Oracle® Fusion Middleware

High Availability Guide

12*c* (12.2.1.0.0)

E56928-01

October 2015

Oracle Fusion Middleware High Availability Guide, 12*c* (12.2.1.0.0)

E56928-01

Primary Author: Christine Ford

Contributing Author: Michael Blevins, Suvendu Ray, Lingaraj Nayak, Maneesh Jain, Peter LaQuerre

Contributor: Gururaj BS

Contents

Preface

This preface contains these sections:

- Audience
- Purpose of this Guide
- Documentation Accessibility
- Related Documents
- Conventions

Audience

The *Oracle Fusion Middleware High Availability Guide*12*c* (12.2.1.0.0) is intended for administrators, developers, and others whose role is to deploy and manage Oracle Fusion Middleware with high availability requirements.

Purpose of this Guide

The purpose of this guide is to serve as a reference document to set up a highly available environment. Use this guide in conjunction with your product's installation and administration guides.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see these Oracle resources:

- *Oracle Fusion Middleware Understanding Oracle Fusion Middleware Concepts*
- *Oracle Fusion Middleware Administering Oracle Fusion Middleware*

- *Oracle Fusion Middleware Tuning Performance Guide*
- *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server*

For definitions of unfamiliar terms found in this and other books, see the Glossary.

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.

Part I

Introduction to High Availability

Part I contains the following topics:

1 Introduction and Roadmap

This section includes information on how and why to use this guide and high availability environments. This section includes the following topics:

- Section 1.1, "How to Use This Guide."
- Section 1.2, "Setting up a Highly Available Environment"
- Section 1.3, "New and Changed Features in This Release"
- Section 1.4, "What is High Availability?"
- Section 1.5, "High Availability Solutions"
- Section 1.6, "Understanding the Oracle Fusion Middleware Standard HA Topology"

1.1 How to Use This Guide

Use this document as a reference guide for information on high availability concepts and tasks as you set up a highly available environment.

Before you use this guide, you must have a standard installation topology set up for your product. This is the required starting point for setting up high availability. See the topics "Understanding the Oracle Fusion Middleware Infrastructure Standard Installation Topology" and "Roadmap for Installing and Configuring the Standard Installation Topology" to set up the standard installation topology.

1.2 Setting up a Highly Available Environment

The following table describes tasks to set up a highly available environment and resources for information that is not in this guide.

Table 1–1 Setting up a Highly Available Environment

Task	Description	For more information
Performing administrative tasks and preparing your environment	Common tasks to perform on a newly-created domain.	See "Administering and Preparing your WebLogic Domain for High Availability" in your product installation guide.
Planning your WebLogic Server Installation	Covers understanding your topology and determining the distribution, components, and features you need.	See *Planning an Installation of Oracle Fusion Middleware*

Table 1–1 (Cont.) Setting up a Highly Available Environment

Task	Description	For more information
Installing the WebLogic Server Software	Describes how to start the installation process and go through installation screens.	See the topic "Installing the Oracle Fusion Middleware Infrastructure Software" in your product installation guide.
Configuring a domain	Creating and configuring a domain	See "Configuring your Oracle Fusion Middleware Infrastructure Domain" in your product installation guide.
Managing Oracle Fusion Middleware	Includes how to: start and stop, change ports, deploy applications, and back up and recover Oracle Fusion Middleware.	See *Oracle Fusion Middleware Administrator's Guide*
Monitoring and optimizing performance in the Oracle Fusion Middleware environment.	For components that affect performance, use multiple components for optimal performance, and design applications for performance.	See *Oracle Fusion Middleware Tuning Performance Guide*
Setting up a product-specific enterprise deployment	Oracle best practices blueprints based on proven Oracle high availability and security technologies and recommendations for a product-specific enterprise deployment.	See your product's Enterprise Deployment Guide
Administering the product environment	To deploy, manage, monitor, and configure applications using the product.	See your product's Administrator's Guide
Configuring Node Manager	Use Node Manager to start, shut down, and restart the Administration Server and Managed Servers from a remote location. It is an essential tool for a high availability environment.	See *Administering Node Manager for Oracle WebLogic Server*

1.3 New and Changed Features in This Release

Oracle Fusion Middleware 12*c* (12.2.1.0.0) includes the following new and changed concepts and features:

- Support for WebCenter. See Chapter 10, "Configuring High Availability for Oracle WebCenter Components."
- Support for BI. See Section 11.3, "Deploying BI."

See Also: For a comprehensive list of new and deprecated:

- **WebLogic Server features** in this release, see *Oracle Fusion Middleware What's New in Oracle WebLogic Server.*
- **Terms** in this release, see "New and Deprecated Terminology for 12c" in *Understanding Oracle Fusion Middleware Concepts*

1.4 What is High Availability?

High availability is the ability of a system or device to be available when it is needed.

A high availability architecture ensures that users can 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.

High availability comes from redundant systems and components. You can categorize high availability solutions by their level of redundancy into **active-active** solutions and **active-passive** solutions.

See the following topics:

- Section 1.4.1, "Active-Active High Availability Solutions"
- Section 1.4.2, "Active-Passive High Availability Solutions"

1.4.1 Active-Active High Availability Solutions

An **active-active solution** deploys two or more active servers to improve scalability and provide high availability. In active-active deployments, all instances handle requests concurrently. Oracle recommends active-active solutions for all single-site middleware deployments.

1.4.2 Active-Passive High Availability Solutions

An **active-passive solution** deploys one active instance that handles requests and one passive instance that is 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 a failed node after failover. Oracle supports active-passive deployments for all components.

1.5 High Availability Solutions

You can categorize high availability solutions into local **high availability solutions** that provide high availability in a single data center deployment, and **disaster recovery solutions**.

Local high availability solutions can protect against process, node, and media failures, as well as human errors, ensuring availability in a single data center deployment.

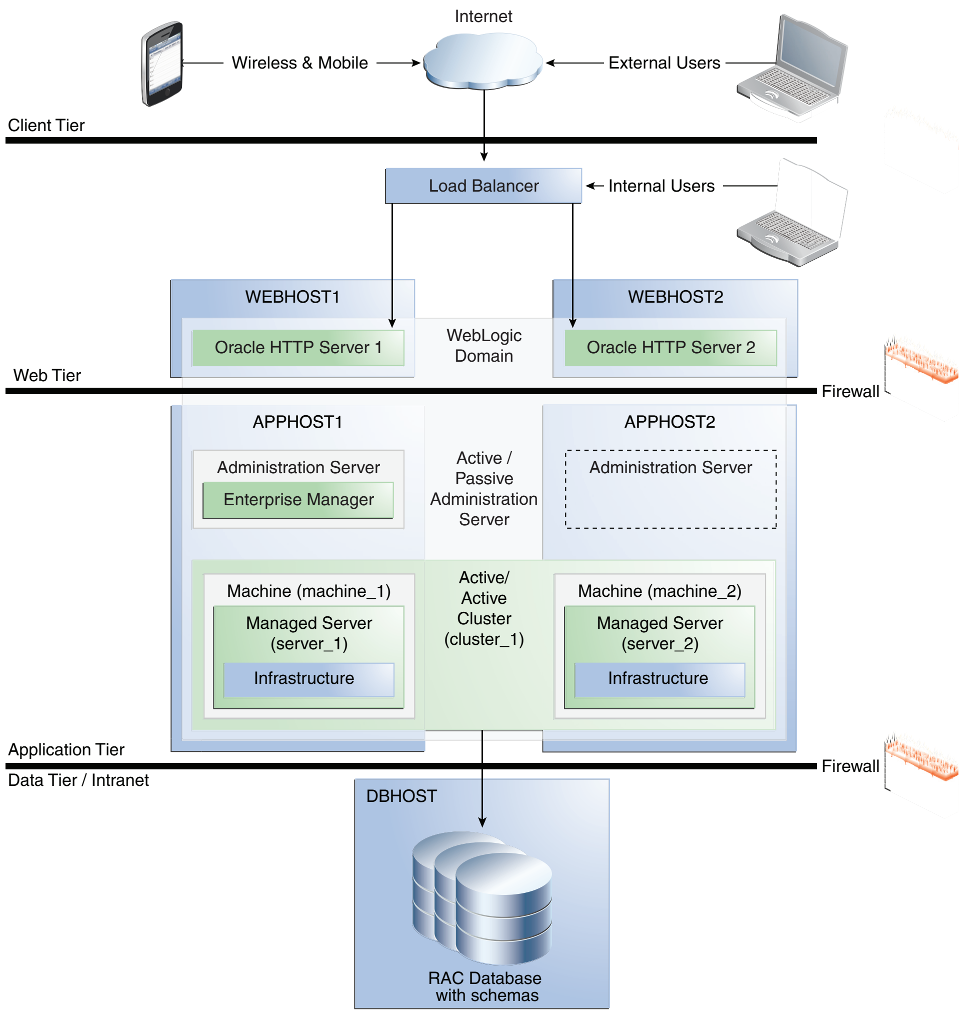
Disaster recovery solutions are usually geographically distributed deployments that protect your applications from disasters such as floods or regional network outages. You can protect against physical disasters that affect an entire data center by deploying geographically-distributed disaster recovery solutions. For more on disaster recovery for Oracle Fusion Middleware components, see *Oracle Fusion Middleware Disaster Recovery Guide*

1.6 Understanding the Oracle Fusion Middleware Standard HA Topology

The following figure shows the recommended standard high availability topology for a local, highly available Oracle Fusion Middleware deployment.

This deployment is consistent with the infrastructure standard installation topology and Oracle HTTP Server standard installation topology if you followed steps in *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure* and *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server*.

Figure 1–1 Oracle Fusion Middleware Highly Available Deployment Topology (Typical Enterprise)



This topology shows a multi-tiered architecture. Users access the system from the client tier. Requests go through a hardware load balancer, which routes them to Web servers running Oracle HTTP Servers in the web tier. Web servers use Proxy Plug-in (`mod_wl_ohs`) to route 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.

The following table describes elements in the preceding figure.

Table 1–2 Description of the Elements in the Oracle Fusion Middleware Infrastructure Standard High Availability Topology

Element	**Description and Links to Additional Documentation**
APPHOST	Machine that hosts the application tier.
WEBHOST	Machine that hosts the web tier.
WebLogic Domain	A logically related group of Java components, in this case, the Administration Server, Managed Servers, and other software components. For more information, see "What is an Oracle WebLogic Server Domain?" in *Understanding Oracle Fusion Middleware.*
Administration Server	Central control entity of a domain. Maintains a domain's configuration objects and distributes configuration changes to Managed Servers.
Enterprise Manager	Oracle Enterprise Manager Fusion Middleware Control. The main tool that you use to manage a domain.
Cluster	A collection of multiple WebLogic Server instances running simultaneously and working together.
Machine	Logical representation of the computer that hosts one or more WebLogic Server instances (servers). Machines are the logical 'glue' between Managed Servers and Node Manager; to start or stop a Managed Server with Node Manager, the Managed Server must be associated with a machine.
Managed Server	Host for applications, application components, Web services, and their associated resources. See "Oracle Enterprise Manager Fusion Middleware Control" in *Understanding Oracle Fusion Middleware*.
Infrastructure	Collection of services that includes: ■ Metadata repository (MDS). Contains metadata for components, such as Oracle Application Developer Framework. See "What is the Metadata Repository?" in *Understanding Oracle Fusion Middleware*. ■ Oracle Application Developer Framework (Oracle ADF) ■ Oracle Web Services Manager (OWSM)

See Also:

- To view a figure of the Infrastructure Standard Installation Topology and follow a roadmap to install it, see "Understanding the Infrastructure Standard Installation Topology" in *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure*.
- To view a figure of the Oracle HTTP Server Standard Installation Topology and follow a roadmap to install it, see "Introducing the Oracle HTTP Server Standard Installation Topologies" in *Installing and Configuring Oracle HTTP Server*.

2 High Availability Concepts

This section describes high availability concepts and includes the following topics:

2.1 Oracle Fusion Middleware Concepts

For information on Oracle Fusion Middleware concepts, see the following topics in *Oracle Fusion Middleware Understanding Oracle Fusion Middleware Concepts*.

Table 2–1 Oracle Fusion Middleware Concepts

For information on...	See this topic...
Oracle Home, Oracle Common, WebLogic Server Domain	"What are the Key Oracle Fusion Middleware Directories?"
WebLogic Server Domain	"What is an Oracle WebLogic Server Domain?"
Administration Server	"What is the Administration Server?"
Managed Servers and Managed Server Clusters	"Understanding Managed Servers and Managed Server Clusters"
Node Manager	"What is Node Manager?"

See Also: "Communications in a Cluster" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server*

2.2 Server Load Balancing in a High Availability Environment

This section includes the following topics:

2.2.1 About Load Balancing

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

Typically, a load balancer front ends high availability deployments. You use Oracle HTTP Server or Oracle Traffic Director to configure load balancing between different components or applications. See Section 2.2.4, "Server Load Balancing with Oracle HTTP Server or Oracle Traffic Director."

You can also combine of a load balancer and Oracle HTTP Server (as Figure 1–1 shows) to provide maximum availability.

2.2.2 Third-Party Load Balancer Requirements

You can use third-party load balancers in your Oracle Fusion Middleware high availability deployments. Oracle recommends load balancers that support 'sticky' session routing. **Sticky session routing** is the ability, for the entire session, to route traffic to the same server that processes the first request.

Your external load balancer must have the following features:

- Ability to load-balance traffic to a pool of real servers through a virtual host name: clients access services using the *virtual* host name instead of *actual* host names. The load balancer can then load balance requests to servers in the pool. Typically, the load balancer can balance across Oracle HTTP Server instances and then the Oracle HTTP Servers can balance across application servers.
- Port translation configuration.
- Monitoring of ports (HTTP and HTTPS).
- Ability to configure virtual server names and ports on the load balancer.
- Configuration of multiple virtual servers. For each virtual server, the load balancer should allow configuration of traffic management on more than one port. For example, for clusters, you must configure the load balancer with a virtual server and ports for HTTP and HTTPS traffic.
- 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 virtual server names.
- Resource monitoring/port monitoring/process failure detection: the load balancer must be able to detect service and node failures and to stop directing non-Oracle Net traffic to a failed node. If your external load balancer can automatically detect failures, Oracle recommends that you use it.
- Fault tolerant mode: Oracle recommends that you configure the load balancer to be in fault-tolerant mode.
- Virtual server returning to calling client: Oracle highly recommends that you configure the load balancer virtual server to return immediately to the calling client when back-end services that it forwards traffic to are unavailable. This is

preferred over a client disconnecting on its own after a timeout based on TCP/IP settings on a client machine.

- SSL acceleration. Oracle recommends this feature but does not require it.
- Client IP Address (Preserving): the load balancer must be able to insert a request's original client IP address in an X-Forwarded-For HTTP header to preserve it.

2.2.3 Configuring Third-Party Load Balancers

Detailed load balancer configuration steps depend on:

- The environment you are using the load balancer in.
- The type of load balancer you are using.

For these reasons, Oracle recommends that you follow your load balancer's documentation. For high-level load balancer configuration steps, see the enterprise deployment guide for the component you are working with.

2.2.4 Server Load Balancing with Oracle HTTP Server or Oracle Traffic Director

This section describes Oracle load balancing products.

Table 2–2 Server Load Balancing Products

Product	Description
Oracle HTTP Server (OHS)	Web server with a built-in WebLogic Server Proxy Plug-In module to act as HTTP front-end for one or more WebLogic servers. Receives incoming requests from clients and load balances each request to one or more WebLogic Servers. The OHS `mod_wl_ohs` module routes requests to one or more WebLogic servers. Has the same load balancing functionality as `mod_weblogic` (Oracle WebLogic Server Plug-in for Apache HTTP Server). See "Configuring the mod_wl_ohs Plug-In for Oracle HTTP Server" in *Oracle Fusion Middleware Using Web Server 1.1 Plug-Ins with Oracle WebLogic Server* for more information. See Section 9.9.1.6, "Configuring mod_wl_ohs.conf" for more on the `mod_wl_ohs` module.
Oracle Traffic Director	Highly available Application Delivery Controller with WebLogic inter-operability enhancements. Efficiently throttles requests to one or more clusters. A fast, reliable, and scalable layer-7 software load balancer. Reliable entry point for all HTTP, HTTPS and TCP traffic. Distributes client requests based on specified load-balancing method, routes requests based on specified rules, caches frequently accessed data, prioritizes traffic, and controls service quality. See Oracle Traffic Director Administrator's Guide for more information.

2.3 Application Failover

This section describes different types of failover and behavior. Topics include the following:

- Section 2.3.1, "About Failover, Application Failover, and State"
- Section 2.3.2, "Session Failover Requirements"
- Section 2.3.3, "Application Failover Expected Behavior"

2.3.1 About Failover, Application Failover, and State

Failover is relocating an overloaded or failed resource such as a server, disk drive, or network to its backup location.

Application failover is when an application component doing a certain job becomes unavailable and a copy of the failed component finishes the job.

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

2.3.2 Session Failover Requirements

Note: Oracle applications meet these session failover requirements unless a specific exception is made for an application.

For seamless failover, an application must meet the following conditions:

- The application is in a cluster and at least one member of the application cluster is available to serve the request.
- For stateful applications, state replication is configured correctly.
- If you use Oracle HTTP Server, the server is configured with the WebLogic Cluster directive to balance among all available application instances.
- If you are using a hardware load balancer, the load balancer is:
  - Routing traffic to all available instances
  - Configured correctly with a health monitor to mark unavailable instances
  - Configured to support persistence of session state

2.3.3 Application Failover Expected Behavior

If you configure the environment correctly, users don't notice when an application instance in a cluster becomes unavailable. The application failover sequence of events is, for example:

1. A user makes a request and a hardware load balancer routes it to Instance A of the application.
2. Instance A of the application becomes unavailable due to node failure, process failure, or network failure.
3. The hardware load balancer marks Instance A as unavailable.
4. The user makes a subsequent request. The request is routed to Instance B.
5. Instance B is configured as a replication partner of Instance A and has the user's session state.
6. The application resumes using the session state on Instance B and the user continues working without interruption.

See Also: See *Domain Template Reference* for domain and extension templates that support high availability.

See Failover and Replication in a Cluster in *Administering Clusters for Oracle WebLogic Server* for more on failover and replication at the application level.

2.4 Real Application Clusters

Oracle Real Application Clusters (RAC) enable you to cluster an Oracle database. A **cluster** comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle RAC uses Oracle Clusterware for the infrastructure to bind multiple servers so they operate as one system. Along with a collection of hardware (cluster), Oracle RAC unites the processing power of each component to become a single, robust computing environment. Oracle RAC provides a highly scalable and highly available database for Oracle Fusion Middleware.

Every Oracle RAC instance in a cluster has equal access and authority. Node and instance failure may affect performance but do not result in downtime because the database service is available or can be made available on surviving server instances.

See Also: For more information on Oracle RAC see:

- *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server*
- *Oracle Real Application Clusters Administration and Deployment Guide*
- *Oracle Clusterware Administration and Deployment Guide*

2.5 Coherence Clusters and High Availability

If you follow *Oracle Fusion Middleware Installing and Configuring Oracle WebLogic Server and Coherence* or a product installation guide, the standard installation topology includes a standard Coherence cluster that is a starting point for additional configuration.

A **Coherence cluster** is a collection of Java Virtual Machine (JVM) processes running Coherence. In 12c, these processes are called **WebLogic Managed Coherence Servers**. JVMs that join a cluster are **cluster members** or **cluster nodes**. Cluster members can be:

- Dedicated storage members
- Client members that have storage disabled
- Proxy members that allow non-cluster members to access Coherence caches

Cluster members communicate using Tangosol Cluster Management Protocol (TCMP). Cluster members use TCMP for both multicast communication (broadcast) and unicast communication (point-to-point communication).

Coherence characteristics include the following:

- Each domain typically contains one Coherence Cluster.
- Each managed Coherence server in a domain is associated with a Coherence cluster, defined through a Coherence Cluster System Resource.

- Each application has its Coherence configuration in a Grid Archive (GAR) file, which deploys with the application and to all dedicated storage nodes.

All applications that use Coherence use the cluster associated with the managed Coherence server and deploy their GAR files co-located with their applications. The following table lists sources of information about Coherence.

Table 2–3 Coherence and Coherence Clusters

For information on...	See this topic...
Coherence concepts and features	"Introduction to Coherence" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*
Creating Coherence clusters	"Setting Up a WebLogic Server Domain Topology for Coherence" in *Coherence Administrator's Guide*
Configuring a Coherence Cluster	"Configuring and Managing Coherence Clusters" in *Administering Clusters for Oracle WebLogic Server*

2.6 Disaster Recovery

For maximum availability, you may need to deploy services at different geographical locations to protect against entire site failures due to unforeseen disasters and natural calamities. Oracle Fusion Middleware products support the configuration of a geographically separate standby site to act as a backup. Applications and services can fail over to this backup in the event of natural or unplanned outages at a production site.

For more on disaster recovery, see *Oracle Fusion Middleware Disaster Recovery Guide*.

2.7 Install Time Configuration (Profiles)

This section includes the following topics:

- Section 2.7.1, "Domain (Topology) Profiles"
- Section 2.7.2, "Component/Service Database and File Persistence Profiles"
- Section 2.7.3, "Post-Configuration Defaults"

2.7.1 Domain (Topology) Profiles

Use the Configuration Wizard or WebLogic Scripting Tool (offline) to set up domains. See *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard* to create, update, and configure domains.

All 12*c* (12.2.1.0.0) installation guides have steps to set up a single machine, multi-server domain, which is the **standard installation topology**. Section 1.6, "Understanding the Oracle Fusion Middleware Standard HA Topology" describes this topology in detail. For more information see:

- *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server*
- *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure*

2.7.2 Component/Service Database and File Persistence Profiles

Persistence profiles are a collection of settings that Oracle recommends for a specific persistence environment. There are two primary persistence profiles: database and file based. The following table shows persistence types for both profiles.

Table 2–4 Persistence Types for Database and File Persistence Profiles

Component/Service	Database Persistence Profile	File Persistence Profile
JMS	WLS JMS in Database Store	WebLogic Server JMS in File Store
JTA	JTA in Database Store	JTA in File Store
OPSS	Database Store	Database Store
MDS	Database Store	Database Store
Service Table	Database Store	Database Store
Failover	Whole Server Migration	Whole Server Migration

Although you can "mix & match" a component or service with the persistence profile, the persistence type groups work together optimally. Oracle recommends that you use all options consistently within their respective profile.

Note: An MDS data source has a WebLogic Server file persistence store allocated along with the data source. Because you use the file persistence store only in development mode, you can ignore it for high availability purposes. You don't need to recover the file persistence store in the event of failure.

See Also: See "Interoperability with Supported Databases" in the *Interoperability and Compatibility Guide* for database version requirements for selected products and features.

2.7.3 Post-Configuration Defaults

When you complete a standard installation, a domain is set up with a file-based persistence profile. To configure database-based persistence for JMS/JTA resources, see Chapter 7, "JMS and JTA High Availability." To set up whole server migration, see Chapter 3, "Whole Server Migration."

Note: Some products may have specific requirements for shared file stores; Oracle recommends that you refer to your product's requirements for details.

The following table describes additional sources of information.

Table 2–5 Domain Configuration Topics

For additional information on...	See this topic...
Shared file systems to use with the file persistence profile	Chapter 4, "Using Shared Storage"
JMS and JTA	Section 7.2, "About Migratable Targets for JMS and JTA Services"
Failover	Section 2.3, "Application Failover"

2.8 Application and Service Failover

*****This topic was not included in 12.1.3, to be revisited for 12.2.1. Whole Server Migration content moved to that new topic.****

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. **Whole server migration** occurs when the server instance migrates to a different physical system upon failure. **Service-level migration** is when services move to a different server instance within the cluster.

This section describes server and service failover for JMS and JTA.

2.8.1 About Automatic Service Migration (JMS/JTA)

Service-level migration in WebLogic Server is the process of moving pinned services from one server instance to a different available server instance in the cluster.

You can configure JMS and JTA services for high availability by using migratable targets. A **migratable target** is 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. High availability is achieved by migrating a migratable target from one clustered server to another when a problem occurs on the original server. When the migratable target migrates, all services that the target hosts migrate.

If a product does not support Automatic Service Migration, you can use whole service migration.

See Also: For more information, see "Service Migration" in *Oracle Fusion Middleware Administering Clusters for Oracle* for the following topics:

- "Understanding the Service Migration Framework"
- "Pre-Migration Requirements"
- "Roadmap for Configuring Automatic Migration of JMS-related Services"
- "Roadmap for Configuring Automatic Migration of the JTA Transaction Recovery Service"

2.9 Roadmap for Setting Up a High Availability Topology

The following table describes high level steps required to set up an example middleware topology with high availability.

Table 2–6 Roadmap for Setting Up a High Availability Topology

Task	Description	Documentation
1. Install Real Application Clusters	Install Real Application Clusters	*Oracle Real Application Clusters Administration and Deployment Guide*

Table 2–6 (Cont.) Roadmap for Setting Up a High Availability Topology

Task	Description	Documentation
2. Install and configure middleware components	Install and configure the application by following instructions in an application installation guide.	*Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure* or the installation guide for your product
3. Install and configure Oracle HTTP Server	Install and configure Oracle HTTP Server in the same domain	*Installing and Configuring Oracle HTTP Server*
4. Configure a load balancer	Configure a third-party load balancer that meets specific requirements, or Oracle HTTP Server/Oracle Traffic Director.	Section 2.2, "Server Load Balancing in a High Availability Environment."
5. Scale out the topology (machine scale out)Oracle Fusion MiddlewareOracle Fusion Middleware	Steps for scaling out a topology (machine scale-out) for all products that are part of a Fusion Middleware WebLogic Server domain.	Chapter 6, "Scaling Out a Topology (Machine Scale Out)"
6. Configure high availability for the Administration Server	Configure high availability for the Administration Server	Chapter 8, "Administration Server High Availability"

3 Whole Server Migration

This section describes whole server migration and how to configure it for Managed Server failover. When **whole server migration** occurs, a server instance migrates to a different physical machine upon failure. You must configure whole server migration if your environment uses special (pinned) services such as JMS and JTA.

This section includes the following topics:

- Section 3.1, "About Whole Server Migration"
- Section 3.2, "Configuring Whole Server Migration for Managed Server Failover"

3.1 About Whole Server Migration

A cluster provides high availability and failover by duplicating an object or service on redundant Managed Servers in the cluster. However, some services, such as JMS servers and JTA transaction recovery service, are designed with the assumption that there is only *one* active instance of the service running in a cluster at any given time. These types of services are known as **pinned services** because they remain active on only one server at a time.

Most services deploy homogeneously on all Managed Servers in a cluster, enabling transparent failover from one Managed Server to another. However, pinned services target *individual* server instances in a cluster. For pinned services, WebLogic Server supports failure recovery with migration instead of failover.

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 is 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 migrates to another system. Also, you can manually start migration of a server instance.

See Also:

See "Whole Server Migration" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* to prepare for automatic whole server migration, configure automatic whole server migration, and server migration processes and communications.

See "Service Details" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* for details on service migration mechanisms.

See Chapter 7, "JMS and JTA High Availability" for details on JMS and JTA services.

3.2 Configuring Whole Server Migration for Managed Server Failover

This section describes high level steps to configure a cluster for failover using whole server migration. If one server in the cluster fails, whole server migration restarts it on another machine.

Your system must meet specific requirements before you configure automatic whole server migration. See "Preparing for Automatic Whole Server Migration" in *Administering Clusters for Oracle WLS*.

To configure whole server migration, follow the steps in "Configuring Whole Server Migration" in *Administering Clusters for Oracle WebLogic Server*.

See these topics for more information on configuring whole server migration:

- "Using High Availability Storage for State Data"
- "Server Migration Processes and Communications"

Part II

Creating a High Availability Environment

Part II describes recommendations and steps you need to take to create a high availability environment.

This part contains the following topics:

4 Using Shared Storage

This section provides basic recommendations for using shared storage in a high availability environment. It describes the benefits of placing artifacts in a common location that multiple hosts or servers share. This common location typically resides in a shared file system, which is mounted on each server with standard operating system protocols such as NFS and CIFS.

This section includes the following topics:

4.1 Overview of Shared Storage

Shared storage allows sharing of dynamic state and server configuration. It simplifies administration, configuration, failover, and backup/recovery.

In a highly available environment, shared storage is required when using file based persistent stores (for JMS and JTA logs) and certain Oracle products. Shared storage is optional for product binaries and domain directories.

The following artifacts are typical candidates to place on a shared file system:

- **Product binaries**: All files and directories related to product executables, JAR files, and scripts that install during product installation.
- **Domain directory**: The directory containing the WebLogic Server domains and their configuration.
- **File-based persistent stores**: File-based persistent stores for JMS persistence and JTA transaction logs.

See the following table for more information about shared storage.

Table 4–1 Shared Storage Topics

Topic/Task	For More Information
Structure and contents of an Oracle home	"Understanding the Oracle Fusion Middleware Infrastructure Directory Structure" in *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure*
Saving JMS and JTA information in a file store	"The WebLogic Persistent Store" in Administering the WebLogic Server Persistent Store.
	"Persistent Store High Availability" in *Administering JMS Resources for Oracle WebLogic Server*
	Default File Store Availability for JTA in *Administering Clusters for Oracle WebLogic Server*

4.2 Shared Storage Prerequisites

The following shared storage prerequisites apply only when you use file-based persistent stores:

- For proper recovery in the event of a failure, you must store both JMS and JTA transaction logs in a location that is accessible to all nodes that can resume operations after a Managed Server failure. This setup requires a shared storage location that multiple nodes can reference. See Section 4.6, "Directory Structure and Configurations" for the recommended directory structure.
- Oracle recommends that you use a shared storage device that is network-attached storage (NAS) or storage area network (SAN).

  If you use NFS-mounted systems, issues related to file locking and abrupt node failures have been detected. See Section 7.6, "Using File Stores on NFS" and check with your storage vendor for the main recommended parameters for mount options.

  The following example command is based on a NAS device. Note: your options may be different from those in this example; see UNIX/Linux documentation for more on the mount command and its options.

  ```
  mount nasfiler:/vol/vol1/u01/oracle /u01/oracle -t nfs -o
  rw,bg,hard,nointr,tcp,vers=3,timeo=300,rsize=32768,wsize=32768
  ```
- For maximum availability, Oracle recommends a *highly available* NAS or SAN device for shared storage. Shared storage devices that are not highly available can be a single point of failure. Check with your storage provider for options to achieve this.

For more information about saving JMS and JTA information in a file store, see "The WebLogic Persistent Store" in Administering the WebLogic Server Persistent Store.

4.3 Using Shared Storage for Binary (Oracle Home) Directories

The following sections describe guidelines for using shared storage for your Oracle home directories:

- Section 4.3.1, "About the Binary (Oracle Home) Directories"
- Section 4.3.2, "About Sharing a Single Oracle Home"
- Section 4.3.3, "About Using Redundant Binary (Oracle Home) Directories"

4.3.1 About the Binary (Oracle Home) Directories

When you install any Oracle Fusion Middleware product, you install the product binaries into an Oracle home (*ORACLE_HOME*). The binary files are read-only and do not change unless you patch or upgrade the Oracle home to a newer version.

In a typical production environment, you save Oracle home files in a separate location from domain configuration files, which you create using the Oracle Fusion Middleware Configuration Wizard.

The Oracle home for an Oracle Fusion Middleware installation contains binaries for Oracle WebLogic Server, the Oracle Fusion Middleware infrastructure files, and any Oracle Fusion Middleware product-specific directories.

Note: By default, the Configuration Wizard writes its logs to the `logs` directory in Oracle home. If you use a read-only Oracle home, you must specify the `-log` option to redirect logs to a different directory.

See Also: For details on the structure and contents of an Oracle home, see "What are the Key Oracle Fusion Middleware Directories?" in *Oracle Fusion Middleware Understanding Oracle Fusion Middleware Concepts*.

4.3.2 About Sharing a Single Oracle Home

You can configure multiple servers from a single Oracle home. This allows you to install the Oracle home in a single location on a shared volume and reuse the Oracle home for multiple servers.

If multiple servers on different hosts share an Oracle home, there are some best practices to keep in mind. For example, because the Oracle inventory directory (`oraInventory`) is updated only on the host from which the Oracle home was originally installed, Oracle recommends that all subsequent operations you perform on the Oracle home (such as patching and upgrade) be carried out from that original host. If that host is unavailable, ensure that the Oracle inventory is updated on another host before applying patches or upgrades to the Oracle home from the other host.

For more information about `oraInventory`, see "Oracle Universal Installer Inventory" in the *Oracle Universal Installer Concepts Guide.*

4.3.3 About Using Redundant Binary (Oracle Home) Directories

For maximum availability, Oracle recommends using redundant binary installations on shared storage.

In this model, you install two identical Oracle homes for your Oracle Fusion Middleware software on two different shared volumes. You then mount one of the Oracle homes to one set of servers and the other Oracle home to the remaining servers. Each Oracle home has the same mount point, so the Oracle home always has the same path, regardless of which Oracle home the server is using.

If one Oracle home becomes corrupted or unavailable, only half your servers are affected. For additional protection, Oracle recommends that you disk mirror these volumes. To restore the affected servers to full functionality, you can simply remount the surviving Oracle Home.

If separate volumes are not available on shared storage, Oracle recommends simulating separate volumes using different directories within the same volume and mounting these to the same mount location on the host side. Although this does not guarantee the protection that multiple volumes provide, it does protect from user deletions and individual file corruption.

Note: For maximum protection, Oracle recommends that you evenly distribute the members of a cluster across redundant binary Oracle homes. This is particularly important if cluster members are not running on all available servers.

4.4 Using Shared Storage for Domain Configuration Files

The following sections describe guidelines for using shared storage for the Oracle WebLogic Server domain configuration files you create when you configure Oracle Fusion Middleware products in an enterprise deployment:

- Section 4.4.1, "About Oracle WebLogic Server Administration and Managed Server Domain Configuration Files"
- Section 4.4.2, "Shared Storage Considerations for Administration Server Configuration Directory"
- Section 4.4.3, "Shared Storage Considerations for Managed Server Configuration Files"

4.4.1 About Oracle WebLogic Server Administration and Managed Server Domain Configuration Files

When you configure an Oracle Fusion Middleware product, you create or extend an Oracle WebLogic Server domain. Each Oracle WebLogic Server domain consists of a single Administration Server and one or more Managed Servers.

WebLogic uses a replication protocol to push persisted changes on the Administration Server to all Managed Servers. This gives redundancy to the Managed Servers so that you can start them without the Administration Server running. This mode is called *Managed Server independence*.

For more information about Oracle WebLogic Server domains, see *Understanding Domain Configuration for Oracle WebLogic Server*.

4.4.2 Shared Storage Considerations for Administration Server Configuration Directory

Oracle does not require you to store domain configuration files in shared storage. However, to support Administration Server recovery, you must place the Administration Server configuration directory on shared storage and mount it on the host that the Administration Server runs on. If that host fails, you can mount the directory on a different host and bring up the failed Administration Server on the other host. See Chapter 8, "Administration Server High Availability."

4.4.3 Shared Storage Considerations for Managed Server Configuration Files

Oracle recommends that you keep Managed Server configuration files in local, or, host private, storage.

You can keep Managed Server configuration files on shared storage. However, doing so can affect performance because multiple servers concurrently access the same storage volume.

4.5 Shared Storage Requirements for JMS Stores and JTA Logs

When you use file-based persistence in a high availability setup, you must configure the JMS persistent stores and JTA transaction log directories to reside in shared storage.

For more information, see Section 7.5, "Considerations for Using File Persistence (WebLogic JMS)."

4.6 Directory Structure and Configurations

When you use shared storage, there are multiple ways to lay out storage elements. Oracle recommends the following best practices:

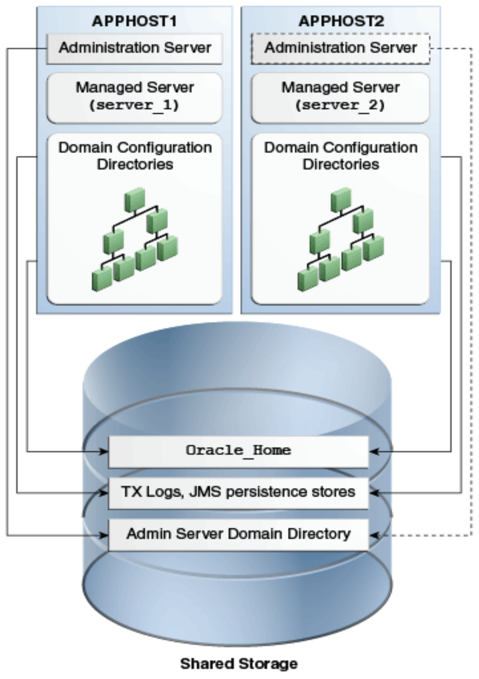
Table 4–2 Shared Storage Elements Directory Structure

Element	Location
ORACLE_HOME	Share in read-only mode by all servers.
JMS file stores and Transaction logs	Place on shared storage if you use file-based persistence.
Administration Server domain configuration directory	Place in shared storage to facilitate failing over the Administration server to a different host.

Note: Place Managed Server domain configuration directories on storage that is local to the corresponding host. See Section 4.4.2, "Shared Storage Considerations for Administration Server Configuration Directory" for more information.

The following figure illustrates the directory structure.

Figure 4–1 Shared Storage Directory Structure


APPHOST1
Administration Server
Managed Server (server_1)
Domain Configuration Directories
APPHOST2
Administration Server
Managed Server (server_2)
Domain Configuration Directories
Oracle_Home
TX Logs, JMS persistence stores
Admin Server Domain Directory
Shared Storage

5 Database Considerations

This section describes what to consider as you configure database connections for Oracle Fusion Middleware in a high availability setup. It also describes the benefits of using Oracle Real Application Clusters (Oracle RAC), a commonly-deployed database high availability solution. Most components use a database as the persistent store for their data. When you use an Oracle database, you can configure it in a variety of highly available configurations.

For more on database options, see *Oracle Database High Availability Overview*.

This section includes the following topics:

- Section 5.1, "About Oracle Real Application Clusters"
- Section 5.2, "Roadmap for Setting Up Oracle Real Application Clusters"
- Section 5.3, "About RAC Database Connections and Failover"
- Section 5.4, "About Data Sources"
- Section 5.5, "Configuring Active GridLink Data Sources with Oracle RAC"
- Section 5.6, "Configuring Multi Data Sources"

5.1 About Oracle Real Application Clusters

A **cluster** comprises multiple interconnected computers or servers that appear as one server to end users and applications. With Oracle RAC, you can cluster an Oracle database so that it is highly scalable and highly available.

All Oracle Fusion Middleware components that you deploy to Oracle WebLogic Server support Oracle RAC.

Every Oracle RAC instance in a cluster has equal access and authority. 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.

5.2 Roadmap for Setting Up Oracle Real Application Clusters

The following table outlines tasks and information to set up Oracle RAC.

Table 5–1 Roadmap for Setting up Oracle RAC

Task/Topic	More Information
About Oracle RAC	"Introduction to Oracle RAC" in *Oracle Real Application Clusters Administration and Deployment Guide*

Table 5–1 (Cont.) Roadmap for Setting up Oracle RAC

Task/Topic	More Information
Installing Oracle RAC	*Oracle Real Application Clusters Administration and Deployment Guide*
Managing Oracle RAC	"Overview of Managing Oracle RAC Environments" in *Oracle Real Application Clusters Administration and Deployment Guide*
Configuring and tuning GridLink and multi data sources	*Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*
Configuring Single Client Access Name (SCAN) URLs. (To specify the host and port for TNS and ONS listeners in WebLogic console.)	"SCAN Addresses" in *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

5.3 About RAC Database Connections and Failover

To establish connection pools, Oracle Fusion Middleware supports Active GridLink data sources and multi data sources for the Oracle RAC back end for both XA and non-XA JDBC drivers. These data sources also support load balancing across Oracle RAC nodes.

If an Oracle RAC node or instance fails, WebLogic Server or Oracle Thin JDBC driver redirects session requests to another node in the cluster. Existing connections don't failover. However, new application connection requests are managed using existing connections in the WebLogic pool or by new connections to the working Oracle RAC instance.

If the database is the transaction manager, in-flight transactions typically roll back.

If 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.

For more on XA Transactions, see Section 5.3.1, "About XA Transactions."

5.3.1 About XA Transactions

XA transaction support enables access to multiple resources (such as databases, application servers, message queues, transactional caches) within the same transaction. A *non-XA transaction* always involves just one resource.

An XA transaction involves a coordinating transaction manager with one or more databases, or other resources such as JMS, all involved in a single global transaction.

Java EE uses the terms **JTA transaction**, **XA transaction**, **user transaction**, and **global transaction** interchangeably to refer to one global transaction. This transaction type may include operations on multiple different XA- capable or non-XA resources and even different resource types. A JTA transaction is always associated with the current thread, and may pass from server to server as one application calls another. A common example of an XA transaction is one that includes both a WebLogic JMS operation and a JDBC (database) operation.

5.4 About Data Sources

A **data source** is an abstraction that components use to obtain connections to a relational database. Specific connection information, such as the URL or user name and password, are set on a data source object as properties. The application's code doesn't need to explicitly define the properties. Due to this abstraction, you can build applications in a portable manner, because they aren't tied to a specific back-end database. The database can change without affecting application code.

Active GridLink data sources and multi data sources support database connection high availability, load balancing, and failover. Oracle recommends the following data source types depending on your Oracle RAC database version:

- If you use Oracle RAC database version 11g Release 2 and later, use Active 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 Active GridLink data sources with Oracle RAC database for maximum availability. For Oracle RAC database versions that don't support Active GridLink data sources, Oracle recommends using multi data sources for high availability.

See the following topics for more information on data source types:

- Section 5.4.1, "Active GridLink Data Sources"
- Section 5.4.2, "Multi Data Sources"

5.4.1 Active GridLink Data Sources

An **Active GridLink data source** provides connectivity between WebLogic Server and an Oracle database service, which may include multiple Oracle RAC clusters. An Active GridLink data source has features of generic data sources plus the following support for Oracle RAC:

- Uses the ONS to respond to state changes in an Oracle RAC.
- Responds to Fast Application Notification (FAN) events to provide Fast Connection Failover (FCF), Runtime Connection Load-Balancing, and RAC instance graceful shutdown. FAN is a notification mechanism that Oracle RAC uses to quickly alert applications about configuration and workload.
- Provides Affinities (or XA Affinity) policies to ensure all database operations for a session are directed to the same instance of a RAC cluster for optimal performance.
- SCAN Addresses
- Secure Communication Using Oracle Wallet

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

- What is an Active GridLink Data Source
- Using Socket Direct Protocol
- Configuring Connection Pool Features
- Configuring Oracle Parameters
- Configuring an ONS Client
- Tuning Active GridLink Data Source Connection Pools

- Monitoring GridLink JDBC Resources

5.4.2 Multi Data Sources

A **multi data source** is an abstraction around a group of data sources that provides load balancing or failover processing. Multi data sources support load balancing for both XA and non-XA data sources.

A multi data source provides an ordered list of data sources to fulfill connection requests. Normally, every connection request to this kind of multi data source is served by the first data source in the list. If a database connection test fails and the connection can't be replaced, or if the data source is suspended, a connection is sought sequentially from the next data source on the list."

Multi data sources are bound to the JNDI tree or local application context just like regular data sources. 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.

See Also: To configure Multi Data Sources with Oracle RAC, see "Using Multi Data Sources with Oracle RAC" in *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

5.5 Configuring Active GridLink Data Sources with Oracle RAC

How you configure an Active GridLink data source depends on:

- The Oracle component that you are working with
- The domain you are creating

This topic describes how to configure component data sources as Active GridLink data sources for a RAC database during domain creation.

This section includes the following topics:

- Section 5.5.1, "Requirements to Configure Component Data Sources as Active Gridlink Data Sources"
- Section 5.5.2, "Configuring Component Data Sources as Active GridLink Data Sources"
- Section 5.5.3, "Using SCAN Addresses for Hosts and Ports"

See Also: To create and configure Active GridLink data sources, see Using Active GridLink Data Sources in *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

5.5.1 Requirements to Configure Component Data Sources as Active Gridlink Data Sources

Your system must meet the following requirements before you configure component data sources as Active GridLink data sources to use with an Oracle RAC database:

- You use Oracle RAC database version 11g Release 2 or later.

- You have run RCU to create component schemas.
- You are using the Configuration Wizard to create or configure a domain and have arrived at the JDBC Component Schema screen where you select **Component Datasources**.

5.5.2 Configuring Component Data Sources as Active GridLink Data Sources

To configure component data sources as Active GridLink data sources:

1. In the JDBC Component Schema screen, select one or more component schemas to configure GridLink data sources for.
2. Select **Convert to GridLink** then select **Next**.
3. In the GridLink Oracle RAC Component Schema screen, select one of the GridLink JDBC drivers.
4. In the **Service Name** field, enter the service name of the database using lowercase characters. For example, `mydb.example.com`.
5. In the **Schema Owner** field, enter the name of the database schema owner for the corresponding component.
6. In the **Schema Password** field, enter the password for the database schema owner.
7. In the **Service Listener**, **Port**, and **Protocol** field, enter the SCAN address and port for the RAC database being used. The protocol for Ethernet is TCP; for Infiniband it is SDP. Click **Add** to enter multiple listener addresses.

   You can identify the SCAN address by querying the appropriate parameter in the database using the TCP protocol:

   ```
   show parameter remote_listener

   NAME TYPE VALUE
   --------------------------------------------------------------------
   remote_listener string db-scan.example.com:1521
   ```

   You can also identify the SCAN address by using the `srvctl config scan` command. Use the command `srvctl config scan_listener` to identify the SCAN listener port.
8. Select **Enable FAN** to receive and process FAN events. Enter one or more ONS daemon listen addresses and port information. Select **Add** to enter more entries.

   Note: Verify that the ONS daemon listen address(es) that you enter is valid. The domain creation process does not validate the address.

   For the ONS host address, use the Oracle RAC database SCAN address and the ONS remote port as reported by the database:

   ```
   srvctl config nodeapps -s

   ONS exists: Local port 6100, remote port 6200, EM port 2016
   ```
9. Select **Enable SSL** for SSL communication with ONS. Enter the Wallet File, which has SSL certificates, and the Wallet Password.
10. Select **Next**. Verify that all connections are successful.

See Also: For more information, see:

- "JDBC Component Schema" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard* for information about the JDBC Component Schema screen.
- "GridLink Oracle RAC Component Schema" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard* for information about configuring component schemas.
- "Using Active GridLink Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server* for information on GridLink RAC data sources.

5.5.3 Using SCAN Addresses for Hosts and Ports

Oracle recommends that you use Oracle Single Client Access Name (SCAN) addresses to specify the host and port for the TNS listener and ONS listener in the WebLogic console. You do not need to update an Active 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 *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

5.6 Configuring Multi Data Sources

You configure multi data sources using:

- Oracle Fusion Middleware Configuration Wizard (during WebLogic Server domain creation)
- Oracle WebLogic Server Administration Console
- WLST Commands

This section includes the following topics:

- Section 5.6.1, "Configuring Multi Data Sources with Oracle RAC"
- Section 5.6.2, "Configuring Multi Data Sources for MDS Repositories"

5.6.1 Configuring Multi Data Sources with Oracle RAC

This section includes the following topics:

- Section 5.6.1.1, "Requirements to Configure Multi Data Sources with Oracle RAC"
- Section 5.6.1.2, "Configuring Component Data Sources as Multi Data Sources"
- Section 5.6.1.4, "Modifying or Creating Multi Data Sources After Initial Configuration"
- Section 5.6.1.6, "Configuring Schemas for Transactional Recovery Privileges"

5.6.1.1 Requirements to Configure Multi Data Sources with Oracle RAC

Verify that your system meets the following requirements before you configure component data sources as multi data sources to use with an Oracle RAC database:

- You are using an Oracle RAC database.
- You have run RCU to create component schemas.

- You are using the Configuration Wizard to create or configure a domain and have arrived at the JDBC Component Schema Screen where you select Component Schemas. Before you arrive at the JDBC Component Schema screen, you must select the option **Manual Configuration** in the Database Configuration Type screen."

5.6.1.2 Configuring Component Data Sources as Multi Data Sources

To configure component data sources as multi data sources:

1. In the Component Datasources screen, select one or more component schemas to configure RAC Multiple data sources for.
2. Select **Convert to RAC multi data source** then select **Next**.
3. In the Oracle RAC Multi Data Source Component Schema screen, the JDBC driver **Oracle's Driver (Thin) for RAC Service-Instance connections; Versions:10 and later**.
4. In the **Service Name** field, enter the database service name enter in lowercase, for example, `mydb.example.com`.
5. In the **Schema Owner** field, enter the username of the database schema owner for the corresponding component.
6. In the **Schema Password** field, enter the password for the database schema owner.
7. In the **Host Name**, **Instance Name**, and **Port** field, enter the RAC node hostname, database instance name, and port. Click **Add** to enter multiple listener addresses.
8. Click **Next**.Verify that all connections are successful.

5.6.1.3 About Adding Multi Data Sources For RAC Databases

Multi data sources have constituent data sources for each RAC instance that provides a database service. Oracle recommends adding an additional data source to the multi data source on the Fusion Middleware tier when you add an additional instance to the RAC back end.

When you migrate a database from a non-RAC to a RAC database, you must create an equivalent, new multi data source for each affected data source. Multi data sources that you create must have constituent data sources for each RAC instance. Data source property values must be identical to the original single instance data source for properties in Section 5.6.1. For example, if a 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.

5.6.1.4 Modifying or Creating Multi Data Sources After Initial Configuration

For multi data sources that you create manually or modify after initial configuration, Oracle strongly recommends specific XA and non-XA data source property values for optimal high availability. Make changes only after careful consideration and testing if your environment requires that you do so.

The following tables describe XA and non-XA data source property values that Oracle recommends.

Table 5–2 Recommended Multi Data Source Configuration

Property Name	Recommended 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 5–3 XA Data Source Configuration

Property Name	Recommended 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

5.6.1.5 Troubleshooting Warning Messages (Increasing Transaction Timeout for XA Data Sources)

If WARNING messages in server logs have the following exception, you may need to increase the XA timeout value in your setup.

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

To increase the transaction timeout for the XA Data Sources setting, use the Administration Console:

1. Access the data source configuration.
2. Select the **Transaction** tab.
3. Set the 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.

Table 5–4 Non-XA Data Source Configuration

Property Name	Recommended 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

5.6.1.6 Configuring Schemas for Transactional Recovery Privileges

You need the appropriate database privileges to enable WebLogic Server transaction manager to:

- Query for transaction state information
- Issue the appropriate commands, such as commit and rollback, during recovery of in-flight transactions after a WebLogic Server container failure.

To configure 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`.

5.6.2 Configuring Multi Data Sources for MDS Repositories

You can configure applications that use an MDS database-based repository for high availability Oracle database access. With this configuration, failure detection, recovery, and retry by MDS (and by the WebLogic infrastructure) protect application read-only MDS operations from Oracle RAC database planned and unplanned downtimes.

The Fusion Middleware Control navigation tree exposes multi data sources as MDS repositories. You can select these multi data sources when you customize the application deployment and use them 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*.

- Registering an MDS multi data source

  In addition to steps in Section 5.6.1, "Configuring Multi Data Sources with Oracle RAC," note the following:

  - You must configure child data sources that comprise a multi data source used for an MDS repository as non-XA data sources.
  - A multi data source's name must have the prefix `mds-`. This ensures that the multi data source is recognized as an MDS repository that can be used for MDS management functionality.

  Note: When you add a MDS data source as a child of a multi data source, this data source is no longer exposed as an MDS repository. 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 things to consider when you convert a data source to a multi data source to verify application configuration:

  - 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.

6 Scaling Out a Topology (Machine Scale Out)

This section describes the steps to scale out a topology (machine scale-out) for all Fusion Middleware products that are a part of a WebLogic Server domain. To enable high availability, it is important to provide failover capabilities to another host computer. When you do so, your environment can continue to serve consumers of your deployed applications if one computer goes down.

This section includes the following topics:

6.1 About Machine Scale Out

Scalability is the ability of a piece of hardware or software or a network to "expand" or "shrink" to meet future needs and circumstances. A scalable system is one that can handle increasing numbers of requests without adversely affecting response time and throughput.

Machine scale-out is the process of moving a server, one of many on one machine, to another machine for high availability. Machine scale out is different from Managed Server **scale up**, which is the process of adding a new Managed Server to a machine that already has one or more Managed Servers running on it. See "Scaling Up Your Environment" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

6.2 Roadmap for Scaling Out Your Topology

The following table describes the typical steps you must take to scale out a topology.

If you already have a standard installation topology (as *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure* and product installation guides such as *Oracle Fusion Middleware Installing and Configuring Oracle SOA Suite and Business Process Management* describe), you do *not* need to repeat the Section 6.5, Section 6.6, and Section 6.7 steps. The following table provides start-to-finish steps if you did not complete standard installation topology steps.

Table 6–1 Roadmap for Scaling Out Your Topology

Task	Description	More Information
Product is ready for scale out	Product is installed, configured, and a cluster of Managed Servers is available; your product is in a standard installation topology	Section 6.4, "About Scale Out Prerequisites"
Verify that you meet resource requirements	You must verify that your environment meets certain requirements	Section 6.5, "Resource Requirements"
Create a new machine and assign servers to it	Use the Administration Console to create a new machine and add Managed Servers to it.	Section 6.6, "Creating a New Machine"
Create a new JMS server and target it	Create a new JMS server and target it to the new Managed Server	Section 6.7, "Configuring WLS JMS After Machine Scale Up or Scale Out"
Run the pack command	Pack up the domain directory	Section 6.8, "Packing the Domain on APPHOST1"
Prepare the new machine	Install the same software that you installed on the first machine	Section 6.9, "Preparing the New Machine"
Run the unpack command	Create a Managed Server template.	Section 6.10, "Running Unpack to Transfer the Template"
Start the server	Starts the Managed Server on the new machine	Section 6.12, "Starting Managed Servers"
Verify the topology	Test the new setup	Section 6.13, "Verifying Machine Scale Out"
Set the cluster messaging mode to Multicast	Modifies messaging mode from Unicast to Multicast (preferred for multi-server domains)	Section 6.14, "Configuring Multicast Messaging for Clusters"

Note: In this section, *APPHOST* refers to a physical host computer. *Machine* refers to the WebLogic Server machine definition describing that host. See "Understanding the Oracle Fusion Middleware Infrastructure Standard Installation Topology" in *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

6.3 Optional Scale Out Procedure

By following the standard installation topology, you have multiple Managed Servers assigned to a single host computer.

This approach is the most flexible way to create and scale out a domain topology so that it can meet changing requirements. It allows you to 1) create and validate a single-host domain, which is targeted to a single machine on a single host computer, and then 2) "retarget" the Managed Servers to additional machines, when additional computing resources are required. Also, it facilitates troubleshooting; you can validate the basic domain then scale up and scale out (and troubleshoot) at a later time.

However, if you know ahead of time what your target topology is, you can create additional machines during domain creation then just run pack and unpack steps.

If you assigned Managed Servers to target machines during installation or through an online administrative operation, skip Section 6.6, "Creating a New Machine" as you go through the roadmap (Section 6.2, "Roadmap for Scaling Out Your Topology").

See product installation guides, such as *Installing and Configuring the Oracle Fusion Middleware Infrastructure*, for more on machine mapping.

6.4 About Scale Out Prerequisites

Before you start the scale out process, you must have a **standard installation topology** set up for your product. The standard installation topology is the starting point for scale out. If you followed the steps in your product installation guide, you should have a standard installation topology. For an example, see the standard installation topology that "Understanding the Oracle Fusion Middleware Infrastructure Standard Installation Topology" describes in *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure.*

See Also: For more on the standard installation topology, see your product's installation guide or "About the Standard Installation Topology" in *Planning an Installation of Oracle Fusion Middleware*.

Note: Application tier products in this release do not support dynamic clusters. **Dynamic clusters** consist of server instances that can dynamically scale up to meet your application resource needs. A dynamic cluster uses a single server template to define configuration for a specified number of generated (dynamic) server instances.

6.5 Resource Requirements

Before you scale out the topology, verify that your environment meets these requirements:

- At least one machine runs multiple Managed Servers configured with a product. This is the result of following your product installation guide or administration guide to add additional servers.
- A host computer in addition to your starting host computer.
- Each host computer can access the Oracle home that contains the product binaries by one of the following means:
  - Shared disk with binaries from the original installation
  - Dedicated disk with a new installation (matches the original installation)
  - Dedicated disk with a clone of the binaries from the original installation

  See Chapter 4, "Using Shared Storage" for more information.
- Sufficient storage available for the domain directory.
- Access to the same Oracle or third-party database used for the original installation.
- A shared disk for JMS and transaction log files (required when using a file persistence profile).

6.6 Creating a New Machine

A **machine** is the logical representation of the computer that hosts one or more WebLogic Server instances (servers). In a WebLogic domain, the machine definitions identify physical units of hardware and are associated with the servers that they host.

Follow these steps to create a new machine:

- Section 6.6.1, "Shutting Down the Managed Server"
- Section 6.6.2, "Creating a New Machine (Using the Administration Console)"
- Section 6.6.3, "Assigning Managed Servers to a New Machine"

6.6.1 Shutting Down the Managed Server

The Managed Server must be shut down before moving to a new machine. If it is up and running, see the topic "Shut down a server instance" in the *Administration Console Online Help* to shut down the Managed Server that the new machine will host.

6.6.2 Creating a New Machine (Using the Administration Console)

This topic describes how to create a new machine using the Administration Console. To use WLST to create a new machine, see "Creating a New Machine for Certain Components" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

Note: The machine you create in this procedure must have a listen address on a specific network interface, not just a local host.

To create a new machine in the domain:

1. Start the domain Administration Server if it is not running. Go to the *DOMAIN_HOME*/bin directory and run:

   ```
   ./startWeblogic.sh
   ```

2. When the Administration Server is running, access the Administration Console. Open a web browser and enter the URL:

   ```
   http://hostname:port/console
   ```

   On the Welcome screen, log in.

3. In the Administration Console Change Center, click **Lock & Edit**.

   > **Note:** For production domains, Oracle recommends creating the domain in **Production** mode, which enables Change Center. If Production mode is *not* enabled, Change Center steps are not required.

4. Under Domain Structure, expand Environment then click **Machines**.
5. Above the Machines table (above Summary), click **New**.
6. In the Create a New Machine screen, enter a name for the machine, such as **machine_2**. Select the **Machine OS** using the drop-down list then click **Next**.
7. On the next screen, for **Type:**, use the drop-down list to select **Plain**.

   For the Node Manager **Listen Address**, enter the IP address or host name of the host computer that will represent this machine. Both machines appear in the Machines table.

8. Click **Finish**.
9. In the Change Center, click **Activate Changes**.

   The message **"All changes have been activated. No restarts are necessary.**" appears. This message indicates that you have a new machine.

6.6.3 Assigning Managed Servers to a New Machine

To add Managed Servers to a newly-created machine, use the Administration Console:

1. In the Change Center, click **Lock & Edit**.
2. In the Machines table, select the checkbox for **machine_1**.
3. Click the machine name (represented as a hyperlink).
4. Under the Settings for **machine_1**, click the Configuration tab then the Servers subtab.
5. Above the Servers table, click **Add**.
6. On the Add a Server to Machine screen, click **Create a new server and associate it with this machine**. Click **Next** then enter the **Server Name** and the **Server Listen Port** in the fields (required).
7. Under Domain Structure, click **Machines**.

   In the Machines table, click the machine **machine_2**.

8. Under the Settings for **machine_2**, click the Configuration tab and then the Servers tab. Above the Servers tab, click **Add**.

   On the **Add a Server to Machine** screen, select the button **Select an existing server, and associate it with this machine**.

   Use the Select a server drop-down list to choose **server_2** then select **Finish**.

   The message **Server created successfully** appears.

9. Verify that all Managed Servers have the correct Server Listen Address. Under Domain Structure, click **Servers**. In the Servers table, click the name of the Managed Server. Select the Configuration tab. Verify/set the Listen Address to the IP address of the machine it is associated with. Click **Save**.
10. To complete the changes, go back to the Change Center. Click **Activate Changes**. The message **All changes have been activated. No restarts are necessary.** appears.

    To see a summary of Managed Server assignments, under Domain Structure, under Environment, click **Servers**. The Servers table shows all servers in the domain and their machine assignments.

6.7 Configuring WLS JMS After Machine Scale Up or Scale Out

When you add a new Managed Server to a cluster, you must manually recreate all JMS system resources on the new Managed Server. The new JMS system resources are "cloned" from an existing Managed Server in the cluster. The new JMS system resources must have unique names in the domain. When you create a domain, the Configuration Wizard creates JMS system resource names that follow a pattern. For ease of use and manageability, Oracle recommends that you follow the same naming pattern.

To configure JMS resources on a new Managed Server `server_`*n*:

1. In the **Domain Structure** tree, select **Services** then select **Messaging**. Select **JMS Servers** to open the JMS Servers table.
2. In the Name column, identify all JMS servers that target one of the existing Managed Servers in the cluster, for example, `server_1`.

   The JMS server name format is *ResourceName*_**auto**_*number*.

   - *ResourceName* is the resource's identifying name
   - *number* identifies the JMS server; servers with the suffix 1 or 2 were created when the domain was created.
3. Note the name of the JMS server and its persistent store. For example, `UMSJMSServer_auto_1` and `UMSJMSFileStore_auto_1`.
4. Click **New** to create a new JMS server.
   a. Name the JMS server for `server_`*n* using the same format as `server_1`. For example, `UMSJMSServer_auto_`*n*.
   b. For the **Persistent Store**, click **Create a New Store**.
   c. For **Type**, select the same persistence profile used that the JMS Server uses on `server_1`. Click **Next**.
   d. Enter a persistent store in the **Name** field. Use the same format as the `server_1` persistent store. For example, `UMSJMSFileStore_auto_`*n*.
   e. In the **Target** field, select the migratable target for migratable target `server_`*n* `(migratable)` from the drop down list.
   f. In the **Directory** field, use the same name as the persistent store. Click **OK** to create the persistent store.
   g. In the Create a New JMS Server screen, select the new persistent store and click **Next**.
   h. For JMS Server Target, select the migratable target for `server_`*n* (migratable) from the drop down list.

i. Click **Finish** to create the JMS server.

5. Update the Subdeployment targets for the corresponding JMS Modules to include the new JMS server:

   a. In the Administration Console's **Domain Structure** tree, expand the **Services** node, expand the **Messaging** node, and select **JMS Modules** to open the JMS Modules table.

   b. Identify the JMS system module that corresponds to the JMS server; its name has a common root. For example for JMS server `UMSJMSServer_auto_1`, the JMS module name is `UMSJMSSystemResource`. Click on the JMS module (a hyperlink) in the Name column to open the Module Settings page.

   c. Open the Subdeployments tab and click on the subdeployment for the JMS module in the Name column.

   d. Select the new JMS Server `server_`*n*. Click **Save**.

   e. Go back to the module Settings page. Select the **Targets** tab. Verify that **All servers in the cluster** is enabled.

   f. Click **Save** if you changed anything.

You now have a new Managed Server that has configured JMS resources in the cluster.

6.8 Packing the Domain on APPHOST1

You create a Managed Server template by running the `pack` command on the WebLogic domain.

Note: The Administration Server should be running on APPHOST1 when you go through the pack and unpack steps.

Run the `pack` command on **APPHOST1** to create a template pack. You unpack the template file on **APPHOST2** later in the scale out process; Section 6.10, "Running Unpack to Transfer the Template" describes the unpack procedure.

For example:

```
ORACLE_HOME/oracle_common/common/bin/pack.sh \
     -domain=DOMAIN_HOME \
     -template=dir/domain_name.jar \
     -managed=true \
     -template_name="DOMAIN"
```

In the preceding example:

- Replace *DOMAIN_HOME* with the full path to the domain home directory.
- Replace *dir* with the full path to a well-known directory where you will create the new template file.
- Replace *domain_name* in the JAR file name with the domain name. This is the name of the template file that you create with the `pack` command. For example: `mydomain.jar`.

See Also: See "pack and unpack Command Reference" in *Creating WebLogic Domains and Domain Templates* for more information about creating a Managed Server template.

6.9 Preparing the New Machine

To prepare the new machine, **machine_2**, verify that **APPHOST2** has access to the shared disk where the Oracle home is installed or install the same software that you installed on **machine_1**.

For example, if you are scaling out an Oracle Fusion Middleware Infrastructure domain, verify that **APPHOST2** can access the Infrastructure Oracle home.

Note: If you use shared storage, you can reuse the same installation location.

Note: If you are performing a new installation or reusing the binaries by means of shared storage, the path location of the new files must match the original machine's path location exactly.

6.10 Running Unpack to Transfer the Template

To unpack the template and transfer the *domain_name*.jar file from **APPHOST1** to **APPHOST2**, run the unpack command:

```
ORACLE_HOME/oracle_common/common/bin/unpack.sh \
    -domain=user_projects/domains/base_domain2 \
    -template=/tmp/domain_name.jar \
    -app_dir=user_projects/applications/base_domain2
```

6.11 Starting the Node Manager

To start Node Manager, run the following:

```
DOMAIN_HOME/bin/startNodeManager.sh &
```

When you use machine scoped Node Manager, see "Using Node Manager" in *Administering Node Manager for Oracle WebLogic Server* for more information on Node Manager start options.

6.12 Starting Managed Servers

To use the Administration Console to start the Managed Servers:

1. In the left pane of the Console, expand **Environment** and select **Servers**.
2. In the **Servers** table, click the name of the Managed Server that you moved to the new machine. Select the **Control** tab.
3. Select the check box next to the name of the server(s) you want to start and click **Start** to start the server(s).

See Also: To use WLST commands or Fusion Middleware Control to start Managed Servers, see "Starting and Stopping Managed Servers" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

6.13 Verifying Machine Scale Out

To determine if the machine scale out succeeded, verify that the server status is **RUNNING** after you use the Administration Console to start it.

6.14 Configuring Multicast Messaging for Clusters

This section includes the following topics:

- Section 6.14.1, "About Multicast and Unicast Messaging for Clusters"
- Section 6.14.2, "Requirements to Configure Multicast Messaging"
- Section 6.14.3, "Configuring Multicast Messaging"

6.14.1 About Multicast and Unicast Messaging for Clusters

Clusters use messaging to broadcast the availability of services and heartbeats that indicate continued availability. You configure clusters to use Unicast or Multicast messaging.

- **Multicast** is a simple broadcast technology that enables multiple applications to subscribe to an IP address and port number, and listen for messages. Multicast set up is more complex than Unicast because it needs hardware configuration and support.
- **Unicast** provides TCP-based, reliable, one-to-many communication for cluster messaging. It is easier to set up than multicast.

When you create clusters in the Configuration Wizard, Unicast is the default cluster messaging mode. When the number of Managed Servers in a domain increases, Multicast messaging is preferable.

6.14.2 Requirements to Configure Multicast Messaging

Requirements to configuring Multicast messaging include:

- A configured domain with at least one cluster.
- A hardware configuration set up to support Multicast in your network.
- The IP address and port numbers for Multicast communications in the cluster. A multicast address is an IP address between 224.0.0.0 and 239.255.255.255. (Weblogic uses default value of 239.192.0.0).
- You have run the MulticastTest utility to verify that your environment can support multicast. The utility debugs multicast problems. It sends out multicast packets and returns data about how effectively multicast works in your network.

See Also: See "Communications In a Cluster" in *Administering Clusters for Oracle WebLogic Server* for details on Multicast messaging and cluster configuration.

6.14.3 Configuring Multicast Messaging

To configure Multicast Messaging:

1. Log in to the Administration Console.
2. Shut down all servers that you want to use for Multicast messaging.

3. In the Domain Structure pane on the left, expand **Environment** and click **Clusters**.
4. Select the cluster name that you want to enable Multicast for.
5. Click the **Configuration** tab then the **Messaging** tab.
6. For Messaging Mode, select **Multicast**. Enter the Multicast Address then the Multicast Port.

   The **Multicast Address** must be unique for each cluster. See "Cluster Multicast Address and Port" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* for guidelines on Multicast addresses.
7. Click **Advanced** to configure these parameters:

Table 6–2 Multicast Advanced Parameters

Parameter	Description
Multicast Send Delay	Amount of time (between 0 and 250) in milliseconds to delay sending message fragments over multicast to avoid OS-level buffer overflow.
Multicast TTL	Number of network hops (between 1 and 255) that a cluster multicast message can travel. If your cluster spans multiple subnets in a WAN, this parameter value must be high enough to ensure that routers don't discard multicast packets before they reach their final destination. This parameter sets the number of network hops a multicast message makes before a packet can be discarded.
Multicast Buffer Size	Multicast socket send/receive buffer size (at least 64 kilobytes).
Idle Periods Until Timeout	Maximum number of periods a cluster member waits before timing out a cluster member.
Enable Data Encryption	Enables multicast data encryption. Only multicast data is encrypted; Multicast header information is not encrypted.

8. Click **Save**.
9. Restart all servers in the cluster.

The cluster can now use Multicast messaging. When you select **Clusters** in the Domain Structure pane in the Administration Console, **Multicast** appears for **Cluster Messaging Mode**.

7 JMS and JTA High Availability

This section describes Java Message Service (JMS) and Java Transaction API (JTA) high availability.

This section includes the following topics:

- Section 7.1, "About JMS and JTA Services for High Availability"
- Section 7.2, "About Migratable Targets for JMS and JTA Services"
- Section 7.3, "Configuring Migratable Targets for JMS and JTA High Availability"
- Section 7.4, "User-Preferred Servers and Candidate Servers"
- Section 7.5, "Considerations for Using File Persistence (WebLogic JMS)"
- Section 7.6, "Using File Stores on NFS"
- Section 7.7, "Configuring WLS JMS with a Database Persistent Store"
- Section 7.8, "Configuring Database Stores to Persist Transaction Logs"

See Also: For more on working with JMS or JTA, see:

- "Configuring WebLogic JMS Clustering" in *Oracle Fusion Middleware Administering JMS Resources for Oracle WebLogic Server*
- "Interoperating with Oracle AQ JMS" in *Oracle Fusion Middleware Administering JMS Resources for Oracle WebLogic Server*
- "Configuring JTA" in *Developing JTA Applications for Oracle WebLogic Server*.
- "Domains:Configuration:JTA" and "Cluster:Configuration:JTA" in *Administration Console Online Help*

7.1 About JMS and JTA Services for High Availability

Java Message Service (JMS) is an application program interface (API) that supports the formal communication known as *messaging* between computers in a network.

Java Transaction API (JTA) specifies standard Java interfaces between a transaction manager and parties involved in a distributed transaction system: the resource manager, the application server, and the transactional applications.

In WebLogic JMS, a message is available only if its host JMS server for the destination is running. If a message is in a central persistent store, the only JMS server that can access the message is the server that originally stored the message. WebLogic has features to restart and/or migrate a JMS server automatically after failures. It also has features for clustering (distributing) a destination across multiple JMS servers within the same cluster.

You automatically restart and / or migrate (fail over) JMS Servers using either Whole Server Migration or Automatic Service Migration.

See Also:: For more information on Whole Server Migration, see Chapter 3, "Whole Server Migration."

7.2 About Migratable Targets for JMS and JTA Services

To configure JMS and JTA services for high availability, you deploy them to a **migratable target**, a special target that can migrate from one server in a cluster to another. A migratable target groups migratable services that should move together; when a migratable target migrates, all services that it hosts also migrate.

A migratable target specifies a set of servers that can host a target. Only one server can host a migratable target at any one time. It can also specify:

- A user-preferred host for services
- An ordered list of backup servers if a preferred server fails

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

You can migrate pinned migratable services manually from one server to another in the cluster if 1) a server fails, or 2) as part of scheduled maintenance. If you *do not* configure a migratable target in the cluster, migratable services can migrate to any server in the cluster.

See Section 7.3, "Configuring Migratable Targets for JMS and JTA High Availability" to configure migratable targets.

See Also: For more on administering JMS, see *Oracle Fusion Middleware Administering JMS Resources for Oracle WebLogic Server*, which includes:

- "High Availability Best Practices"
- "Interoperating with Oracle AQ JMS"

7.3 Configuring Migratable Targets for JMS and JTA High Availability

To configure migratable targets, see these topics in *Administration Console Online Help*:

- "Configure migratable targets for JMS-related services"
- "Configure migratable targets for the JTA Transaction Recovery Service"

7.4 User-Preferred Servers and Candidate Servers

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

You can create separate migratable targets for JMS services so that you can 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 CCSs for both JTA and JMS, to ensure that services stay co-located on the same server in the cluster.

7.5 Considerations for Using File Persistence (WebLogic JMS)

You can configure the file system to store JMS messages and JTA logs. For high availability, you must use a shared file system. See Chapter 4, "Using Shared Storage" for what to consider when you use a shared file system.

See Also: For more information, see "WebLogic JMS Architecture and Environment" in *Administering JMS Resources for Oracle WebLogic Server.*

7.6 Using File Stores on NFS

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

This section includes the following topics:

- Section 7.6.1, "Verifying Server Restart Behavior"
- Section 7.6.2, "Disabling File Locking for the Default File Store"
- Section 7.6.3, "Disabling File Locking for a Custom File Store"
- Section 7.6.4, "Disabling File Locking for a JMS Paging File Store"
- Section 7.6.5, "Disabling File Locking for Diagnostics File Stores"

7.6.1 Verifying Server Restart Behavior

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 mode 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 doesn't restart after abrupt machine failure, review server log files to verify whether or not it is due to an I/O exception similar to the following:

```
<MMM dd, yyyy hh:mm:ss a z> <Error> <Store> <BEA-280061> <The persistent
store "_WLS_server_1" could not be deployed:
weblogic.store.PersistentStoreException: java.io.IOException:
[Store:280021]There was an error while opening the file store file
"_WLS_SERVER_1000000.DAT"
weblogic.store.PersistentStoreException: java.io.IOException:
[Store:280021]There was an error while opening the file store file
"_WLS_SERVER_1000000.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 doesn't release locks on 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 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.

When you use a file store, always configure the database-based leasing option for server migration. This option enforces additional locking mechanisms using database tables and prevents automated restart of more than one instance of a particular Managed Server.

7.6.2 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.
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>
```

7.6.3 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 to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Services** node. 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, you must restart the server for changes to take effect.

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

```
<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>
```

7.6.4 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 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 a JMS server to modify.
4. On the **Configuration > General** tab for the JMS Server, scroll down. 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 changes to take effect.

The `config.xml` file entry looks like the following example:

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

7.6.5 Disabling File Locking for Diagnostics File Stores

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

1. If necessary, click **Lock & Edit** in the Change Center to get an Edit lock for the domain.
2. In the **Domain Structure** tree, expand the **Diagnostics** node. 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 looks 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>
```

See Also: For more information, see "Configure JMS Servers and Persistent Stores" in *Oracle Fusion Middleware Administering JMS Resources for Oracle WebLogic Server*.

7.7 Configuring WLS JMS with a Database Persistent Store

This topic describes how to change the WLS JMS configuration from a file-based persistent store (default configuration) to a database persistent store.

This section includes the following topics:

- Section 7.7.1, "About the Persistent Store"
- Section 7.7.2, "Prerequisites for Configuring WLS JMS with a Database Persistent Store"
- Section 7.7.3, "Switching WLS JMS File-Based Persistent Stores to Database Persistent Store"

7.7.1 About the Persistent Store

The **persistent store** is a built-in storage solution for WebLogic Server subsystems and services that require persistence. For example, it can store persistent JMS messages. The persistent store supports persistence to a file-based store or to a JDBC-accessible store in a database.

For information on the persistent store, see "The WebLogic Persistent Store" in Administering the WebLogic Server Persistent Store.

For information on typical tasks to monitor, control, and configure WebLogic messaging components, see "WebLogic Server Messaging" in *Administering Oracle WebLogic Server with Fusion Middleware Control*.

7.7.2 Prerequisites for Configuring WLS JMS with a Database Persistent Store

To configure WLS JMS with database persistent stores, verify that your setup meets these requirements:

- An Oracle Fusion Middleware installation with at least one cluster and one or more JMS servers
- JMS servers that use file persistent stores, the default configuration.

7.7.3 Switching WLS JMS File-Based Persistent Stores to Database Persistent Store

You must follow the steps in this procedure for each JMS server that you must configure to use database persistent stores.

1. Create a JDBC store. See "Using a JDBC Store" in *Oracle Fusion Middleware Administering Server Environments for Oracle WebLogic Server*.

   Note: You must specify a prefix to uniquely name the database table for the JDBC store.

2. Associate the JDBC store with the JMS server:
   a. In the Weblogic Server Administration Console, go to **Services**->**Messaging**->**JMS Servers**.
   b. Verify that there are no pending messages in this server. In the Control tab, stop production and insertion of messages for all destinations and wait for any remaining messages to drain.

c. Select the General Configuration tab. Under Persistent Store, select the new JDBC store then click **Save**.

The JMS server starts using the database persistent store.

7.8 Configuring Database Stores to Persist Transaction Logs

This section includes the following topics:

- Section 7.8.1, "Requirements for Configuring JDBC TLOG Stores"
- Section 7.8.2, "Configuring JDBC TLOG Stores"

7.8.1 Requirements for Configuring JDBC TLOG Stores

You must have a standard Oracle Fusion Middleware installation before you configure a JDBC Transaction Logs (TLOG) Store.

Post installation, the TLOG Store is configured in the file system. In some instances, Oracle recommends that you configure TLOGs to store in the database. To configure JDBC TLOGs to stored to a database store, see "Using a JDBC TLOG Store" in *Administering the WebLogic Persistent Store*.

7.8.2 Configuring JDBC TLOG Stores

When you configure JDBC TLOG Stores:

- You must repeat the procedure for each Managed Server in the cluster.
- Use the Managed Server name as a prefix to create a unique TLOG store name for each Managed Server.
- Verify that the data source that you created for the persistent store targets the cluster for a high availability setup.

When you finish the configuration, TLOGs are directed to the configured database-based persistent store.

Note: When you add a new Managed Server to a cluster by scaling up or scaling out, you must also create the corresponding JDBC TLOG Store for the new Managed Server.

8 Administration Server High Availability

This section describes high availability aspects of the Administration Server.

This section includes the following topics:

For more information on the Administration Server and Node Manager, see the following topics:

Table 8–1 Administration Server and Node Manager Topics

For information on...	See this topic...
Starting and Stopping the Administration Server	"Starting and Stopping Administration Server" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*
Configuring virtual hosting	"Configuring Virtual Hosting" in *Administering Server Environments for Oracle WebLogic Server*
Using Node Manager	In *Administering Node Manager for Oracle WebLogic Server*: ■ "Node Manager and System Crash Recovery" ■ "How Node Manager Works in a WLS Environment" ■ "Node Manager Configuration and Log Files"

8.1 Role of the Administration Server

The Administration Server is the central control entity for configuring the entire domain, and manage and monitor all domain resources. It maintains the domain's configuration documents and distributes changes in these documents to Managed Servers. Each domain requires one server that acts as the Administration Server.

8.1.1 Administration Server Failure and Restart

Administration Server failure does not affect how Managed Servers in a domain operate. If an Administration Server fails due to a hardware or software failure on its host computer, other server instances on the same computer may also be affected.

For more information on Administration Server failure, see "Impact of Managed Server Failure" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

To restart an Administration Server, see *Oracle Fusion Middleware Using Clusters for Oracle Server*.

8.1.2 Shared Storage and Administration Server High Availability

With shared storage, a backup host can access the same artifacts (Oracle binaries, configuration files, domain directory, and data) that an active host can. You configure this access by placing artifacts in storage that all hosts in the highly available Administration Server configuration can access. Shared storage is a network-attached storage (NAS) or storage area network (SAN) device. See Chapter 4, "Using Shared Storage".

8.2 Role of Node Manager

For each WebLogic Server domain you create, a *per domain* Node Manager configuration is created by default. This Node Manager comes complete with security credentials, a properties file, domain registration, and start scripts

You can configure the scope of Node Manager:

- **per domain** Node Manager is associated with a *domain* to control all servers for the domain on a machine. Default configuration.
- **per host** Node Manager is associated with a specific *machine*, not a domain. One Node Manager process can control server instances in any domain, as long as the server instances reside on the same machine as the Node Manager process. A per host Node Manager must run on each computer that hosts WebLogic Server instances that you want to control with Node Manager, whether the WebLogic Server instances are an Administration Server or Managed Server(s).

Note: Oracle recommends that you run Node Manager as an operating system service so that it restarts automatically if its host machine restarts.

Node Manager failure doesn't affect any servers running on the machine.

For more information, see "What is Node Manager?" in *Oracle Fusion Middleware Understanding Oracle Fusion Middleware Concepts*.

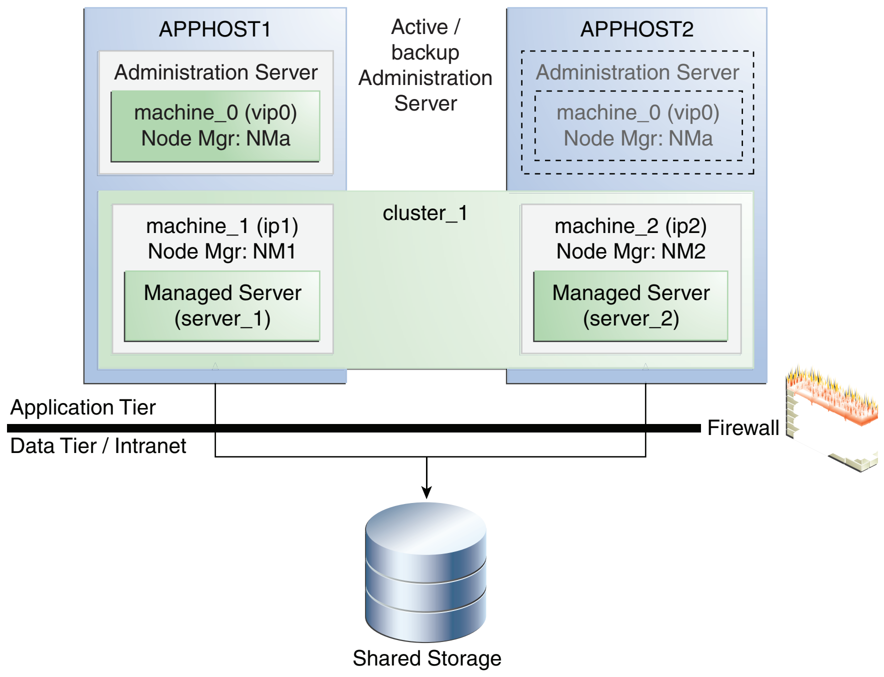
8.3 Administration Server High Availability Topology

Administration Server can be active on one host at any one time. To set up high availability, you configure the Administration Server on a virtual host so that if the machine that it runs on fails, you can fail it over to another host in the domain. Administration Server is configured to use a virtual IP to overlap the backup hosts. You configure Administration Server to listen on this virtual IP. The benefit of using a virtual host and virtual IP is that you don't need to add a third machine; if failover occurs, you can map the virtual host to a surviving host in the domain by "moving" the virtual IP.

The two hosts share a virtual hostname and a virtual IP. However, only the active host can use this virtual IP at any one time. If the active host fails, the backup host becomes active and you must move (manually) the virtual IP to the new active host. The new active host then services all requests through the virtual IP. (You configure a high availability deployment to listen on this virtual IP.)

The following figure shows a highly available Administration Server. In this topology, the Administration Server runs on a virtual host, `APPHOST0`. `APPHOST0` overlaps to `APPHOST1` or `APPHOST2` by means of a virtual IP. At first, `APPHOST0` maps to `APPHOST1`, but if the Administration Server fails due to the failure of `APPHOST1`, `APPHOST0` is failed over to `APPHOST2` by moving the virtual IP.

Figure 8–1 Administration Server High Availability Topology



8.4 Configuring Administration Server High Availability

This section includes the following topics

- Section 8.4.1, "Administration Server High Availability Requirements"
- Section 8.4.2, "Configuring the Administration Server for High Availability"

In a highly available Administration Server environment, both hosts must have access to shared storage. The domain directory is maintained in shared storage. During normal operation, the Administration Server on the active host owns the domain directory in shared storage. If the active host fails, the backup host takes over and restarts the Administration Server from the shared domain directory.

8.4.1 Administration Server High Availability Requirements

To configure a highly available Administration Server, your environment must meet the following requirements:

- Conform to the standard installation topology. See Section 1.6, "Understanding the Oracle Fusion Middleware Standard HA Topology" and Figure 1–1, "Oracle Fusion Middleware Highly Available Deployment Topology (Typical Enterprise)".
- Include two hosts, APPHOST1 and APPHOST2, to implement a WebLogic Server cluster (`cluster_1`). In Section 8.4.2, "Configuring the Administration Server for High Availability", IP addresses for APPHOST1 and APPHOST2 are `ip1` and `ip2`, respectively.

- APPHOST1 and APPHOST2 mount a common directory from shared storage and have read/write access rights to the directory. This directory is for installing products and storing domain directories.
- A reserved virtual IP address (`vip0`) to "point" to the host that runs the Administration Server. This floating virtual server IP address is configured dynamically on the host that the Administration Server runs on.
- Node Manager instances to manage the Administration Server and migrate it from the failed host to the designated standby host.

8.4.2 Configuring the Administration Server for High Availability

To configure the Administration Server for high availability, you must start with the standard high availability topology that has one cluster (`cluster_1`). See Section 1.6, "Understanding the Oracle Fusion Middleware Standard HA Topology".

To set up a highly available Administration Server, you run the Administration Server on a separate, virtual host (`APPHOST0`). You set up `APPHOST0` so that it maps to one of the existing hosts in the cluster (APPHOST1 or APPHOST2) by configuring a (virtual) server IP for `APPHOST0` on that existing host. If failover occurs, `APPHOST0` fails over by moving its virtual server IP to a surviving host. The domain configuration is on shared storage so that the surviving host can access it.

Note: There are multiple ways for Administration Server services to accomplish configuration tasks. No matter which method you use, the Administration Server must be running when you change the configuration.

Table 8–2 Host and Node Manager Terms

Term	Description
`APPHOST0`, `machine_0`	Virtual machine that the Administration Server runs on
`APPHOST1`, `APPHOST2`	Machines that host the application tier.
`vip0`	Virtual server IP address that the Administration Server listens on
`NMa`	Per-domain Node Manager that manages the Administration Server that runs on APPHOST0
`NM1`, `NM2`	Node Manager instances that run on APPHOST1 and APPHOST2, respectively
`ip1`, `ip2`	IP addresses of APPHOST1 and APPHOST2, respectively

To configure the Administration Server for high availability:

1. Configure a virtual server IP address (`vip0`) on `APPHOST1` to represent virtual host `APPHOST0`.

   See "Configuring Virtual Hosting" in *Administering Server Environments for Oracle WebLogic Server* for more information.

2. Use an Oracle Fusion Middleware expanded installation procedure to install the Oracle Fusion Middleware binaries and configure the domain into a directory on shared storage. Use `vip0` as the Administration Server listen address.

3. Create a virtual machine, `machine_0`, and add the Administration Server to it. The machine `machine_0` represents the virtual server host `APPHOST0` with the IP address `vip0`.
4. Create a cluster (`cluster_1`) that has two Managed Servers, `server_1` and `server_2`, that are assigned to `machine_1` and `machine_2`, respectively.
   - `machine_1` represents `APPHOST1` and `machine_2` represents `APPHOST2`.
   - `server_1` and `server_2` are set up to listen on `ip1` and `ip2`, respectively.
5. Scale out the virtual server to `APPHOST1` and `APPHOST2`. See Section 6.2, "Roadmap for Scaling Out Your Topology." To scale out you pack the domain in shared storage and unpack it to a local directory on `APPHOST1` and `APPHOST2`.
6. From `APPHOST1`, start a per-domain Node Manager (`NMa`) to manage the Administration Server listening on the configured virtual server IP `vip0` on `APPHOST1`. Start this instance of Node Manager from the domain directory in shared storage.
7. On `APPHOST1`, start a per-domain Node Manager (`NM1`) to manage `server_1` listening on `ip1`. Start this Node Manager (`NM1`) from the domain directory that you unpacked to local storage in `APPHOST1` in step 5.
8. On `APPHOST2`, start a per-domain Node Manager `(NM2`) to manage `server_2` listening on `ip2`. Start this Node Manager (`NM2)` from the domain directory that you unpacked to local storage in `APPHOST2` step 5.
9. Use Node Manager (`NMa`) to start the Administration Server on `APPHOST1`.
10. Start Managed Servers `server_1` and `server_2` using `NM1` and `NM2`, respectively.
11. Verify that the Administration Server and the Managed Servers are working properly. Connect to the Administration Server using the virtual IP address `vip0`.

8.5 Failing Over or Failing Back Administration Server

See the following topics to fail over or fail back the Administration Server after host failure:

- Section 8.5.1, "Failing Over the Administration Server if Original Host Fails"
- Section 8.5.2, "Failing Back the Administration Server to the Original Host"

8.5.1 Failing Over the Administration Server if Original Host Fails

To fail over the Administration Server to APPHOST2 if APPHOST1 fails.

1. Configure `vip0` on `APPHOST2`.
2. Start Node Manager NMa on `APPHOST2` from the domain directory in shared storage.
3. Start the Administration Server on `APPHOST2` using `NMa`.
4. Start the Administration Console to verify that the Administration Server is running.

8.5.2 Failing Back the Administration Server to the Original Host

You fail back the Administration Server to its original host after it restarts.

To fail back the Administration Server to `APPHOST1` when `APPHOST1` comes back online:

1. Stop the Administration Server on `APPHOST2` using Node Manager NMa.
2. Remove `vip0` from `APPHOST2`.
3. Stop Node Manager NMa on `APPHOST2`.
4. Configure vip0 on `APPHOST1`.
5. Start Node Manager NMa on `APPHOST1` using the domain directory in shared storage.
6. Use Node Manager NMa to start the Administration Server on `APPHOST1`.
7. Start the Administration Console to verify that the Administration Server is running.

Part III

Component Procedures

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

This part includes the following topics:

9 Configuring High Availability for Web Tier Components

Oracle Fusion Middleware's Web Tier is the architecture's outermost tier, closest to the end user. This section includes the following topics:

- Section 9.1, "Oracle HTTP Server and High Availability Concepts"
- Section 9.2, "Oracle HTTP Server Single-Instance Characteristics"
- Section 9.4, "Oracle HTTP Server Startup and Shutdown Lifecycle"
- Section 9.5, "Starting and Stopping Oracle HTTP Server"
- Section 9.6, "Oracle HTTP Server High Availability Architecture and Failover Considerations"
- Section 9.7, "Oracle HTTP Server Failure Protection and Expected Behaviors"
- Section 9.8, "Configuring Oracle HTTP Server Instances on Multiple Machines"
- Section 9.9, "Configuring Oracle HTTP Server for High Availability"

9.1 Oracle HTTP Server and High Availability Concepts

Oracle HTTP Server (OHS) is the Web server component for Oracle Fusion Middleware and the key Web Tier component. It provides a listener for Oracle WebLogic Server and a framework to host static pages, dynamic pages, and applications over the Web.

See Also: For more information on working with OHS, see:

- "Managing Oracle HTTP Server" in *Administering Oracle HTTP Server*. Includes the topics "Performing Basic OHS Tasks," "Creating an OHS Instance," and "Managing and Monitoring Server Processes.
- *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server*

9.2 Oracle HTTP Server Single-Instance Characteristics

Oracle HTTP Server (OHS) is based on Apache infrastructure and includes Oracle modules that you can use to extend OHS core functionality. Oracle HTTP Server has the following components to handle client requests:

- **HTTP listener** handles incoming requests and routes them to the appropriate processing utility.
- **Modules (mods)** implement and extend OHS functionality. Oracle HTTP Server includes many standard Apache modules. Oracle also includes modules that are specific to OHS to support OHS and OHS component integration.

Oracle HTTP Server can also be a proxy server, both forward and reverse. A reverse proxy enables content served by different servers to appear as if it comes from one server.

9.3 Oracle HTTP Server and Domains

Oracle HTTP Server does not require a WebLogic domain but you usually use it with one. Oracle recommends associating OHS with a domain because it enables OHS to be incorporated into the Administration Console, where you can manage and monitor it.

The `mod_wl_ohs` module handles the link to Managed Servers. This module is configured by routing requests of a particular type, such as JSPs, or by routing requests destined to a URL to specific Managed Servers.

Oracle HTTP Server usually front ends a cluster. In this configuration, a special `mod_wl_ohs` directive, `WebLogicCluster`, specifies a comma-separated list of cluster members.

The following steps describe the `mod_wl_ohs` directive process:

1. `mod_wl_ohs` receives a request for a Managed Server then sends the request to one cluster member in the directive. At least one Managed Server must be available to fulfill the request.
2. The Managed Server receives the request, processes it, and sends a complete list of cluster members back to `mod_wl_ohs`.
3. When `mod_wl_ohs` receives the updated list, it dynamically adds previously unknown servers to the known servers list. By doing this, all future requests are load balanced across the cluster member list. The benefit is that new Managed Servers are added to a cluster without updating `mod_wl_ohs` or adding OHS.

Note: The `mod_wl_ohs` directive `DynamicServerList` controls whether or not unknown servers are added to the known servers list. You must set `DynamicServerList` to `ON` to enable dynamic addition of servers.

Note: When you start, you don't need to include all current Managed Servers in the `mod_wl_ohs` directive. A high availability setup requires only two cluster members in the list for the first call to work. See "Configuring the WebLogic Proxy Plug-In for Oracle HTTP Server" in *Oracle Fusion Middleware Using Oracle WebLogic Server Proxy Plug-Ins* for more on running an OHS high availability deployment.

See Also: For more information on Oracle WebLogic clusters, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

9.4 Oracle HTTP Server Startup and Shutdown Lifecycle

After Oracle HTTP Server starts, it is ready to listen for and respond to HTTP(S) requests.

The request processing model is different on Microsoft Windows systems compared to UNIX systems:

- For Microsoft Windows, there is *one* parent process and *one* child process. The child process creates threads that handle client requests. The number of created threads is static and you can configure them for performance.
- For UNIX, there is *one* parent process that manages *multiple* child processes. Child processes handle requests. The parent process brings up more child processes as necessary, based on configuration.

See Also: For more information on the OHS processing model, see "Oracle HTTP Server Processing Model" in *Administrator's Guide for Oracle HTTP Server*.

9.5 Starting and Stopping Oracle HTTP Server

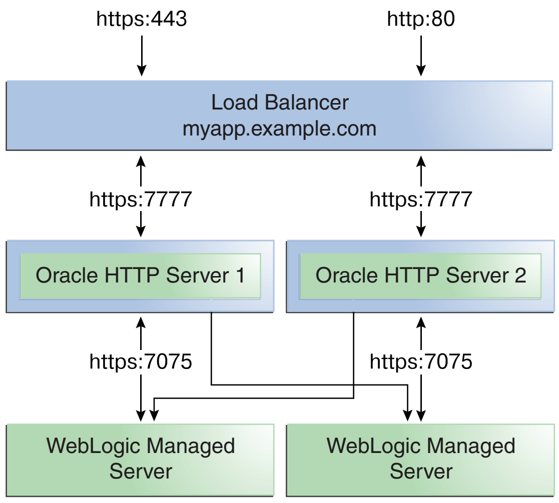
Use Fusion Middleware Control or the WebLogic Scripting Tool (WLST) to start, stop, and restart Oracle HTTP Server. If you plan to use WLST, you should familiarize yourself with that tool; see "Getting Started Using the Oracle WebLogic Scripting Tool (WLST)" in the *Oracle Fusion Middleware Administrator's Guide*.

See Also: For steps to start and stop OHS, see "Performing Basic Oracle HTTP Server Tasks" in *Administrator's Guide for Oracle HTTP Server*.

9.6 Oracle HTTP Server High Availability Architecture and Failover Considerations

The following figure shows two Oracle HTTP Servers behind a load balancer.

Figure 9–1 Oracle HTTP Server High Availability Architecture



The load balancer receives user requests and forwards them to connected Oracle HTTP Servers. The load balancer receives requests on standard HTTP/HTTPS ports (80/443). However, it then passes requests to Oracle HTTP Servers using completely different ports. Advantages of this setup are:

- Actual ports are hidden from users.
- Users don't have to add port numbers to the URL.

On UNIX-based systems, starting OHS with root privileges isn't mandatory. Only root can start a process that uses a port less than 1024. However, for processes that use a port number below 1024, you must use root privilege to start a process.

The load balancer routes requests to the functioning Oracle HTTP Servers.

The preceding figure also shows how OHS distributes requests to Managed Servers. For high availability, each pair of components (OHS and Managed Servers) should reside on different host computers. Managed Servers belong to the same cluster; to load balance across a set of Managed Servers, they must belong to the same cluster.

9.7 Oracle HTTP Server Failure Protection and Expected Behaviors

Oracle HTTP Server has two failure types: **process failures** and **node failures**. An individual operating system process may fail. A node failure can involve failure of the entire host computer that Oracle HTTP Server (OHS) runs on.

Table 9–1 OHS Failure Types and Failure Protections

Failure Type	Protection
Process	Node Manager protects and manages OHS processes. If an OHS process fails, Node Manager automatically restarts it.
Node	Load balancer in front of OHS sends a request to another OHS if the first one doesn't respond or URL pings indicate it failed.
Managed Server	If a Managed Server in a cluster fails, `mod_wl_ohs` automatically redirects requests to one of the active cluster members. If the application stores state, state replication is enabled within the cluster, which enables redirected requests access to the same state information.
Database	Typically, an issue only when using `mod_oradav` or `mod_plsql`. With Oracle RAC databases, the Oracle RAC connection determines failure characteristics. **If client connection failover is configured**, in-flight transactions roll back. Database reconnection is required. **If Transparent Application Failover (TAF) is configured**, any in-flight database write rolls back but automatic database reconnection occurs and select statements recover automatically. TAF fails over select statements only; package variables are lost. TAF, a JDBC Oracle Call Interface driver feature, enables an application to automatically reconnect to a database if the database instance the connection is made to fails. In this case, active transactions roll back.

9.8 Configuring Oracle HTTP Server Instances on Multiple Machines

If you use the Configuration Wizard to configure Oracle HTTP Server (OHS) and OHS is part of a domain, update the `mod_wl_ohs.conf` file for each instance. The file is in the *DOMAIN_HOME*`/config/fmwconfig/components/OHS/`*componentName* directory.

Restart the Administration Server to propagate changes to all OHS instances in the domain, even if they reside on a different host.

See Also: See Section 9.9.1.6, "Configuring mod_wl_ohs.conf" for more information on the `mod_wl_ohs` file.

Note: If you install and configure OHS instances in separate domains, you must manually copy changes to other Oracle HTTP servers. You must verify that the changes apply to all OHS instances and that they are synchronized.

9.9 Configuring Oracle HTTP Server for High Availability

This section describes how to configure an example high availability deployment of Oracle HTTP Server (OHS).

This section includes the following topics:

- Section 9.9.1, "Prerequisites to Configure a Highly Available OHS"
- Section 9.9.2, "Installing and Validating Oracle HTTP Server on WEBHOST2"
- Section 9.9.3, "Configuring and Validating an OHS High Availability Deployment"

9.9.1 Prerequisites to Configure a Highly Available OHS

See the following prerequisites before configuring a high availability Oracle HTTP Server deployment.

- Section 9.9.1.1, "Load Balancer Prerequisites"
- Section 9.9.1.2, "Configuring Load Balancer Virtual Server Names and Ports"
- Section 9.9.1.3, "Managing Load Balancer Port Numbers"
- Section 9.9.1.4, "Installing and Validating Oracle HTTP Server on WEBHOST1"
- Section 9.9.1.5, "Creating Virtual Host(s) on WEBHOST1"
- Section 9.9.1.6, "Configuring mod_wl_ohs.conf"

9.9.1.1 Load Balancer Prerequisites

To distribute requests against Oracle HTTP Servers, you must use an external load balancer to distribute HTTP(S) requests between available Oracle HTTP Servers. If you have an external load balancer, it must have features that Section 2.2.2, "Third-Party Load Balancer Requirements"describes.

9.9.1.2 Configuring Load Balancer Virtual Server Names and Ports

In an OHS installation, these virtual servers are configured for HTTP connections, which are distributed across the HTTP servers.

If your site serves requests for HTTP and HTTPS connections, Oracle recommends that HTTPS requests terminate at the load balancer and pass through as HTTP requests. To do this, the load balancer should be able to perform the protocol conversion and must be configured for persistent HTTP sessions.

This example configuration assumes that the load balancer is configured as:

- **Virtual Host:** `Myapp.example.com`
- **Virtual Port:** `7777`
- **Server Pool:** `Map`
- **Server:** `WEBHOST1, Port 7777, WEBHOST2, Port 7777`

9.9.1.3 Managing Load Balancer Port Numbers

Many Oracle Fusion Middleware components and services use ports. As an administrator, you must know the port numbers that services use and ensure that two services do not use the same port number on your host computer.

Most port numbers are assigned during installation. It is important that any traffic going from the Oracle HTTP Servers to the Oracle WebLogic Servers has access through any firewalls.

9.9.1.4 Installing and Validating Oracle HTTP Server on WEBHOST1

To install OHS on WEBHOST1, see the steps in "Installing the Oracle HTTP Server Software" in *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server*.

Validate the installation using the following URL to access the OHS home page:

```
http://webhost1:7777/
```

9.9.1.5 Creating Virtual Host(s) on WEBHOST1

For each virtual host or site name that you use, add an entry to the OHS configuration. Create a file named `virtual_hosts.conf` in the *ORACLE_HOME*`/config/fmwconfig/components/OHS/`*componentName*`/moduleconf` directory as follows:

```
NameVirtualHost *:7777
<VirtualHost *:7777>
 ServerName http://myapp.example.com:80
 RewriteEngine On
 RewriteOptions inherit
 UseCanonicalName On
</VirtualHost>
```

If you are using SSL/SSL Termination (*):

```
NameVirtualHost *:7777
<VirtualHost *:7777>
 ServerName https://myapp.example.com:443
 RewriteEngine On
 RewriteOptions inherit
 UseCanonicalName On
</VirtualHost>
```

Note: You can also use Fusion Middleware Control to create virtual hosts. See "Wiring Components Together" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*

9.9.1.6 Configuring mod_wl_ohs.conf

After you install and configure Oracle HTTP Server, link it to any defined Managed Servers by editing the `mod_wl_ohs.conf` file located in *DOMAIN_HOME*`/config/fmwconfig/components/OHS/`*componentName* directory.

See "Configuring the WebLogic Proxy Plug-In for Oracle HTTP Server" in *Oracle Fusion Middleware Using Oracle WebLogic Server Proxy Plug-Ins 12.1.2* for more information on editing the `mod_wl_ohs.conf` file.

Note: You can also use Fusion Middleware Control to link OHS to Managed Servers. See "Wiring Components Together" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*

The following example shows `mod_wl_ohs.conf` entries:

```
LoadModule weblogic_module PRODUCT_HOME/modules/mod_wl_ohs.so

<IfModule mod_weblogic.c>
 WebLogicCluster apphost1.example.com:7050, apphost2.example.com:7050
 MatchExpression *.jsp
 </IfModule>

<Location /weblogic>
 SetHandler weblogic-handler
 WebLogicCluster apphost1.example.com:7050,apphost2.example.com:7050
 DefaultFileName index.jsp
</Location>

<Location /console>
 SetHandler weblogic-handler
 WebLogicCluster apphost1.example.com
 WebLogicPort 7003
</Location>
```

These examples show two different ways to route requests to Managed Servers:

- The `<ifModule>` block sends any requests ending in *.jsp to the WebLogic Managed Server cluster located on APPHOST1 and APPHOST2.
- The `<Location>` block sends any requests with URLs that have a `/weblogic` prefix to the Managed Server cluster located on APPHOST1 and APPHOST2.

9.9.1.7 Configuring mod_wl_conf if you use SSL Termination

If you use SSL termination AND route requests to WebLogic, you must take the following additional configuration step:

1. In the WebLogic console, verify that WebLogic Plugin Enabled is set to true, either at the domain, cluster, or Managed Server level.
2. Add these lines to the Location block, which directs requests to Managed Servers:

   ```
   WLProxySSL ON
   WLProxySSLPassThrough ON
   ```

For example:

```
<Location /weblogic>
 SetHandler weblogic-handler
 WebLogicCluster apphost1.example.com:7050,apphost2.example.com:7050
```

```
 WLProxySSL On
 WLProxySSLPassThrough ON
 DefaultFileName index.jsp
</Location>
```

After you enable the WebLogic plugin, restart the Administration Server.

9.9.2 Installing and Validating Oracle HTTP Server on WEBHOST2

To install Oracle HTTP Server on WEBHOST2, see "Installing the Oracle HTTP Server Software" in *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server*.

Validate the installation on WEBHOST2 by using the following URL to access the Oracle HTTP Server home page:

`http://webhost2:7777/`

9.9.3 Configuring and Validating an OHS High Availability Deployment

To configure and validate the OHS high availability deployment, follow these steps:

- Section 9.9.3.1, "Configuring Virtual Host(s) on WEBHOST2"
- Section 9.9.3.2, "Validating the Oracle HTTP Server Configuration"

9.9.3.1 Configuring Virtual Host(s) on WEBHOST2

Update the `mod_wl_ohs.conf` file located in *DOMAIN_HOME*`/config/fmwconfig/components/OHS/`*componentName* directory. You must then restart the Administration Server to propagate changes to all OHS instances in the domain.

9.9.3.2 Validating the Oracle HTTP Server Configuration

Validate the configuration by using the following URLs:

`http://myapp.example.com/`

`https://myapp.example.com` (if using SSL/SSL termination)

`http://myapp.example.com:7777/weblogic`

10 Configuring High Availability for Oracle WebCenter Components

This section describes how to configure high availability for Oracle WebCenter products.

This section includes the following topics:

For more information on WebCenter, see:

- Oracle Fusion Middleware Installing and Configuring Oracle WebCenter Content
- Oracle Fusion Middleware Installing and Configuring Oracle WebCenter Portal
- Oracle Fusion Middleware Installing and Configuring Oracle WebCenter Sites
- Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter Portal

10.1 About Extending WebCenter Content: Inbound Refinery Components

WebCenter Content: Inbound Refinery components do not use WebLogic Server clustering for scale up and failover. You can deploy and configure multiple, independent Inbound Refinery Managed Servers for WebCenter Content to use.

10.2 About WebCenter Content Scale Up

You cannot scale up Oracle WebCenter Content.

10.3 About Creating WebCenter Portal Components on Multiple Nodes

In a WebCenter Portal high availability setup, you might see a delay of data appearing on the second node due to a 10-minute default refresh interval. For example, you may see this if you are trying to access a new Portal component that you just created on Node 2 if the Node 1 is processing the request.

10.4 About Creating a WebCenter Capture Domain with Oracle SOA

If you create a WebCenter Capture domain with an Oracle SOA component, you must select the JRF-MAN-SVR and WSMPM-MAN-SVR server groups. These server groups ensure that the oracle JRF and Oracle Web Services Manager (OWSM) services target the Managed Servers you are creating.

10.5 About Scaling Out WebCenter Capture and Configuring OWSM

OWSM uses cross-component wiring to auto-discover the Policy Manager in a domain. When you use the Configuration Wizard to create or update a domain that includes OWSM, Policy Manager URLs publish to the Local Service table. The OWSM Agent is automatically wired to the OWSM Policy Manager using endpoint entries published to the Local Service table. If, however, you change the domain using tools other than the Configuration Wizard (such as WebLogic Administration Console, Fusion Middleware Control, or WLST), any changes to the Policy Manager URL automatically publish to the Local Service table but the OWSM Agent client is not automatically bound to the new URL. In this case, you must manually bind OWSM Agent to the Policy Manager URL. For more information, see Verifying Agent Bindings Using Fusion Middleware Control in *Oracle Fusion Middleware Securing Web Services and Managing Polices with Oracle Web Services Manager*.

10.6 About WebCenter Sites Component Connections

WebCenter Sites requires that an external component be up and running when trying to establish connections. If the external component is not up and running or the connection has timed out, WebCenter Sites doesn't reestablish the connection.

WebCenter Sites verifies that database connections are active and reestablishes connections only when they are active.

10.7 About WebCenter Sites and Multicast

WebCenter Sites relies on multicast support to provide inCache replication across nodes in a clustered environment. For more about the inCache support in WebCenter Sites, see Using the inCache Framework in Oracle Fusion Middleware Administering Oracle WebCenter Sites.

11 Configuring High Availability for Other Components

This section describes information unique to certain component products.

For this release, this section includes the following topics:

- Section 11.1, "Deploying Oracle Data Integrator"
- Section 11.2, "Deploying Oracle Application Development Framework"
- Section 11.3, "Deploying BI"
- Section 11.4, "Deploying Forms"
- Section 11.5, "Deploying Reports"

11.1 Deploying Oracle Data Integrator

This topic describes considerations for configuring Oracle Data Integrator repository connections to Oracle Real Application Clusters:

- Section 11.1.1, "Oracle RAC Retry Connectivity for Source and Target Connections"
- Section 11.1.2, "Configuring ODI Repository Connections to Oracle RAC"
- Section 11.1.3, "About Oracle Data Integrator Scheduler Node Failure"

11.1.1 Oracle RAC Retry Connectivity for Source and Target Connections

When you configure Oracle Data Integrator (ODI) Oracle Real Application Clusters (RAC) connections, Oracle RAC retry is supported for the ODI master or ODI work repository. ODI uses transactional connections to source and target connections while running ODI scenarios. For these source and target connections, ODI does not support RAC retry connectivity. You cannot migrate these transactions to another node in Oracle RAC.

11.1.2 Configuring ODI Repository Connections to Oracle RAC

When you create an ODI repository using Repository Creation Utility (RCU), you specify the work repository connection JDBC URL. RCU stores the URL in the master repository contents. If the work repository JDBC URL is a single node URL, you should modify the URL to include the Oracle Real Application Clusters (Oracle RAC) failover address.

- If Oracle RAC is *not* configured with Single Client Access Name (SCAN), you can provide details of the Oracle RAC instances. In the work repository JDBC URL field, enter the Oracle RAC connectivity address in the format *host*:*port*. See the following example.

- If Oracle RAC is configured with SCAN, provide Oracle RAC instance details with the SCAN address.

The following example shows the JDBC URL format to connect to an Oracle RAC with two hosts when it does not use SCAN:

```
jdbc:oracle:thin:(DESCRIPTION =(LOAD_BALANCE=ON) (ADDRESS =(PROTOCOL =tcp)
(HOST =host1)(PORT =port1)) (ADDRESS =(PROTOCOL =tcp)(HOST =host2)
(PORT =port2)) (CONNECT_DATA =(SERVER=dedicated)
(SERVICE_NAME=service_name)))
```

See "Creating a Work Repository" in *Administering Oracle Data Integrator* for more information.

11.1.3 About Oracle Data Integrator Scheduler Node Failure

If a WebLogic Server failover occurs, the other WebLogic Server instance becomes the scheduler. A Coherence cache handles the scheduler lifecycle. Locking guarantees the scheduler uniqueness, and event notification provides scheduler migration. When an agent restarts and computes its schedule, it takes into account schedules in progress, which automatically continue their execution cycle beyond the server startup time. New sessions trigger as if the scheduler never stopped. Stale sessions move to an error state and remain in that state when they restart.

In an Oracle Data Integrator Agent cluster, if the Agent node that is the scheduler node fails, another node in the cluster takes over as the scheduler node. The new scheduler node reinitializes and runs all schedules from that point forward.

If a scheduled scenario with a repeatable execution cycle is running when the node crashes, the scenario does not continue its iterations on the new scheduler node from the point at which the scheduler node failed. For example, if a scheduled scenario is configured to repeat the execution 10 times after an interval of two minutes and the scheduler node fails during the third execution, the new scheduler node doesn't continue running the scenario for the next eight executions.

11.2 Deploying Oracle Application Development Framework

This section includes the following topic:

- Section 11.2.1, "Oracle JRF Asynchronous Web Services (Pinned Service Behavior)"

See Also: For more on Oracle Application Development Framework (ADF), see:

- Oracle ADF Key Concepts in *Understanding the Oracle Application Development Framework*
- *Oracle Fusion Middleware Administering Oracle ADF Applications*

11.2.1 Oracle JRF Asynchronous Web Services (Pinned Service Behavior)

When you use Oracle JRF Asynchronous Web Services, the asynchronous web service is pinned to a service and does not fail over. When you use a reliability protocol such as WS-RM, the higher-level protocol reconnects to a new server after a failure.

For more on Oracle JRF Asynchronous Web Services, see the *Domain Template Reference*.

11.3 Deploying BI

Topics in this section include the following:

- Section 11.3.1, "About BI Session Failover"
- Section 11.3.2, "About BI Essbase"
- Section 11.3.3, "About BI Studio"
- Section 11.3.4, "About Specifying Ports for Multiple Node Managers"
- Section 11.3.5, "About RAC Database Post Installation Configuration"
- Section 11.3.6, "About Scaling Out BI Publisher"

11.3.1 About BI Session Failover

If a BI managed server and/or host crashes, a user may need to login again. This depends on which application they are using at the time of the crash and whether or not SSO is in use.

11.3.2 About BI Essbase

Essbase does not support a high availability configuration. If a server fails, there is no loss of state; you can recover from a failure by redploying Essbase Cube.

11.3.3 About BI Studio

Studio does not support a high availability configuration. Oracle recommends performing xml import/export on a regular basis. This is the best practice for Studio recovery from a catalog failure.

11.3.4 About Specifying Ports for Multiple Node Managers

If you have more than one node manager per machine, verify that you specify your ports. For more information, see the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence*.

11.3.5 About RAC Database Post Installation Configuration

Oracle Business Intelligence requires additional configuration steps for whole server migration after installation. See "Using Whole Server Migration and Service Migration in an Enterprise Deployment" in *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence* for these steps.

11.3.6 About Scaling Out BI Publisher

Oracle Business Intelligence requires additional steps after you follow scale out steps in Chapter 6, "Scaling Out a Topology (Machine Scale Out).". Oracle BI requires you to change `setDomainEnv.sh` to the updated singleton-data-directory setting (SDD).

To complete Oracle BI scale out:

1. Move the SDD from local to shared storage so that all hosts can access it using the same path. For example, move it to:

   ```
   DOMAIN_HOME/bidata/net/yoursystem/scratch/sdd
   ```

2. Open DOMAIN_HOME/config/fmwconfig/bienv/core/bi-environment.xml (element bi:singleton-data-directory).

3. Change the `xdo.server.config.dir` path to refer to the new SDD path you just created.
4. Restart the server.

11.4 Deploying Forms

Topics in this section include the following:

- Section 11.4.1, "About Forms HTTP Session Failover"

11.4.1 About Forms HTTP Session Failover

If a Forms HTTP session fails, you must reconnect and restart your session with the Forms application.

11.5 Deploying Reports

Reports has the following considerations in a high availability set up:

- Section 11.5.1, "About Scaling Up in Reports"
- Section 11.5.1, "About Scaling Up in Reports"
- Section 11.5.2, "About Reports Multicast Communication"
- Section 11.5.3, "About Reports Shared-File Based Cache"
- Section 11.5.4, "About Reports Database Service Failure"
- Section 11.5.5, "About Reports OID/Shared Cache File System Failure"

11.5.1 About Scaling Up in Reports

If you scale up Reports components, Oracle recommends that you bring down all nodes and then restart them when configuration is complete.

See "Starting and Stopping Oracle Reports Server" in *Oracle Fusion Middleware Publishing Reports to the Web with Oracle Reports Services*.

11.5.2 About Reports Multicast Communication

Reports cluster members or individual clients use multicast to discover other nodes. There is no workaround to using multicast.

11.5.3 About Reports Shared-File Based Cache

Reports has shared file based cache as a singleton. If the cache fails, high availability also fails.

There is no workaround. If shared-file based cache fails, you must restart Reports servers.

11.5.4 About Reports Database Service Failure

Reports components can tolerate database failure. Reports retries the database connection three times. After the database is up, you must run Reports again.

11.5.5 About Reports OID/Shared Cache File System Failure

Reports does not have retries for OID/Shared cache file system failures. There is no workaround. After the external system is up, you must run Reports again.