle Volume Shadow Service (VSS).

These are mandatory arguments for all database resources:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **db**.

-oraHome, -oh

This argument specifies the database Oracle Home location.

-oraSID, -sid

This argument specifies the Oracle SID name.

-disk

This argument specifies the shared disks hosting the Oracle Home and data files. For a database instance component, the value for this parameter is a space-separated or comma-separated list of ASCRS disk resource names; for other components, its value is the disk resource hosting the Oracle home.

The other non-mandatory options:

-componentID, -c

This argument identifies the database component to be managed. It assumes **dbinstance**, **dbconsole**, **jobscheduler** (Windows only), or **vsswriter** (Windows only). If this is not specified, **dbinstance** is assumed.

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–1, where, except for **as**, **st** and **ra**, all the other numbers are in seconds. These properties can be configured through the ASCRS configuration file. If a property is neither configured nor specified with this option, the default is assumed.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ASCRS configuration file.

The valid values are **normal** or **fast**. If it not specified, **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

Component-specific options:

For database instance component:

-lsnr

This argument specifies the listener resource used by this database. It is mandatory.

-pfile, -pf

This argument specifies the database pfile for starting the database. It is only valid for the dbinstance component.

For database console component:

-vip

This argument specifies the virtual IP resource used by this database console resource. It is mandatory.

EXAMPLE(S)

```
UNIX:
ascrsctl create -n mydb -t db -oh /cfcdb1 -sid orcl
          -disk ohdisk datafiledisk -lsnr mydblsnr

Windows:
ascrsctl create -n mydb -t db -oh c:\oraasshare\cfcdb1 -sid orcl
          -disk ohdisk datafiledisk -lsnr mydblsnr
```

D.13 update/db

TOPIC

```
update/db - update db ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> -type db [-oraHome <string>]
          [-oraSID <string>] [-disk <string> [<string> ...]]
          [-lsnr <string>] [-pfile <string>]
          [-vip <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update an ASCRS **db** resource registered in CRS.

Mandatory argument:

-name, -n

This argument specifies the resource name to be updated.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type. Its value must be **db**. If the resource name is in canonical form, the **-type** option can be omitted.

-oraHome, -oh

This argument specifies the database Oracle Home location.

-oraSID, -sid

This argument specifies the Oracle SID name

-disk

For a database instance component, the value for this parameter is a space-specified or comma-specified list of ASCRS disk resource names; for other components, its value is the disk resource hosting the Oracle home.

lsnr

This argument specifies the listener resource. It is only applicable for the dbinstance component.

-vip

This argument specifies the virtual IP resource used by a database console, so it is applicable for a database console resource only.

The other non-mandatory options:

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all the nodes.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–2, where, except for **as**, **st** and **ra**, all the other numbers are in seconds.

-pfile, -pf

This argument specifies the database pfile for starting the database. The special value "nil" sets this file to its default. It is only valid for the dbinstance component.

EXAMPLE(S)

```
ascrsctl update -n ora.mydb.cfcdb -disk ohdisk -lsnr newlsnr
            -p st=60,rt=60,pt=60
```

D.14 create/as

TOPIC

```
create/as - create as ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type as -componentHome <string>
            [-componentIDs <string> [<string> ...]]
            -vip <string> -disk <string> [<string> ...]
            [-db <string> [<string> ...]]
            [-as <string> [<string> ...]]
            [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a middleware resource in CRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **as**.

-componentHome, -ch

This argument specifies the location of the WebLogic domain that contains the targeted Administration Server, managed servers, or the OPMN managed servers.

-vip

This argument specifies the virtual IP resource the middleware servers depend upon.

-disk

This argument specifies the shared disks hosting the WebLogic software, the WebLogic server domain directory, or, if it is for OPMN managed components, the instance home, and other shared disks this resource directly depends upon. If a shared disk is used for multiple purposes, it needs to be specified only once. The value for this parameter is a space-separated or comma-separated list of ASCRS disk resource names.

Non-mandatory options:

-componentIDs, -ci

This argument specifies the names of the servers to be managed in the WebLogic domain or the OPMN instance. If this argument is missing or if the value is **default**, all the servers are included.

db

This argument specifies the database resources this **as** resource directly depends upon.

as

This argument specifies the **as** resources this resource directly depends upon.

-m

This argument specifies the health monitors that the WebLogic servers should use. If this option is missing or its value is **default**, the TCP ping monitor is used for all the managed servers. To use user-defined monitors, this option should be followed by a list of monitor assignments such as '**-m** AdminServer=mon1 wlsapp=mon2', where AdminServer and wlsapp are valid server names, while mon1 and mon2 are valid monitor names defined in *CRS_HOME*`/ascrs/config/mconfig.xml`.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–1, where, except for **as**, **st** and **ra**, all the other numbers are in seconds. These properties can be configured through the ASCRS configuration file. If a property is neither configured nor specified with this option, the default is assumed.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ASCRS configuration file.

The valid values are **normal** or **fast**. If it is not specified, **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
UNIX:
ascrsctl create -n idm.weblogic -t as
          -ch /sharedisk/fmw/user_projects/domains/IDMDomain
          -vip idmvip -disk idmdisk

ascrsctl create -n idm.weblogic -t as
          -ch /sharedisk/fmw/user_projects/domains/IDMDomain
          -vip idmvip -disk idmdisk -p st=800,rt=800,pt=800

ascrsctl create -n idm.opmn -t as
          -ch /sharedisk/fmw/asinst_1
          -vip idmvip -disk idmdisk -p st=800,rt=800,pt=800

Windows:
ascrsctl create -n adminserver -t as
          -ch c:\oracle\asdisk\fmw\user_projects\domains\adminserverDomain
          -vip adminservervip -disk adminserverdisk
```

D.15 update/as

TOPIC

update/as - update **as** ASCRS resource

COMMAND

```
ascrsctl update -name <string> [-type as] [-componentHome <string>]
```

```
[-componentIDs <string> [<string> ...]]
[-vip <string>] [-disk <string> [<string> ...]]
[-db <string> [<string> ...]]
[-as <string> [<string> ...]]
[options]
```

DESCRIPTION

This ASCRS command is used to update an **as** resource created with ASCRS. On Windows, OPMN resources are not supported.

Mandatory argument:

-name, -n

This argument specifies the resource name to be updated.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type. Its value must be **as**. If the resource name is in canonical form, the **-type** option can be omitted.

-componentHome, -ch

This argument specifies the location of the WebLogic domain or OPMN instance home.

-componentIDs, -ci

This argument specifies the names of the servers managed in the WebLogic domain or the OPMN instance. Value 'default' means all the server names are included.

-vip

This argument specifies the virtual IP resource this component depends upon.

-disk

This argument specifies the shared **disk** resources this **as** resource directly depends upon. The value is a space-separated or comma-separated list of disk resource names.

-db

This argument specifies the database resources this **as** resource directly depends upon. Value 'nil' removes all these dependencies.

-as

This argument specifies the **as** resources this resource directly depends upon. Value 'nil' removes all these dependencies.

-m

This argument specifies the health monitors that WebLogic servers should use. If its value is **default**, the TCP ping monitor is used for all the managed servers. To fine-tune the monitor assignments, this option should be followed by a list of monitor assignments such as '**-m** AdminServer=mon1 wlsapp=mon2', where AdminServer and wlsapp are valid server names, while mon1 and mon2 are either valid monitor names defined in CRS_HOME/ascrs/config/mconfig.xml or 'default'.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster that can host this resource. The value is a space-separated or comma-separated list of node names in the cluster. The special value **default** includes all the nodes.

-resourceParams, -params, -p

This argument specifies space-separated or comma-separated "name=value" pairs to set the CRS properties for this resource, for example, **as**=1,**rt**=400.

The property names and their value ranges are listed in Table D–2, where, except for **as**, **st** and **ra**, all the other numbers are in seconds.

EXAMPLE(S)

```
ascrsctl update -n myas -t as -vip newvip -disk instdisk wldisk
ascrsctl update -n ora.myas.cfcas -p st=800,rt=800,pt=800
```

E

Configuring Distributed Notifications for MDS

The MDS database repository has the ability to use Java Object Cache for caching MetadataObjects and their contents. Each node using MDS has its own Java Object Cache instance and will cache MetadataObjects as they are read from the repository by that node, invalidating them from the local cache when local updates to documents comprising that MetadataObject occur.

MDS does not use Distributed Java Object Cache for replicating MetadataObjects. MDS detects changes made to the repository by other nodes whenever a local change is flushed to that repository, and invalidates the relevant local cache entries accordingly. In addition, each cluster member polls the repository every 30 seconds (this is the default interval, which can be configured) to learn if any documents have changed and will invalidate local cache entries in the same way.

However, the MDS database repository also has the capability of sending cluster-wide notifications when a cluster member changes a document in that repository. When this feature is enabled, all cluster members are notified of changes to documents when the change occurs, instead of waiting for a local commit to be performed or for the repository poll to be run. Therefore, in a high availability environment, Oracle recommends enabling distributed notifications for performance reasons. Configuring the distributed Java Object Cache on every cluster member enables distributed notifications. When a cluster member receives notification of a change to a document, it invalidates its current local version of that document and uses the updated document instead.

For this feature to work, you must configure Java Object Cache on each node, ensure that the JNDI name that each node is using to refer to the repository is identical on every node, and verify that polling is enabled. (Polling is enabled by default, but it must not be disabled).

For instructions on configuring the distributed Java Object Cache on the nodes in a cluster, see the topic "Setting up the Java Object Cache" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.