

Oracle® Fusion Middleware

High Availability Guide

12c (12.1.3.0.0)

E49894-02

July 2014

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

Oracle Fusion Middleware High Availability Guide, 12c (12.1.3.0.0)

E49894-02

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Christine Ford

Contributing Author: Michael Blevins, Suvendu Ray, Peter LaQuerre

Contributor: Gururaj BS

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface	vii
Audience	vii
Purpose of this Guide	vii
Documentation Accessibility	vii
Related Documents	vii
Conventions	viii
Part I Introduction to High Availability	
1 Introduction and Roadmap	
1.1 How to Use This Guide	1-1
1.2 New and Changed Features in This Release	1-2
1.3 What is High Availability?	1-3
1.4 High Availability Solutions	1-3
1.5 Understanding the Oracle Fusion Middleware Standard HA Topology	1-3
2 High Availability Concepts	
2.1 Server Load Balancing in a High Availability Environment	2-1
2.1.1 Third-Party Load Balancer Requirements	2-2
2.1.2 Third-Party Load Balancer Configuration	2-3
2.1.3 Server Load Balancing with Oracle HTTP Server or Oracle Traffic Director	2-3
2.2 Application Failover	2-4
2.3 Real Application Clusters	2-5
2.4 Coherence Clusters and High Availability	2-5
2.5 Disaster Recovery	2-6
2.6 Install Time Configuration	2-6
2.6.1 Domain (Topology) Profiles	2-6
2.6.2 Persistence Profiles	2-7
2.7 Roadmap for Setting Up a High Availability Topology	2-8
3 Whole Server Migration	
3.1 About Whole Server Migration	3-1
3.2 Configuring Whole Server Migration for Managed Server Failover	3-2
3.2.1 Prerequisites for Configuring Automatic Whole Server Migration	3-2

3.2.2	Configuring Whole Server Migration.....	3-2
-------	---	-----

Part II Creating a High Availability Environment

4 Using Shared Storage

4.1	Overview of Shared Storage.....	4-1
4.2	Shared Storage Prerequisites.....	4-2
4.3	Using Shared Storage for Binary (Oracle Home) Directories.....	4-2
4.3.1	About the Binary (Oracle Home) Directories	4-3
4.3.2	About Sharing a Single Oracle Home	4-3
4.3.3	About Using Redundant Binary (Oracle Home) Directories	4-3
4.4	Using Shared Storage for Domain Configuration Files	4-4
4.4.1	About Oracle WebLogic Server Administration and Managed Server Domain Configuration Files 4-4	
4.4.2	Shared Storage Considerations for Administration and Managed Server Domain Configuration Files 4-4	
4.5	Shared Storage Requirements for JMS Stores and JTA Logs.....	4-5
4.6	Directory Structure and Configurations.....	4-5

5 Database Considerations

5.1	About Oracle Real Application Clusters	5-1
5.2	About RAC Database Connections and Failover	5-2
5.2.1	About XA Transactions.....	5-2
5.3	About Data Sources	5-2
5.3.1	Active GridLink Data Sources	5-3
5.3.2	Multi Data Sources	5-4
5.4	Configuring Active GridLink Data Sources with Oracle RAC	5-4
5.4.1	Requirements.....	5-4
5.4.2	Configuring Component Data Sources as Active GridLink Data Sources.....	5-5
5.4.3	Using SCAN Addresses for Hosts and Ports.....	5-6
5.5	Configuring Multi Data Sources.....	5-6
5.5.1	Configuring Multi Data Sources with Oracle RAC	5-6
5.5.1.1	Requirements	5-7
5.5.1.2	Configuring Component Data Sources as Multi Data Sources.....	5-7
5.5.1.3	Modifying or Creating Multi Data Sources After Initial Configuration	5-7
5.5.1.4	Configuring Schemas for Transactional Recovery Privileges.....	5-9
5.5.2	Configuring Multi Data Sources for MDS Repositories.....	5-9

6 JMS and JTA High Availability

6.1	About JMS and JTA Services for High Availability	6-1
6.2	Configuring JMS and JTA Services for High Availability	6-2
6.3	User-Preferred Servers and Candidate Servers.....	6-2
6.4	Considerations for Using File Persistence (WebLogic JMS)	6-3
6.4.1	Considerations for Using File Stores on NFS	6-3
6.5	Configuring WLS JMS with a Database Persistent Store	6-6
6.5.1	Prerequisites for Configuring WLS JMS with a Database Persistent Store	6-7

6.5.2	Switching WLS JMS File-Based Persistent Stores to Database Persistent Store	6-7
6.6	Configuring Database Stores to Persist Transaction Logs	6-8

7 Scaling Out a Topology (Machine Scale Out)

7.1	About Machine Scale Out	7-1
7.2	Roadmap for Scaling Out Your Topology	7-2
7.3	Optional Scale Out Procedure	7-3
7.4	About Scale Out Prerequisites	7-3
7.5	Resource Requirements	7-3
7.6	Creating a New Machine	7-4
7.6.1	Shutting Down the Managed Server	7-4
7.6.2	Creating a New Machine	7-4
7.6.3	Assigning Managed Servers to the New Machine	7-5
7.7	Configuring WLS JMS After Machine Scale Up or Scale Out	7-6
7.8	Packing the Domain on APPHOST1	7-7
7.9	Preparing the New Machine	7-8
7.10	Running Unpack to Transfer the Template	7-8
7.11	Starting the Node Manager	7-8
7.12	Starting the Managed Servers	7-8
7.13	Verifying Machine Scale Out	7-9
7.14	Configuring Multicast Messaging for WebLogic Server Clusters	7-9
7.14.1	Requirements for Configuring Multicast Messaging	7-9
7.14.2	Configuring Multicast Messaging	7-10

8 Administration Server High Availability

8.1	Role of the Administration Server	8-1
8.2	Role of Node Manager	8-2
8.3	Administration Server High Availability Topology	8-2
8.4	Configuring Administration Server High Availability	8-3
8.4.1	Requirements	8-3
8.4.2	Configuring the Administration Server	8-4
8.4.2.1	Failing Over the Administration Server	8-5
8.4.2.2	Failing Back the Administration Server to the Original Host	8-6

Part III Component Procedures

9 Configuring High Availability for Web Tier Components

9.1	Oracle HTTP Server and High Availability Concepts	9-1
9.2	Oracle HTTP Server Single-Instance Characteristics	9-2
9.2.1	Oracle HTTP Server and Oracle WebLogic Server	9-2
9.3	Oracle HTTP Server Startup and Shutdown Lifecycle	9-3
9.4	Starting and Stopping Oracle HTTP Server	9-3
9.5	Oracle HTTP Server High Availability Architecture and Failover Considerations	9-3
9.6	Oracle HTTP Server Protection from Failures and Expected Behaviors	9-4
9.7	Configuring Oracle HTTP Server Instances on Multiple Machines	9-5
9.8	Configuring Oracle HTTP Server for High Availability	9-5

9.8.1	Prerequisites	9-6
9.8.1.1	Configuring the Load Balancer	9-6
9.8.1.2	Installing Oracle HTTP Server on WEBHOST1	9-7
9.8.1.3	Creating Virtual Host(s) on WEBHOST1	9-7
9.8.1.4	Configuring mod_wl_ohs.conf.....	9-7
9.8.2	Installing Oracle HTTP Server on WEBHOST2	9-8
9.8.3	Configuring and Validating the OHS High Availability Deployment.....	9-9
9.8.3.1	Configuring Virtual Host(s) on WEBHOST2	9-9
9.8.3.2	Validating the Oracle HTTP Server Configuration	9-9

10 Configuring High Availability for Oracle SOA Suite

10.1	Configuring Oracle BAM for High Availability.....	10-1
10.1.1	Configuring Oracle BAM Managed Server JMS System Resources After Scale Up	10-1
10.1.1.1	Configuring Oracle BAM Server JMS Server	10-2
10.1.1.2	Configuring Oracle BAM CQService JMS Server	10-2
10.1.2	Configuring Automatic Service Migration for Oracle BAM	10-4

11 Configuring High Availability for Other Components

11.1	Deploying Oracle Data Integrator	11-1
11.1.1	Oracle RAC Retry Connectivity for Source and Target Connections	11-1
11.1.2	Configuring Repository Connections to Oracle RAC	11-1
11.1.3	Scheduler Node Failure	11-2
11.2	Deploying Oracle Application Development Framework.....	11-2
11.2.1	Oracle JRF Asynchronous Web Services (Pinned Service Behavior)	11-2
11.2.2	Modifying the mdsDS Data Source URL	11-3

Preface

This preface contains these sections:

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

Audience

The *Oracle Fusion Middleware High Availability Guide12c* (12.1.3.0.0) 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 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.
<i>italic</i>	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 chapters:

- [Chapter 1, "Introduction and Roadmap"](#)
- [Chapter 2, "High Availability Concepts"](#)
- [Chapter 3, "Whole Server Migration"](#)

Introduction and Roadmap

This chapter includes introductory information on how and why to use this guide and high availability environments.

This chapter includes the following topics:

- [Section 1.1, "How to Use This Guide."](#)
- [Section 1.2, "New and Changed Features in This Release"](#)
- [Section 1.3, "What is High Availability?"](#)
- [Section 1.4, "High Availability Solutions"](#)
- [Section 1.5, "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.

[Table 1–1](#) 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 the topic "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 the guide <i>Planning an Installation of Oracle Fusion Middleware</i>

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 progress through the 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 the topic "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 <i>Oracle Fusion Middleware Administrator's Guide</i>
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 <i>Oracle Fusion Middleware Tuning Performance Guide</i>
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	Node Manager enables you to start, shut down, and restart the Administration Server and Managed Server instances from a remote location, making this an essential utility for any high availability environment.	See the guide <i>Administering Node Manager for Oracle WebLogic Server</i>

1.2 New and Changed Features in This Release

Oracle Fusion Middleware 12c (12.1.3.0.0) includes the following new and changed concepts and features from previous Oracle Fusion Middleware releases:

- Support for Whole Server Migration. See [Chapter 3, "Whole Server Migration."](#)
- Support for Administration Server High Availability. See [Chapter 8, "Administration Server High Availability."](#)
- Support for SOA. See [Chapter 10, "Configuring High Availability for Oracle SOA Suite."](#)

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*
-
-

1.3 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.

An **active-active solution** deploys two or more active servers and can be used 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. **Active-passive solutions** deploy an active instance that handles requests and a passive instance that is on standby.

1.4 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.

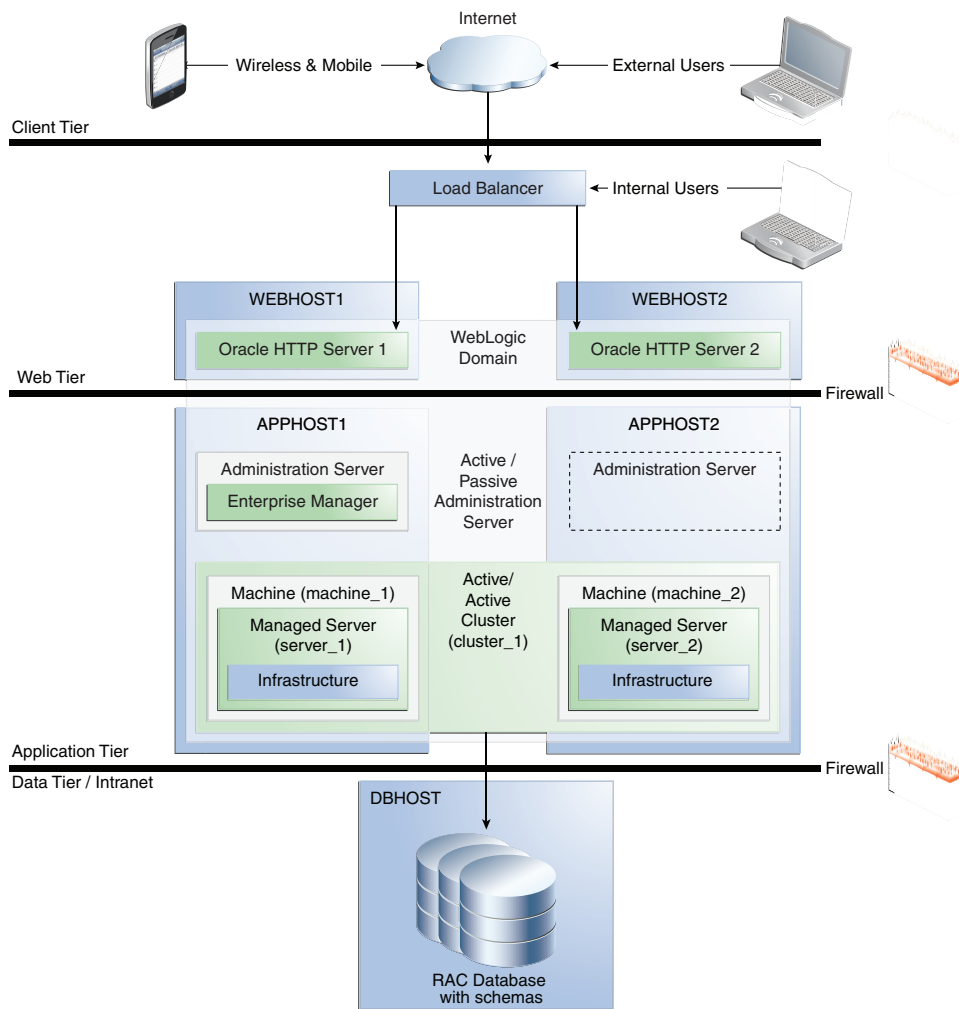
Disaster recovery solutions are usually geographically distributed deployments that protect your applications from disasters such as floods or regional network outages. You can protect against physical disasters that affect an entire data center by deploying geographically-distributed disaster recovery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

1.5 Understanding the Oracle Fusion Middleware Standard HA Topology

[Figure 1–1](#) 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 instructions in the *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure* and *Oracle Fusion Middleware Installing and Configuring Oracle HTTP Server* guides.

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



This topology represents 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.

Table 1–2 describes elements in Figure 1–1.

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	Refers to the machine that hosts the application tier.
WEBHOST	Refers to the 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 related software components. For more information, see "What is an Oracle WebLogic Server Domain?" in <i>Understanding Oracle Fusion Middleware</i> .

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

Element	Description and Links to Additional Documentation
Administration Server	The central control entity of a domain which maintains the domain's configuration objects and distributes configuration changes to Managed Servers.
Enterprise Manager	Oracle Enterprise Manager Fusion Middleware Control. This is 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 also the logical glue between Managed Servers and the 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 your applications, application components, Web services, and their associated resources. For more information, see "Oracle Enterprise Manager Fusion Middleware Control" in <i>Understanding Oracle Fusion Middleware</i> .
Infrastructure	Collection of services that includes: <ul style="list-style-type: none"> ■ Metadata repository (MDS) This contains metadata for Oracle Fusion Middleware components, such as the Oracle Application Developer Framework. For more information, see "What is the Metadata Repository?" in <i>Understanding Oracle Fusion Middleware</i>. ■ 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 Oracle Fusion Middleware Infrastructure Standard Installation Topology" in the guide *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 the guide *Installing and Configuring Oracle HTTP Server*.

High Availability Concepts

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

- Section 2.1, "Server Load Balancing in a High Availability Environment"
- Section 2.2, "Application Failover"
- Section 2.3, "Real Application Clusters"
- Section 2.4, "Coherence Clusters and High Availability"
- Section 2.5, "Disaster Recovery"
- Section 2.6, "Install Time Configuration"
- Section 2.7, "Roadmap for Setting Up a High Availability Topology"

For information on Oracle Fusion Middleware concepts, see the following topics in the guide *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*

2.1 Server Load Balancing in a High Availability Environment

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

Typically, Oracle Fusion Middleware high availability deployments are front ended by a load balancer that you configure to distribute incoming requests using various

algorithms. You can configure load balancing between different components or applications.

You can also configure load balancing between different components or applications by using Oracle HTTP Server or, on Exalogic only, Oracle Traffic Director. See [Section 2.1.3, "Server Load Balancing with Oracle HTTP Server or Oracle Traffic Director"](#) for more information.

You can also use a combination of a load balancer and the Oracle HTTP Server (as [Figure 1–1](#) shows) to provide maximum availability.

This section includes the following topics:

- [Section 2.1.1, "Third-Party Load Balancer Requirements"](#)
- [Section 2.1.2, "Third-Party Load Balancer Configuration"](#)
- [Section 2.1.3, "Server Load Balancing with Oracle HTTP Server or Oracle Traffic Director"](#)

2.1.1 Third-Party Load Balancer Requirements

You can use third-party load balancers in your Oracle Fusion Middleware high availability deployments.

Note: Oracle recommends a third-party load balancer that supports '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 using actual host names. The load balancer can then load balance requests to the servers in the pool.

Note: 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).
- Virtual servers and port configuration: ability to configure virtual server names and ports on your external load balancer, and the virtual server names and ports must meet the following requirements:
 - The load balancer should allow 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 Oracle WebLogic Clusters, the load balancer must be configured with a virtual server and ports for HTTP and HTTPS traffic.
 - The virtual server names must be associated with IP addresses and be part of your DNS. Clients must be able to access the external load balancer through the virtual server names.

- Ability to detect node failures and immediately stop routing traffic to the failed node.
- Resource monitoring /port monitoring /process failure detection: the load balancer must be able to detect service and node failures through notification or some other means and to stop directing non-Oracle Net traffic to the failed node. If your external load balancer has the ability to 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 the back-end services that it forwards traffic to are unavailable. This is preferred over the client disconnecting on its own after a timeout based on the TCP/IP settings on the client machine.
- SSL acceleration. Oracle recommends this feature but does not require it.
- Ability to Preserve the Client IP Addresses: the load balancer must have the capability to insert the original client IP address of a request in an X-Forwarded-For HTTP header to preserve the Client IP Address.

2.1.2 Third-Party Load Balancer Configuration

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 the documentation for your own load balancer. For high-level load balancer configuration steps, see the enterprise deployment guide for the component you are working with.

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

This section describes Oracle products you can use to provide load balancing.

Server Load Balancing with Oracle HTTP Server

Oracle HTTP Server is a web server with built-in WebLogic Server Proxy Plug-In module to act as HTTP front-end for one or more WebLogic servers. This built-in receives the incoming request from the client and load balances the request to one or more WebLogic Servers.

Oracle HTTP Server includes the `mod_wl_ohs` module, which routes requests to Oracle WebLogic Server. The `mod_wl_ohs` module provides the same load balancing functionality as `mod_weblogic`, the 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 9.8.1.4, "Configuring `mod_wl_ohs.conf`"](#) for more information on the `mod_wl_ohs` module.

Server Load Balancing with Oracle Traffic Director

Oracle Traffic Director is a highly available Application Delivery Controller with WebLogic inter-operability enhancements to allow incoming requests to be efficiently throttled to one or more WebLogic server clusters. Oracle Traffic Director is a fast, reliable, and scalable layer-7 software load balancer. You can set up Oracle Traffic

Director to serve as the reliable entry point for all HTTP, HTTPS and TCP traffic to application servers and web servers in the back end. Oracle Traffic Director distributes the requests that it receives from clients to servers in the back end based on the specified load-balancing method, routes the requests based on specified rules, caches frequently accessed data, prioritizes traffic, and controls the quality of service. You can use Oracle Traffic Director with Exalogic only.

2.2 Application Failover

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

Application failover is when an application component doing a particular job becomes unavailable for any reason and a copy of the failed component finishes the job.

Information about what has been done on a job is called **state**. WebLogic Server maintains state information using techniques called session replication and replica-aware stubs. When 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.

Session Failover Requirements

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

For seamless application 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 are using 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

Expected Behavior for Application Failover

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

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 because of 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 the *Domain Template Reference* for information on domain and extension templates that support high availability.

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

2.3 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 a single 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 simultaneously provides a highly scalable and highly available database for Oracle Fusion Middleware.

Every Oracle RAC instance in the cluster has equal access and authority. Node and instance failure may affect performance but does 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*
-

2.4 Coherence Clusters and High Availability

If you follow *Oracle Fusion Middleware Installing and Configuring Oracle WebLogic Server and Coherence* or a product installation guide, such as *Oracle Fusion Middleware Installing and Configuring Oracle Fusion Middleware for Oracle Application Development Framework*, the standard installation topology includes a standard Coherence cluster that serves as 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 referred to as WebLogic Managed Coherence Servers. JVMs that join a cluster are called **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 includes its Coherence configuration in a Grid Archive (GAR) file. The GAR file is deployed with the application and to all dedicated storage nodes.

All Fusion Middleware applications that use Coherence use the cluster associated with the managed Coherence server and deploy their GAR files co-located with their applications. [Table 2–2](#) provides additional sources of information about Coherence.

Table 2–2 Coherence and Coherence Clusters

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

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

2.6 Install Time Configuration

This section includes the following topics:

- [Section 2.6.1, "Domain \(Topology\) Profiles"](#)
- [Section 2.6.2, "Persistence Profiles"](#)

2.6.1 Domain (Topology) Profiles

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

All 12c (12.1.3.0.0) installation guides provide instructions for setting up a single machine, multi-server domain, referred to as the standard installation topology, which [Section 1.5, "Understanding the Oracle Fusion Middleware Standard HA Topology"](#) describes. See one of the following guides for more information:

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

2.6.2 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.

Table 2–3 shows persistence types for database and file persistence profiles.

Table 2–3 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 in Table 2–3 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 the file persistence store is used only in development mode, you can ignore it for high availability purposes. There is no 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.

Post-Configuration Defaults

Following a standard installation, the domain is set up with a file-based persistence profile. To configure database-based persistence for JMS/JTA resources, see [Chapter 6, "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.

Table 2–4 describes additional sources of information.

Table 2–4 Domain Configuration Topics

For additional information on...	See this topic...
Shared file systems for use with the file persistence profile	Chapter 4, "Using Shared Storage"
JMS and JTA	Section 6.2, "Configuring JMS and JTA Services for High Availability"
Failover	Section 2.2, "Application Failover"

2.7 Roadmap for Setting Up a High Availability Topology

This section provides high level steps for configuring an example middleware topology with high availability, such as the topology that [Section 1.5, "Understanding the Oracle Fusion Middleware Standard HA Topology"](#) describes.

[Table 2–5](#) describes the steps required to set up a high availability topology.

Table 2–5 Roadmap for Setting Up a High Availability Topology

Task	Description	Documentation
1. Install Real Application Clusters	Install Real Application Clusters	See <i>Oracle Real Application Clusters Administration and Deployment Guide</i>
2. Install and configure middleware components	Install and configure the application by following instructions in an application installation guide.	See <i>Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure</i> or the installation guide for your product
3. Install and configure Oracle HTTP Server	Install and configure Oracle HTTP Server in the same domain	See <i>Installing and Configuring Oracle HTTP Server</i>
4. Configure a load balancer	Configure a third-party load balancer that meets specific requirements, or Oracle HTTP Server/Oracle Traffic Director.	See Section 2.1, "Server Load Balancing in a High Availability Environment."
5. Scale out the topology (machine scale out)	Steps for scaling out a topology (machine scale-out) for all Fusion Middleware products that are a part of a Fusion Middleware WebLogic Server domain.	See Chapter 7, "Scaling Out a Topology (Machine Scale Out)"
6. Configure high availability for the Administration Server	Configure high availability for the Administration Server	See Chapter 8, "Administration Server High Availability"

Whole Server Migration

This chapter describes whole server migration and how to configure it for Managed Server failover. Whole server migration is needed when using special services such as JMS and JTA.

When **whole server migration** occurs, the server instance migrates to a different physical machine upon failure.

For detailed information on whole server migration, the topic "Whole Server Migration" in the guide *Administering Clusters for Oracle WebLogic Server*.

This chapter 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 the 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 referred to as **pinned services** because they remain active on only one server instance at a time.

In a cluster, most services deploy homogeneously on all Managed Server instances in the cluster, enabling transparent failover from one Managed Server to another. However, pinned services such as JMS and the JTA transaction recovery system are targeted at individual server instances within a cluster. For these services, WebLogic Server supports failure recovery with migration 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 becomes unavailable for any reason, for example, if it hangs, loses network connectivity, or its host system fails—migration is automatic. Upon failure, a migratable server automatically restarts on the same system if possible. If the migratable server cannot restart on the system it failed on, it migrates to another system. In addition, an administrator can manually initiate migration of a server instance.

See Also:

See "Whole Server Migration" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* for more information on preparing for automatic whole server migration, configuring 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 6, "JMS and JTA High Availability"](#) for more details on JMS and JTA services.

3.2 Configuring Whole Server Migration for Managed Server Failover

This section describes the 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.

Topics in this section include the following:

- [Section 3.2.1, "Prerequisites for Configuring Automatic Whole Server Migration"](#)
- [Section 3.2.2, "Configuring Whole Server Migration"](#)

3.2.1 Prerequisites for Configuring Automatic Whole Server Migration

Your system must meet specific requirements before you configure automatic whole server migration. See [Preparing for Automatic Whole Server Migration](#) in the guide *Administering Clusters for Oracle WLS*.

3.2.2 Configuring Whole Server Migration

Before configuring server migration, ensure that your environment meets the requirements outlined in [Section 3.2.1, "Prerequisites for Configuring Automatic Whole Server Migration"](#).

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

See the following topics for additional 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 contains the following chapters:

- [Chapter 4, "Using Shared Storage"](#)
- [Chapter 5, "Database Considerations"](#)
- [Chapter 6, "JMS and JTA High Availability"](#)
- [Chapter 7, "Scaling Out a Topology \(Machine Scale Out\)"](#)
- [Chapter 8, "Administration Server High Availability"](#)

Using Shared Storage

This chapter 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.

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.

This chapter includes the following topics:

- [Section 4.1, "Overview of Shared Storage"](#)
- [Section 4.2, "Shared Storage Prerequisites"](#)
- [Section 4.3, "Using Shared Storage for Binary \(Oracle Home\) Directories"](#)
- [Section 4.4, "Using Shared Storage for Domain Configuration Files"](#)
- [Section 4.5, "Shared Storage Requirements for JMS Stores and JTA Logs"](#)
- [Section 4.6, "Directory Structure and Configurations"](#)

4.1 Overview of Shared Storage

Shared storage allows sharing of dynamic state and server configuration and 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.

See [Table 4-1](#) for additional 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 <i>Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure</i>
Saving JMS and JTA information in a file store	"Using the WebLogic Persistent Store" in <i>Administering Server Environments for Oracle WebLogic Server</i> . Includes the topic "High Availability for Persistent Stores" "Persistent Store High Availability" in <i>Administering JMS Resources for Oracle WebLogic Server</i> Default File Store Availability for JTA in <i>Administering Clusters for Oracle WebLogic Server</i>

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 6.4.1, "Considerations for 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 that your options may be different from those in this example; see UNIX/Linux documentation for more information 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,wsiz=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 "Using the WebLogic Persistent Store" in *Administering Server Environments for Oracle WebLogic Server*.

4.3 Using Shared Storage for Binary (Oracle Home) Directories

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

- [Section 4.3.1, "About the Binary \(Oracle Home\) Directories"](#)
- [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 the Oracle home is patched or upgraded to a newer version.

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

The Oracle home for an Oracle Fusion Middleware installation contains the 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 more information about 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

Oracle Fusion Middleware enables you to 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 your 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 and Managed Server Domain 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 and Managed Server Domain Configuration Files

This section describes considerations for Administration Server and Managed Server configuration files.

Administration Server Configuration Directory

Oracle does not require that you 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 is running the Administration Server. If that host fails, you can mount the directory on a different host and bring up the failed Administration Server on the other host. For more information, see [Chapter 8, "Administration Server High Availability."](#)

Managed Server Configuration Files

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

It is possible to keep Managed Server configuration files on shared storage. However, doing so can affect performance due to multiple servers concurrently accessing 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 6.4, "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 the storage elements. Oracle recommends the following best practices:

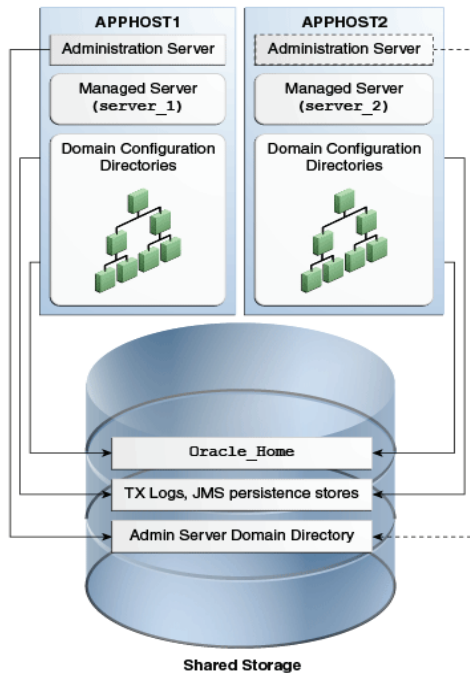
Table 4–2 *Shared Storage Elements Directory Structure*

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 and Managed Server Domain Configuration Files"](#) for more information.

Figure 4–1 illustrates the directory structure.

Figure 4-1 Shared Storage Directory Structure



Database Considerations

This chapter 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 Fusion Middleware 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 information on Oracle database options, see the *Oracle Database High Availability Overview*.

This chapter includes the following topics:

- [Section 5.1, "About Oracle Real Application Clusters"](#)
- [Section 5.2, "About RAC Database Connections and Failover"](#)
- [Section 5.3, "About Data Sources"](#)
- [Section 5.4, "Configuring Active GridLink Data Sources with Oracle RAC"](#)
- [Section 5.5, "Configuring Multi Data Sources"](#)

5.1 About Oracle Real Application Clusters

A **cluster** comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle RAC enables you to cluster an Oracle database, providing a highly scalable and highly available database for Oracle Fusion Middleware.

All Oracle Fusion Middleware components deployed to Oracle WebLogic Server support Oracle RAC.

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

[Table 5–1](#) outlines tasks and corresponding sources of information for setting up Oracle RAC.

Table 5–1 Roadmap for Setting up Oracle RAC

Task/Topic	More Information
About Oracle RAC	"Introduction to Oracle RAC" in the <i>Oracle Real Application Clusters Administration and Deployment Guide</i>

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

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

5.2 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. See [Section 5.2.1, "About XA Transactions"](#) for more information about XA transactions. These data sources also support load balancing across Oracle RAC nodes.

When an Oracle RAC node or instance fails, Oracle WebLogic Server or the Oracle Thin JDBC driver redirect session requests to another node in the cluster. There is no failover of existing connections. However, new connection requests from the application are managed using existing connections in the Oracle WebLogic pool or by new connections to the working Oracle RAC instance.

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

When 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.

5.2.1 About XA Transactions

XA transaction support enables multiple resources (such as databases, application servers, message queues, transactional caches) to be accessed 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 a single global transaction. This type of transaction 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 be passed 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.

5.3 About Data Sources

A **data source** is an abstraction that application 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 and do not need to be explicitly defined in an application's code. This abstraction allows applications to be built in a portable manner, because the application is not tied to a specific back-end database. The database can change without affecting the application code.

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

- If you use Oracle RAC database version 11g Release 2 and later, use 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 the Active GridLink data sources with Oracle RAC database for maximum availability. For versions of Oracle RAC databases where Active GridLink data sources are not supported, Oracle recommends using multi data sources for high availability.

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

- [Section 5.3.1, "Active GridLink Data Sources"](#)
- [Section 5.3.2, "Multi Data Sources"](#)

5.3.1 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 includes the 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 the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide for more information 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

5.3.2 Multi Data Sources

A **multi data source** is an abstraction around a group of data sources that provides load balancing or failover processing at the time of connection requests, between the data sources associated with the multi data source. Multi data sources support load balancing for both XA and non-XA data sources.

A multi data source provides an ordered list of data sources to use to satisfy 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 cannot 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: For more information about configuring Multi Data Sources with Oracle RAC, see "Using Multi Data Sources with Oracle RAC" in the guide *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

5.4 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 and the domain you are creating.

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

This topic includes the following sections:

- [Section 5.4.1, "Requirements"](#)
- [Section 5.4.2, "Configuring Component Data Sources as Active GridLink Data Sources"](#)
- [Section 5.4.3, "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*.

5.4.1 Requirements

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

- You are using 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**.

5.4.2 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 address is not validated during the domain creation process.

For the ONS host address, use the SCAN address for the Oracle RAC database 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 the SSL certificates, and the Wallet Password.

10. Select **Next**. Verify that all connections are successful.

Note: See "[Modifying the mdsDS Data Source URL](#)" to use an Active GridLink data source with a customer-provided ADF application.

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.
-

5.4.3 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 both the TNS listener and the 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 the *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server* guide.

5.5 Configuring Multi Data Sources

You can configure multi data sources using the following:

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

This section includes the following topics:

- [Section 5.5.1, "Configuring Multi Data Sources with Oracle RAC"](#)
- [Section 5.5.2, "Configuring Multi Data Sources for MDS Repositories"](#)

5.5.1 Configuring Multi Data Sources with Oracle RAC

This section describes the requirements and procedure to configure multi data sources with Oracle RAC.

This section includes the following topics:

- [Section 5.5.1.1, "Requirements"](#)
- [Section 5.5.1.2, "Configuring Component Data Sources as Multi Data Sources"](#)
- [Section 5.5.1.3, "Modifying or Creating Multi Data Sources After Initial Configuration"](#)

- [Section 5.5.1.4, "Configuring Schemas for Transactional Recovery Privileges"](#)

5.5.1.1 Requirements

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."

5.5.1.2 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.

5.5.1.3 Modifying or Creating Multi Data Sources After Initial Configuration

The multi data sources have constituent data sources for each RAC instance providing the database service. Oracle recommends that you add an additional data source to the multi data source on the Fusion Middleware tier when you 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 data source that is affected. The multi data source that you create must have constituent data sources for each RAC instance. The data source property values must be identical to the original single instance data source for the properties in [Section 5.5.1](#). For example, if the single instance data source driver is `oracle.jdbc.xa.client.OracleXADataSource`, it must be `oracle.jdbc.xa.client.OracleXADataSource` for each constituent data source of the new multi data source.

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"](#)
- [Table 5–3, "XA Data Source Configuration"](#)
- [Table 5–4, "Non-XA Data Source Configuration"](#)

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

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

If you see WARNING messages in the server logs that include the following exception, this message may indicate that the XA timeout value you have in your setup must be increased.

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

You can increase XA timeout for individual data sources when these warnings appear.

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

5.5.1.4 Configuring Schemas for Transactional Recovery Privileges

You need the appropriate database privileges to enable the 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 the schemas for transactional recovery privileges:

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

```
sqlplus "/ as sysdba"
```
2. Grant select on sys.dba_pending_transactions to the appropriate_user.
3. Grant force any transaction to the appropriate_user.

5.5.2 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) result in application read-only MDS operations being protected from Oracle RAC database planned and unplanned downtimes.

Multi data sources are exposed as MDS repositories in the Fusion Middleware Control navigation tree. 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* for more information.

- **Registering an MDS multi data source**

In addition to the steps in [Section 5.5.1, "Configuring Multi Data Sources with Oracle RAC,"](#) consider the following:

- The child data sources that constitute a multi data source used for an MDS repository must be configured as non-XA data sources.
- The multi data source's name must 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 through Fusion Middleware Control, WLST, and JDeveloper.

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

- **Converting a data source to a multi data source**

There are two things to consider when you convert a data source to a multi data source to verify that the application is configured correctly:

- To create a new multi data source with a new, unique name, redeploy the application and select this new multi data source as the MDS repository during deployment plan customization.
- To avoid redeploying the application, you can delete the data source and recreate the new multi data source using the same name and `jndi-name` attributes.

Note: See "[Modifying the mdsDS Data Source URL](#)" to use a multi data source with a customer-provided ADF application.

JMS and JTA High Availability

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

This chapter includes the following topics:

- [Section 6.1, "About JMS and JTA Services for High Availability"](#)
- [Section 6.2, "Configuring JMS and JTA Services for High Availability"](#)
- [Section 6.3, "User-Preferred Servers and Candidate Servers"](#)
- [Section 6.4, "Considerations for Using File Persistence \(WebLogic JMS\)"](#)
- [Section 6.5, "Configuring WLS JMS with a Database Persistent Store"](#)
- [Section 6.6, "Configuring Database Stores to Persist Transaction Logs"](#)

See Also: For more information on working with JMS or JTA, see one of the following topics:

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

6.1 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 the 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 includes features for automatically restarting and/or migrating a JMS server after a failure. It also includes 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."](#)

6.2 Configuring JMS and JTA Services for High Availability

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

To configure a JMS service for migration, you must deploy it to a migratable target. The migratable target specifies a set of servers that can host a target, and can specify a user-preferred host for the services and an ordered list of candidate backup servers if the preferred server fails. Only one server can host the migratable target at any one time.

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

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

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

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

6.3 User-Preferred Servers and Candidate Servers

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

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

See Also: For more information, see the following topics in the *Administration Console Online Help*:

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

6.4 Considerations for Using File Persistence (WebLogic JMS)

You can configure JMS messages and JTA logs to be stored on the file system. For high availability, you must use a shared file system. See [Chapter 4, "Using Shared Storage"](#) for information on what to consider when using a shared file system to store these artifacts.

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

6.4.1 Considerations for Using File Stores on NFS

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

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

- If you configured the server for server migration, it should start automatically in failover 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 Oracle WebLogic Server does not restart after abrupt machine failure, verify whether or not it is due to an I/O exception that is similar to the following by reviewing the server log files:

```
<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 does not release locks on the file stores. WebLogic Server maintains locks on files that store JMS data and transaction logs to prevent data corruption that can occur if you accidentally start two instances of the same Managed Server. Because the NFSv3 storage device doesn't track lock owners, NFS holds the lock indefinitely if a lock owner fails. As a result, after abrupt machine

failure followed by a restart, subsequent attempts by WebLogic Server to acquire locks may fail.

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

- **For NFSv4 environments**, you can set a tuning parameter on the NAS server to release locks within the approximate time required to complete server migration; you do not need to follow the procedures in this section. See your storage vendor's documentation for information on locking files stored in NFS-mounted directories on the storage device, and test the results.
- **For NFSv3 environments**, the following sections describe how to disable WebLogic file locking mechanisms for: the default file store, a custom file store, a JMS paging file store, a diagnostics file store.

WARNING: NFSv3 file locking prevents severe file corruptions that occur if more than one Managed Server writes to the same file store at any point in time.

If you disable NFSv3 file locking, you must implement administrative procedures /policies to ensure that only one Managed Server writes to a specific file store. Corruption can occur with two Managed Servers in the same cluster or different clusters, on the same node or different nodes, or on the same domain or different domains.

Your policies could include: never copy a domain, never force a unique naming scheme of WLS-configured objects (servers, stores), each domain must have its own storage directory, no two domains can have a store with the same name that references the same directory.

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.

Disabling File Locking for the Default File Store

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

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

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

```
<server>
<name>examplesServer</name>
```



```

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

```

Disabling File Locking for a Custom File Store

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

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

```

Disabling File Locking for a JMS Paging File Store

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

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

6. Restart the server you modified for the changes to take effect.

The `config.xml` file entry will look 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>
```

Disabling File Locking for a Diagnostics File Store

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

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

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

```
<server>
<name>examplesServer</name>
...
<server-diagnostic-config>
<diagnostic-store-dir>data/store/diagnostics</diagnostic-store-dir>
<diagnostic-store-file-locking-enabled>>false</diagnostic-store-file-locking-enabled>
<diagnostic-data-archive-type>FileStoreArchive</diagnostic-data-archive-type>
<data-retirement-enabled>>true</data-retirement-enabled>
<preferred-store-size-limit>100</preferred-store-size-limit>
<store-size-check-period>1</store-size-check-period>
</server-diagnostic-config>
</server>
```

See Also: For more information, see one of the following topics:

- "Configure JMS Servers and Persistent Stores" in *Oracle Fusion Middleware Administering JMS Resources for Oracle WebLogic Server*.
 - "Using the WebLogic Persistent Store" in *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*.
-
-

6.5 Configuring WLS JMS with a Database Persistent Store

The persistent store provides a built-in, high-performance storage solution for WebLogic Server subsystems and services that require persistence. For example, it can store persistent JMS messages.

The WLS JMS persistent store supports persistence to a file-based store or to a JDBC-accessible store in a database. This topic describes how to change the WLS JMS configuration from a file-based persistent store (the default configuration) to a database persistent store.

For information on the persistent store, see "Using the WebLogic Persistent Store" in the guide *Administering Server Environments for Oracle WebLogic Server*.

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

This section includes the following topics:

- [Section 6.5.1, "Prerequisites for Configuring WLS JMS with a Database Persistent Store"](#)
- [Section 6.5.2, "Switching WLS JMS File-Based Persistent Stores to Database Persistent Store"](#)
- [Section 6.6, "Configuring Database Stores to Persist Transaction Logs"](#)

6.5.1 Prerequisites for Configuring WLS JMS with a Database Persistent Store

To configure WLS JMS with database persistent stores, verify that your setup meets the following 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.

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

This section describes the steps for switching to a database-based persistent store for WLS JMS from the default file-based 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 the guide *Oracle Fusion Middleware Administering Server Environments for Oracle WebLogic Server* for detailed steps.

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.

6.6 Configuring Database Stores to Persist Transaction Logs

This section describes the steps to configure database-based stores for the Transaction Logs (TLOGs). Before you configure a JDBC TLOG Store, you must have a standard Oracle Fusion Middleware installation.

After a standard installation, the TLOG Store is configured to be in the file system. In some instances, Oracle recommends that you configure the TLOGs to be stored in the database.

To configure the JDBC TLOGs to be stored in the database store, see the detailed steps in "Using a JDBC TLOG Store" in the guide *Oracle Fusion Middleware Administering Server Environments for Oracle WebLogic Server*.

Note the following:

- 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.
- For a high availability setup, verify that the data source that you created for the persistent store is targeted to the cluster.

When you complete 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.

Scaling Out a Topology (Machine Scale Out)

This chapter describes the steps for scaling out a topology (machine scale-out) for all Fusion Middleware products that are a part of a Fusion Middleware 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 the consumers of your deployed applications if one computer goes down.

This chapter includes the following topics:

- [Section 7.1, "About Machine Scale Out"](#)
- [Section 7.2, "Roadmap for Scaling Out Your Topology"](#)
- [Section 7.3, "Optional Scale Out Procedure"](#)
- [Section 7.4, "About Scale Out Prerequisites"](#)
- [Section 7.5, "Resource Requirements"](#)
- [Section 7.6, "Creating a New Machine"](#)
- [Section 7.7, "Configuring WLS JMS After Machine Scale Up or Scale Out"](#)
- [Section 7.8, "Packing the Domain on APPHOST1"](#)
- [Section 7.9, "Preparing the New Machine"](#)
- [Section 7.10, "Running Unpack to Transfer the Template"](#)
- [Section 7.11, "Starting the Node Manager"](#)
- [Section 7.12, "Starting the Managed Servers"](#)
- [Section 7.13, "Verifying Machine Scale Out"](#)
- [Section 7.14, "Configuring Multicast Messaging for WebLogic Server Clusters"](#)

7.1 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. For more information on scaling up your environment, see "Scaling Up Your Environment" in the guide *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

7.2 Roadmap for Scaling Out Your Topology

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

Table 7–1 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 7.4, "About Scale Out Prerequisites"
Verify that you meet resource requirements	You must verify that your environment meets certain requirements	Section 7.5, "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 7.6, "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 7.7, "Configuring WLS JMS After Machine Scale Up or Scale Out"
Run the pack command	Pack up the domain directory	Section 7.8, "Packing the Domain on APPHOST1"
Prepare the new machine	Install the same software that you installed on the first machine	Section 7.9, "Preparing the New Machine"
Run the unpack command	Create a Managed Server template.	Section 7.10, "Running Unpack to Transfer the Template"
Start the server	Starts the Managed Server on the new machine	Section 7.12, "Starting the Managed Servers"
Verify the topology	Test the new setup	Section 7.13, "Verifying Machine Scale Out"
Set the cluster messaging mode to Multicast	Modifies messaging mode from Unicast to Multicast (preferred for multi-server domains)	Section 7.14, "Configuring Multicast Messaging for WebLogic Server Clusters"

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

7.3 Optional Scale Out Procedure

Following the standard installation topology results in 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 it can meet a variety of 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, as additional computing resources are required. An additional benefit of this approach is that it facilitates troubleshooting; you can validate the basic domain then perform and troubleshoot scale up and scale out steps 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 simply perform the pack and unpack steps.

If you have already assigned Managed Servers to their target machines, either in the initial installation process or through an online administrative operation, simply skip [Section 7.6, "Creating a New Machine"](#) as you progress through the roadmap ([Table 7-1, "Roadmap for Scaling Out Your Topology"](#)).

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

7.4 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 serves as 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 the topic "Understanding the Oracle Fusion Middleware Infrastructure Standard Installation Topology" describes in the guide *Oracle Fusion Middleware Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

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

7.5 Resource Requirements

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

- At least one machine running 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).

7.6 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 WebLogic Server instances that they host.

Follow these steps to create a new machine:

- [Section 7.6.1, "Shutting Down the Managed Server"](#)
- [Section 7.6.2, "Creating a New Machine"](#)
- [Section 7.6.3, "Assigning Managed Servers to the New Machine"](#)

7.6.1 Shutting Down the Managed Server

The server must be in a shutdown state before moving to a new machine. If the server 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.

7.6.2 Creating a New Machine

This topic describes how to create a new machine using the Administration Console. To create a new machine using WLST commands see "Creating a New Machine for Certain Components" in the guide *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 using the Administration Console, perform the following steps:

1. If the domain Administration Server is not running, you must start it. Go to the DOMAIN_HOME/bin directory and run:

```
./startWeblogic.sh
```

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

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

On the Welcome screen, log in.

3. In the Change Center of the Administration Console, 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.

7.6.3 Assigning Managed Servers to the New Machine

To add Managed Servers to the newly-created machine, use the Administration Console to perform the following steps:

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 on the right shows all servers in the domain and their machine assignments.

7.7 Configuring WLS JMS After Machine Scale Up or Scale Out

This section describes the steps to configure 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 one of the existing Managed Servers 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. Oracle recommends that you follow the same naming pattern for manageability and usability.

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 persistent store for *server_1*. 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 will have a common root. For example for the 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. Navigate 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.

When you complete these steps, a new Managed Server that includes configured JMS resources is available in the cluster.

Note: See [Section 10.1.1, "Configuring Oracle BAM Managed Server JMS System Resources After Scale Up"](#) to configure WLS JMS for Oracle BAM components. [Section 10.1.1](#) describes additional steps that you must complete for Oracle BAM components after you configure JMS system resources.

7.8 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 will unpack the template file on **APPHOST2** later in the scale out process; [Section 7.10, "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 are creating 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.

7.9 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.

7.10 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
```

7.11 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.

7.12 Starting the 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 the guide *Oracle Fusion Middleware Administering Oracle Fusion Middleware*.

7.13 Verifying Machine Scale Out

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

7.14 Configuring Multicast Messaging for WebLogic Server Clusters

This section describes the steps to configure Multicast messaging for WebLogic Server cluster communication.

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

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

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

This section includes the following topics:

- [Section 7.14.1, "Requirements for Configuring Multicast Messaging"](#)
- [Section 7.14.2, "Configuring Multicast Messaging"](#)

7.14.1 Requirements for Configuring Multicast Messaging

Requirements for configuring Multicast messaging include the following:

- A configured domain with at least one cluster.
- A hardware configuration that is 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 MulticastTest utility helps you debug multicast problems when you configure a cluster. The utility sends out multicast packets and returns information about how effectively multicast is working on your network.

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

7.14.2 Configuring Multicast Messaging

To configure Multicast Messaging:

1. Log in to the Administration Console.
2. Shut down all servers for which you want to use Multicast messaging.
3. In the Domain Structure pane on the left, expand **Environment** and click **Clusters**.
4. Select the cluster name for which you want to enable Multicast.
5. Click on **Configuration** tab then the **Messaging** tab.
6. For Messaging Mode, select **Multicast**. Enter the Multicast Address then enter the Multicast Port.

The **Multicast Address** must be unique for each cluster. See "Cluster Multicast Address and Port" in the guide *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* for guidelines on Multicast addresses.

7. Click **Advanced** to configure the following parameters:

Table 7-2 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 is allowed to travel. If your cluster spans multiple subnets in a WAN, the value of this parameter must be high enough to ensure that routers do not discard multicast packets before they reach their final destination. This parameter sets the number of network hops a multicast message makes before the packet can be discarded.
Multicast Buffer Size	The multicast socket send/receive buffer size (at least 64 kilobytes).
Idle Periods Until Timeout	Maximum number of periods that a cluster member will wait before timing out a member of a cluster.
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 is ready to use Multicast messaging. When you select **Clusters** in the Domain Structure pane in the Administration Console, **Multicast** appears for the **Cluster Messaging Mode**.

Administration Server High Availability

This chapter describes high availability aspects of the Administration Server.

This chapter includes the following topics:

- [Section 8.1, "Role of the Administration Server"](#)
- [Section 8.2, "Role of Node Manager"](#)
- [Section 8.3, "Administration Server High Availability Topology"](#)
- [Section 8.4, "Configuring Administration Server High Availability"](#)

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

Table 8–1 Administration Server Topics

For information on...	See this topic...
Starting and Stopping the Administration Server	"Starting and Stopping Administration Server" in the guide <i>Oracle Fusion Middleware Administering Oracle Fusion Middleware</i>
Configuring virtual hosting	"Configuring Virtual Hosting" in the guide <i>Administering Server Environments for Oracle WebLogic Server</i>
Using Node Manager	In the guide <i>Administering Node Manager for Oracle WebLogic Server</i> : <ul style="list-style-type: none"> ■ "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. It maintains the domain's configuration documents and distributes changes in the configuration documents to Managed Servers. The Administration Server serves as a central location from which to manage and monitor all resources in a domain.

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

Administration Server Failure

The failure of an Administration Server does not affect the operation of Managed Servers in the domain. If an Administration Server fails because of a hardware or software failure on its host computer, other server instances on the same computer may be similarly affected.

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

For the procedure to restart an Administration Server, see *Oracle Fusion Middleware Using Clusters for Oracle Server*.

Shared Storage

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

8.2 Role of Node Manager

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

You can configure the scope of Node Manager:

- **per domain** With a per-domain Node Manager, the Node Manager is associated with a domain and is configured to control all servers for the domain on a machine.
- **per host** With a per host Node Manager, the Node Manager process is not associated with a specific WebLogic Server domain but with a machine. You can use the same Node Manager process to control server instances in any WebLogic Server 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—whether Administration Server or Managed Server—that you want to control with Node Manager.

Node Manager failure does not affect the operation of any of the servers running on the machine.

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

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

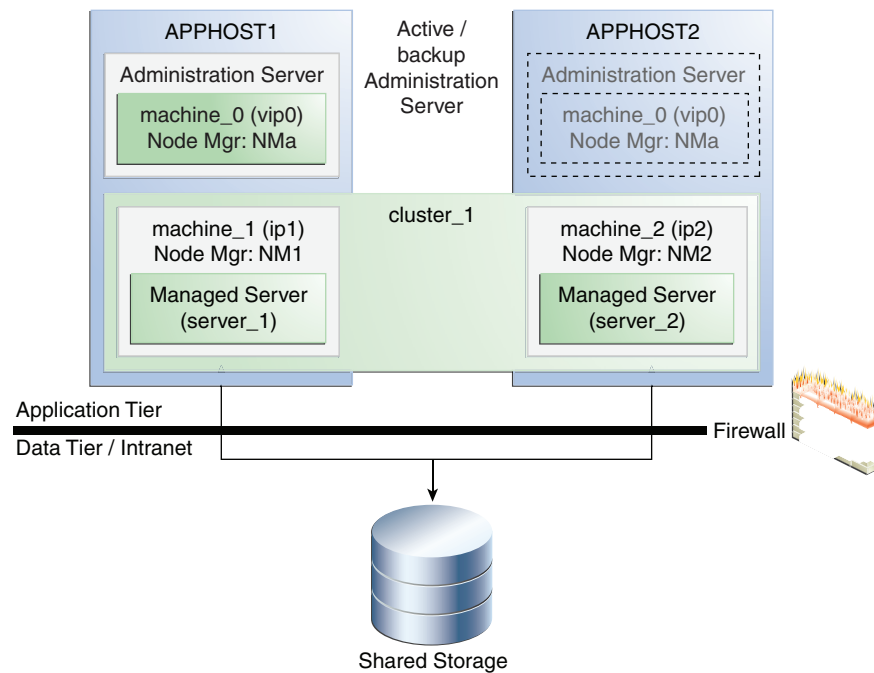
8.3 Administration Server High Availability Topology

As a single entity, the Administration Server can be active on a single host at any one time. To achieve high availability, you configure the Administration Server on a virtual host so that if the machine that the Administration Server runs on fails, you can fail it over to another host in the domain. The Administration Server is configured to use a virtual IP to overlap the backup hosts. You configure the Administration Server to listen on this virtual IP. The benefit of using a virtual host and virtual IP is that you do not need to add a third machine; if failover occurs, the virtual host can be mapped 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 is made active and you must move (manually) the virtual IP to the new active host. The new active host will then service all requests through the virtual IP. (You configure a high availability deployment to listen on this virtual IP.)

Figure 8–1 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 describes how to configure the Administration Server for high availability. It includes the following topics

- [Section 8.4.1, "Requirements"](#)
- [Section 8.4.2, "Configuring the Administration Server"](#)

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 Requirements

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

- Conform to the standard installation topology. See [Section 1.5, "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"](#), the 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 used for installing the products and storing the domain directories.
- A reserved virtual IP address (`vip0`) to "point" to the host that is running 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

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 1.5, "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 a variety of ways to invoke 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, <code>machine_0</code>	Virtual machine that the Administration Server runs on
APPHOST1, APPHOST2	Machines that host the application tier.
<code>vip0</code>	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	Instances of Node Manager that run on APPHOST1 and APPHOST2, respectively
<code>ip1</code> , <code>ip2</code>	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 the guide *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 7.2, "Roadmap for Scaling Out Your Topology."](#) You scale out by packing the domain in shared storage and unpacking 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`.

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

- [Section 8.4.2.1, "Failing Over the Administration Server"](#)
- [Section 8.4.2.2, "Failing Back the Administration Server to the Original Host"](#)

8.4.2.1 Failing Over the Administration Server

This procedure describes how 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.

8.4.2.2 Failing Back the Administration Server to the Original Host

This section describes how to fail back the Administration Server. You fail back the Administration Server to its original host after it restarts.

When APPHOST1 comes back online, fail back the Administration Server to APPHOST1 using the following procedure

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 chapters:

- [Chapter 9, "Configuring High Availability for Web Tier Components"](#)
- [Chapter 10, "Configuring High Availability for Oracle SOA Suite"](#)
- [Chapter 11, "Configuring High Availability for Other Components"](#)

Configuring High Availability for Web Tier Components

The Web Tier of Oracle Fusion Middleware is the outermost tier in the architecture, closest to the end user. The key component of the Web Tier is Oracle HTTP Server. This chapter describes high availability concepts and configuration procedures for Oracle HTTP Server and includes the following topics:

- Section 9.1, "Oracle HTTP Server and High Availability Concepts"
- Section 9.2, "Oracle HTTP Server Single-Instance Characteristics"
- Section 9.3, "Oracle HTTP Server Startup and Shutdown Lifecycle"
- Section 9.4, "Starting and Stopping Oracle HTTP Server"
- Section 9.5, "Oracle HTTP Server High Availability Architecture and Failover Considerations"
- Section 9.6, "Oracle HTTP Server Protection from Failures and Expected Behaviors"
- Section 9.7, "Configuring Oracle HTTP Server Instances on Multiple Machines"
- Section 9.8, "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. It provides a listener for Oracle WebLogic Server and the framework for hosting static pages, dynamic pages, and applications over the Web.

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

- "Managing Oracle HTTP Server" in the guide *Administering Oracle HTTP Server*. This section includes topics such as "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 is based on Apache infrastructure and includes modules developed by Oracle that you can use to extend Oracle HTTP Server core functionality. Oracle HTTP Server has the following components to handle client requests:

- **HTTP listener**, to handle incoming requests and route them to the appropriate processing utility.
- **Modules (mods)**, to implement and extend the basic functionality of Oracle HTTP Server. Many of the standard Apache modules are included with Oracle HTTP Server. Oracle also includes several modules that are specific to Oracle HTTP Server to support integration between Oracle HTTP Server and other Oracle HTTP Server components.

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.

9.2.1 Oracle HTTP Server and Oracle WebLogic Server

Oracle HTTP Server does not require an Oracle WebLogic domain but is usually used in conjunction with one. Oracle recommends associating Oracle HTTP Server with a WebLogic domain because it enables Oracle HTTP Server to be incorporated into the Administration Console for centralized management and monitoring.

The link to WebLogic Managed Servers is handled through the `mod_wl_ohs` module. This module is configured by routing requests of a particular type, for example, JSPs, or by routing requests destined to a URL to specific Managed Servers.

Typically you use Oracle HTTP Server to front end a cluster of WebLogic servers. When Oracle HTTP Server front-ends a cluster of WebLogic servers, a special `mod_wl_ohs` directive, `WebLogicCluster`, specifies a comma-separated list of cluster members. The process works as follows:

1. When `mod_wl_ohs` receives a request requiring a Managed Server, it sends that request to one of the WebLogic cluster members listed in the directive. At least one server must be available to service the request.
2. When the Managed Server receives the request, it processes it and sends a complete list of cluster members back to `mod_wl_ohs`.
3. When the `mod_wl_ohs` directive receives the updated list, it dynamically adds any previously unknown servers to the list of known servers. This action enables all future requests to be load balanced across the full cluster member list. The advantage of this process is that it enables new Managed Servers to be added to the cluster without updating `mod_wl_ohs` or adding the Oracle HTTP Server.

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 this dynamic addition of servers.

Note: You do not need to include all current Managed Servers in the `mod_wl_ohs` directive when you start; 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 the guide *Oracle Fusion Middleware Using Oracle WebLogic Server Proxy Plug-Ins* for more information on running a high availability deployment of Oracle HTTP Server.

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

9.3 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 on Microsoft Windows systems differs from that on UNIX systems:

- For Microsoft Windows, there is a single parent process and a single child process. The child process creates threads that are responsible for handling client requests. The number of created threads is static and can be configured for performance.
- For UNIX, there is a single parent process that manages multiple child processes. The child processes are responsible for handling requests. The parent process brings up additional child processes as necessary, based on configuration.

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

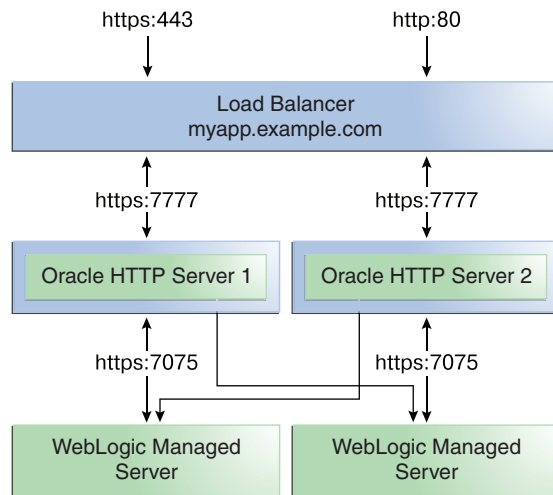
9.4 Starting and Stopping Oracle HTTP Server

You can 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 information on starting and stopping OHS, see Starting, Stopping, and Restarting Oracle HTTP Server in the *Administrator's Guide for Oracle HTTP Server*.

9.5 Oracle HTTP Server High Availability Architecture and Failover Considerations

Figure 9–1 shows two Oracle HTTP Servers placed behind a load balancer.

Figure 9–1 Oracle HTTP Server High Availability Architecture

The load balancer receives requests from users and forwards them on to the connected Oracle HTTP Servers. In [Figure 9–1](#), the load balancer receives the requests on the standard HTTP/HTTPS ports (80/443). However, it then passes the requests on to the Oracle HTTP Servers using completely different ports. This setup has the following advantages:

- Actual ports are hidden from users.
- Users do not have to add the port numbers to the URL.

On UNIX-based systems, it is not mandatory to start Oracle HTTP Server with root privileges. Only root can start a process that uses a port less than 1024. However, for processes that use a port number below 1024, root privilege is required to start the process.

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

[Figure 9–1](#) also shows how Oracle HTTP Server distributes requests to WebLogic Managed Servers. For high availability, it is assumed that each pair of components (Oracle HTTP Server and WebLogic Managed Servers) exist on different host computers. In [Figure 9–1](#), the Managed Servers belong to the same cluster. To load balance across a set of Managed Servers, they must belong to the same cluster.

9.6 Oracle HTTP Server Protection from Failures and Expected Behaviors

There are two categories of Oracle HTTP Server failures: **process failures** and **node failures**. An individual operating system process may fail. A node failure could involve failure of the entire host computer that Oracle HTTP Server runs on. This section describes failure types and the systems or processes that take over if they occur.

Process Failure

Node manager protects and manages Oracle HTTP Server processes. If an Oracle HTTP Server process fails, Node Manager automatically restarts the process.

Node Failure

If an entire node fails, the load balancer in front of Oracle HTTP Server sends a request to another Oracle HTTP Server if the first one does not respond, or is determined to be failed through URL pings.

WebLogic Managed Server Failure

In a high availability deployment, Oracle WebLogic Managed Servers are part of a cluster. If one of the Managed Servers 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 Failure

Typically, database failures are an issue only when `mod_oradav` or `mod_plsql` is used. If this is an Oracle Real Application Clusters (Oracle RAC) database, the failure characteristics are determined by the defined Oracle RAC connection.

If client connection failover is configured, any in-flight transactions are rolled back, and a database reconnection is required.

If Transparent Application Failover (TAF) is configured, any in-flight database write is rolled back but an automatic database reconnection takes place and select statements are automatically recovered. In this scenario, TAF fails over select statements only; package variables are lost. TAF is a feature of the Java Database Connectivity (JDBC) Oracle Call Interface driver. It enables the application to automatically reconnect to a database if the database instance to which the connection is made fails. In this case, the active transactions roll back.

9.7 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 (located in the `DOMAIN_HOME/config/fmwconfig/components/OHS/componentName` directory) for each instance. 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 9.8.1.4, "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.

9.8 Configuring Oracle HTTP Server for High Availability

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

This section includes the following topics:

- [Section 9.8.1, "Prerequisites"](#)
- [Section 9.8.2, "Installing Oracle HTTP Server on WEBHOST2"](#)
- [Section 9.8.3, "Configuring and Validating the OHS High Availability Deployment"](#)

9.8.1 Prerequisites

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

- [Section 9.8.1.1, "Configuring the Load Balancer"](#)
- [Section 9.8.1.2, "Installing Oracle HTTP Server on WEBHOST1"](#)
- [Section 9.8.1.3, "Creating Virtual Host\(s\) on WEBHOST1"](#)
- [Section 9.8.1.4, "Configuring mod_wl_ohs.conf"](#)

9.8.1.1 Configuring the Load Balancer

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 the features that [Section 2.1.1, "Third-Party Load Balancer Requirements"](#) describes, including:

- Virtual server name and port configuration
- Process failure detection
- Monitoring of ports (HTTP, HTTPS) for Oracle HTTP and HTTPS
- SSL Protocol Conversion (if required)

The following sections describe steps to configure your load balancer:

- ["Configuring Virtual Server Names and Ports for the Load Balancer"](#)
- ["Managing Port Numbers"](#)

Configuring Virtual Server Names and Ports for the Load Balancer

For each application, such as `myapp.example.com`, configure the load balancer with a virtual server name and associated ports. In an Oracle HTTP Server installation, these virtual servers are configured for HTTP connections, which are distributed across the HTTP servers.

If your site is serving 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`

Managing Port Numbers

Many Oracle Fusion Middleware components and services use ports. As an administrator, you must know the port numbers that the services use and ensure that two services are not configured to 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.

9.8.1.2 Installing Oracle HTTP Server on WEBHOST1

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

Validating the OHS Installation

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

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

9.8.1.3 Creating Virtual Host(s) on WEBHOST1

For each virtual host or site name that you use, add an entry to the Oracle HTTP Server configuration. Create a file named `virtual_hosts.conf` located 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. For more information see "Wiring Components Together" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*

9.8.1.4 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 the guide *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. For more information see "Wiring Components Together" in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*

The following is an example of `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>
```

Note: If you use SSL termination AND route requests to WebLogic, then the following additional configuration is required.

In the WebLogic console, WebLogic Plugin Enabled must be set to true, either at the domain, cluster or Managed Server level.

In the Location block, which directs requests to the Managed Servers, you must add the following lines:

```
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.

These examples show two different ways of routing 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 prefixed by `/weblogic` to the Managed Server cluster located on Apphost1 and Apphost2.

9.8.2 Installing Oracle HTTP Server on WEBHOST2

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

Validating the OHS Installation

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

```
http://webhost2:7777/
```

9.8.3 Configuring and Validating the OHS High Availability Deployment

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

- [Section 9.8.3.1, "Configuring Virtual Host\(s\) on WEBHOST2"](#)
- [Section 9.8.3.2, "Validating the Oracle HTTP Server Configuration"](#)

9.8.3.1 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.

9.8.3.2 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
```

Configuring High Availability for Oracle SOA Suite

This chapter describes how to configure high availability for Oracle SOA Suite products.

This chapter includes the following topic:

- [Section 10.1, "Configuring Oracle BAM for High Availability"](#)

Note: For more information on SOA, see the following documents:

- *Oracle Fusion Middleware Installing and Configuring Oracle SOA Suite and Business Process Management*
 - *Oracle Fusion Middleware Upgrading Oracle SOA Suite and Business Process Management*
 - *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*
-
-

10.1 Configuring Oracle BAM for High Availability

This section describes how to complete high availability configuration for Oracle Business Activity Monitoring (Oracle BAM). There are additional steps you must take for Oracle BAM high availability because the product uses local queues to configure JMS system resources.

Note: You must complete all steps in [Section 7.7, "Configuring WLS JMS After Machine Scale Up or Scale Out"](#) before you start the Oracle BAM procedures in this topic.

This section includes the following topics:

- [Section 10.1.1, "Configuring Oracle BAM Managed Server JMS System Resources After Scale Up."](#)
- [Section 10.1.2, "Configuring Automatic Service Migration for Oracle BAM."](#)

10.1.1 Configuring Oracle BAM Managed Server JMS System Resources After Scale Up

Oracle BAM requires manual JMS configuration steps in addition to the typical configuration steps described in [Section 7.7, "Configuring WLS JMS After Machine Scale Up or Scale Out."](#)

Note the following when configuring JMS system resources for Oracle BAM:

- Procedures in this topic use the name *bam_server2* to represent the new Managed Server that you are adding. Replace *bam_server2* with the name of the Managed Server that your configuration uses.
- Pay close attention to the targeting of the BAM JMS system resources, particularly whether the resources are targeted to the Managed Server *bam_server2* or to the migratable target *bam_server2 (migratable)*.
- If the procedures in this topic do *not* specify a JMS system resource property, accept the default value.

This topic includes the following sections:

- [Section 10.1.1.1, "Configuring Oracle BAM Server JMS Server."](#)
- [Section 10.1.1.2, "Configuring Oracle BAM CQService JMS Server"](#)

10.1.1.1 Configuring Oracle BAM Server JMS Server

This section describes the steps to manually configure JMS system resources for Oracle BAM Server JMS Server for a new Managed Server.

To configure Oracle BAM Server JMS Server resources for a new Managed Server:

1. Create the JMS persistent store `BamServerJmsFileStore_bam_server2` with the directory `BamServerJmsFileStore_bam_server2` targeting to Managed Server *bam_server2*.
2. Create the JMS Server `BamServerJmsServer_bam_server2` with the persistent store `BamServerJmsFileStore_bam_server2` targeting *bam_server2*.
3. Select **JMS Modules** then select **BamServerJmsSystemResource** in the table.
4. Select the **Subdeployments** tab. Click the Subdeployment **BamServerSubdeployment** then select **BamServerJmsServer_bam_server2** in the JMS Servers section.
5. Click **Save**.

10.1.1.2 Configuring Oracle BAM CQService JMS Server

To manually configure the Oracle BAM CQService JMS Server for the new Managed Server:

1. Create a JMS Persistent Store `BamCQServiceJmsFileStore_bam_server2` with directory `BamCQServiceJmsFileStore_bam_server2` targeting *bam_server2*.
2. Create JMS Server `BamCQServiceJmsServer_bam_server2` with the persistent store `BamCQServiceJmsFileStore_bam_server2` targeting *bam_server2*.
3. Create the JMS System Module `BamCQServiceJmsSystemResource_bam_server2` targeting it to **All servers in the cluster**. Select **Would you like to add resources to this JMS system module** to add resources to the JMS System Module.

Note: In the following steps:

- The JMS Connection Factories and Queues that you create for this JMS System Module specify the **Local JNDI Name** (local name) only; leave the **JNDI Name** (global name) field blank.
 - All references to "Queues" are for "Queues" only, *not* "Distributed Queues."
-
-

In the following steps you add resources for this JMS System Module:

4. Create Subdeployment `BamCQServiceReportCacheSubdeployment_bam_server2` targeting BAM JMS Server `BamCQServiceJmsServer_bam_server2`.
5. Create Subdeployment `BamCQServiceAlertEngineSubdeployment_bam_server2` targeting the BAM JMS Server `BamCQServiceJmsServer_bam_server2`.
6. Create JMS Connection Factory `BamCQServiceReportCacheConnectionFactory`. Leave the field **JNDI Name** blank.
 - a. Set **Subscription Sharing Policy** to **Exclusive**.
 - b. Set **Client ID Policy** to **Unrestricted**.
 - c. Select **XA Connection Factory Enabled**.
 - d. Target the Connection Factory to the SubDeployment `BamCQServiceReportCacheSubdeployment_bam_server2`. To do this, select **Advanced Targeting** in the Create a New JMS System Module Resource screen. Select the SubDeployment from the pull down list.
 - e. After the Connection Factory is created, click on it. In the **General** tab, expand the Advanced section. Enter `queue/oracle.beam.cqservice.mdb.reportcache` in the **Local JNDI Name** field.
7. Create the JMS Queue `BamCQServiceReportCacheQueue`. Leave the **JNDI Name** field blank.
 - a. Set the SubDeployment to `BamCQServiceReportCacheSubdeployment_bam_server2`.
 - b. After the Queue is created, click on it. In the **General** tab, expand the Advanced section. Enter `queue/oracle.beam.cqservice.mdb.reportcache` in the **Local JNDI Name** field
8. Create the JMS Connection Factory `BamCQServiceAlertEngineConnectionFactory`. Leave the **JNDI Name** field blank.
 - a. Set **Subscription Sharing Policy** to **Exclusive**.
 - b. Set **Client ID Policy** to **Unrestricted**.
 - c. Select **XA Connection Factory Enabled**.
 - d. Target the Connection Factory to the SubDeployment `BamCQServiceAlertEngineSubdeployment_bam_server2`. To do this, select **Advanced Targeting** in the Create a New JMS System Module Resource screen and then select the SubDeployment from the pull down menu.
 - e. After the Connection Factory is created, click on it. In the **General** tab, expand the Advanced section. Enter `queuecf/oracle.beam.cqservice.mdb.alertengine` in the **Local JNDI Name** field
9. Create JMS Queue `BamCQServiceAlertEngineQueue`. Leave the **JNDI Name** field blank and SubDeployment set to **BamCQServiceAlertEngineSubdeployment_bam_server2**.
 - a. After the Queue is created, click on it and go to **General** tab.
 - b. Expand the Advanced section and enter `queue/oracle.beam.cqservice.mdb.alertengine` in the **JNDI Local Name** field.

10. Click **Save** and then click **Activate All Changes**.

10.1.2 Configuring Automatic Service Migration for Oracle BAM

Oracle BAM Managed Servers use per-server JMS services. These per-server queues are also referred to as "pinned" services, because they are pinned to a specific Managed Server, rather than to the cluster. To ensure that these server-specific queues are highly available and can fail over to another Managed Server, you must configure the servers for automatic service migration.

For more information about Automatic Server Migration, see "Service Migration" in *Administering Clusters for Oracle WebLogic Server*.

A data source and a leasing configuration is created during the configuration of the Oracle BAM servers, but you must manually enable automated migration for the specific Oracle BAM services, as follows:

Task 1 Log in to the Oracle WebLogic Server Administration Console

For example:

`ADMVHN:7001/console`


Task 2 Select the Managed Servers to configure for service migration

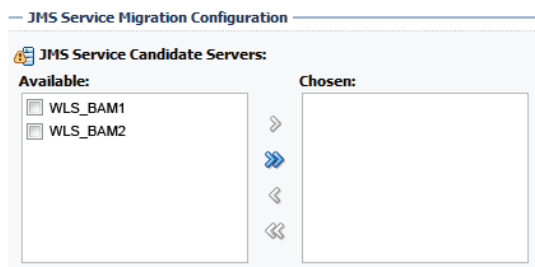
1. In the Domain Structure pane, expand the **Environment** node and then click the **Servers** node.

The Summary of Servers page appears.

2. Click the name of the server **WLS_BAM1** (represented as a hyperlink) in Name column of the table.

The settings page for the selected server appears and defaults to the Configuration tab.

3. Click the **Migration** tab.
4. Click **Lock and Edit** to start making modifications.
5. In the **JMS Service Migration Configuration** section of the page, select the **WLS_BAM1** and **WLS_BAM2** Managed Servers in the **Available** list box, and then click the move button  to move them into the **Chosen** list box.



6. Click **Save**.

Task 3 Set the service migration policy for each Managed Servers

1. In the Domain Structure pane, select **Environment**, then **Clusters**, then **Migratable Targets**.
2. Click **WLS_BAM1 (Migratable)**.
3. Click the **Migration** tab.

4. In the **Service Migration Policy** drop-down list, select **Auto-Migrate Exactly-Once Services**.
5. Click **Save**.
6. Repeat steps 3 through 5 for the WLS_BAM2 Managed Server.

Task 4 Manually Target the UMS JMS Server and Persistent Stores to non-migratable targets

Oracle User Messaging Service (UMS) 12c (12.1.3.0.0) does not support service migration. As a result, you must manually retarget the UMS JMS server and persistent stores to a non-migratable target; otherwise, service migration for Oracle BAM will fail.

To target the UMS services to non-migratable targets:

1. In the Domain Structure window, expand **Environment**, expand **Services**, and click **Persistent Stores**.

The console displays a list of persistent stores. In the list you will see a set of UMS persistent stores names called UMSJMSFileStore_auto_, followed by a number, one for each Managed Server.

<input type="checkbox"/>	UMSJMSFileStore_auto_1
<input type="checkbox"/>	UMSJMSFileStore_auto_2
<input type="checkbox"/>	UMSJMSFileStore_auto_3
<input type="checkbox"/>	UMSJMSFileStore_auto_4
<input type="checkbox"/>	UMSJMSFileStore_auto_5

2. Click the first UMS file store that is targeted to **WLS_BAM1 (migratable)**.
3. On the settings page for the selected persistent store, change the value selected in the **Target** drop-down menu from **WLS_BAM1 (migratable)** to **WLS_BAM1**.
4. Click **Save**.

If an error occurs, explaining that the JMS server is not targeted to the same target as its persistent store, then you can ignore the error.

5. Repeat steps 2 through 4 for the remaining UMS file store entries that are targeted to **WLS_BAM1 (migratable)**.
6. In the Domain Structure window, expand **Environment**, expand **Services**, expand **Messaging**, and then click **JMS Servers**.

The console displays a list of JMS Servers. In the list you will see a set of UMS persistent stores names called UMSJMSServer_auto_, followed by a number, one for each Managed Server.

<input type="checkbox"/>	UMSJMServer_auto_1
<input type="checkbox"/>	UMSJMServer_auto_2
<input type="checkbox"/>	UMSJMServer_auto_3
<input type="checkbox"/>	UMSJMServer_auto_4
<input type="checkbox"/>	UMSJMServer_auto_5

7. Click on the first JMS store that is currently targeted to **WLS_BAM1 (migratable)**.
8. On the settings page for the selected JMS server, click the **Targets** tab.
9. Change value in the **Target** drop-down menu from to **WLS_BAM1 (migratable)** to **WLS_BAM1**.
10. Click **Save**.

11. Repeat steps 7 through 10 for the remaining UMS JMS Servers that are targeted to **WLS_BAM1 (migratable)**.
12. Click **Activate Changes**.

Note: If automatic service migration occurs, and the required Oracle BAM services are automatically migrated to another Managed Server, note that those services will remain targeted to the failover server, even after the original Managed Server is back online and has rejoined the cluster.

Oracle recommends that you manually fail the services back to the original server. For more information, see the topic *Failing Back Oracle BAM Services After Automatic Service Migration* in *Enterprise Deployment Guide for Oracle SOA Suite*.

Configuring High Availability for Other Components

This chapter describes information unique to certain component products.

For this release, this chapter includes the following topics:

- [Section 11.1, "Deploying Oracle Data Integrator"](#)
- [Section 11.2, "Deploying Oracle Application Development Framework"](#)

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 Repository Connections to Oracle RAC"](#)
- [Section 11.1.3, "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 Repository Connections to Oracle RAC

When you create an ODI repository using the 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 [Example 11-1](#).
- If Oracle RAC is configured with SCAN, provide Oracle RAC instance details with the SCAN address.

[Example 11–1](#) shows the JDBC URL format to connect to an Oracle RAC with two hosts when it does not use SCAN:

Example 11–1 JDBC URL Format (Oracle RAC Not Configured with 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 the topic "Creating a Work Repository" in *Administering Oracle Data Integrator* for more information.

11.1.3 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 uniqueness of the scheduler 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 crashed. 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 does not continue running the scenario for the next eight executions.

11.2 Deploying Oracle Application Development Framework

This topic describes information specific to Oracle Application Development Framework (ADF).

Topics in this section include the following:

- [Section 11.2.1, "Oracle JRF Asynchronous Web Services \(Pinned Service Behavior\)"](#)
- [Section 11.2.2, "Modifying the mdsDS Data Source URL"](#)

See Also: For more information on ADF, see the following:

- [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 information on Oracle JRF Asynchronous Web Services, see the *Domain Template Reference*.

11.2.2 Modifying the mdsDS Data Source URL

When you use a GridLink data source or a multi data source with a customer-provided ADF application, you must change the mdsDS data source URL in the Weblogic Server Administration Console after you start the Administration Server.

To modify the mdsDS data source URL:

1. Log into the WebLogic Server Administration Console. Select **Services** then **Data Sources**.
2. Select **mdsDS** then **Connection Pool**.
3. Change the JDBC URL to specify it in the format:

```
jdbc:oracle:thin:@hostname:port/service-name
```

For example, if the JDBC URL is:

```
jdbc:oracle:thin:@(DESCRIPTION=(ENABLE=BROKEN)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=slc00erterw-r.us.example.com)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=srv2_koala.us.example.com)))
```

Replace the URL with the following:

```
jdbc:oracle:thin:@slc00erterw-r.us.example.com:1521/srv2_koala.us.example.com
```

4. Save the changes then restart all servers, including the Administration Server.

